IBM 801 Microprocessor Oral History Panel

Frank Carrubba, Peter Markstein, Richard Freitas, Rich Oehler, and Vicky Markstein

Moderated by:
Douglas Fairbairn and Robert Garner

Recorded: October 27, 2014
Mountain View, California

Cisco Telepresence

CHM Reference number: X7409.2015

Fairbairn: My name is Doug Fairbairn. I'm here with Robert Garner and we are going to be interviewing several of the key participants in the IBM RISC 801 program that took place during the 1970s. The first step of this interview is we're going to ask each of the participants that is Rich Freitas, Peter Markstein, Vicky Markstein and Frank Carrubba and Rich Oehler to give us a little bit of their own personal background so that we'll understand sort of how they arrived on this project and what kind of background and perspective they brought to it. So this oral history is being recorded at the Cisco Studios in San Jose and in Albany, New York. It's being done on behalf of the Computer History Museum in Mountain View. The date is October 27, 2014 and with that I think we're ready to begin. So I'm going to ask Rich Freitas who is sort of primary organizer of this activity to give us a brief personal background and how he arrived on the 801 project.

Freitas: Okay. My name is Rich Freitas. I got my Ph.D. at the University of California Berkeley in 1976. At Berkeley I worked on two research projects, first on Project Genie as a student researcher which was a time sharing system. And later as both a graduate student and a full time staff member of the university I worked on the Computer System's Research Project which was building a fault-tolerant time sharing system. In '76 I left Berkeley to join Frank's group at Yorktown Heights and there I worked on the IO portion of the machine, maintaining the architecture documents, the design of the various IO devices, interconnects, 150-megahertz optical link, things of that nature.

Fairbairn: Okay. Thank you, Rich. That's perfect. Peter, can you join us and tell us about your background and how you arrived on the project?

Markstein: Sure, my name is Peter Markstein. I had done my undergraduate work at MIT in mathematics and then joined IBM in 1959 where I worked on various research activities. I guess my favorite one at the time was operating systems for the Stretch computer. Then at the same time I worked on my Ph.D. which I got from NYU Courant Institute in operating systems. But NYU was very strong in optimizing compilers, that's because of its director Jack Schwartz. So I certainly got a good bit of training in that too. Eventually I worked on computer security. And then when the 801 project started I was recruited to work on providing optimization for compiler for the PL 0.8 language which was a language we developed especially for this project. I also ultimately did some architecture work mainly in the floating point, on the floating point side.

Fairbairn: Okay. Excellent. That's a perfect summary. Thank you. Now Vicky Markstein is there with you as well. So there's obviously some connection. Maybe we ought to cover how that connection happened along the way, Vicky.

<group laughter>

Markstein: It happened before I joined the project. Actually, NYU is our common link. And Jack Schwartz was my thesis advisor and he was in optimizing compilers. He wrote a lot of complicated-- and that's where I met John Cocke. And John Cocke convinced me that coming to IBM may not be such a bad thing and it wasn't. And I worked for him for the next fifteen years, I guess after that. So it was a long stretch. And what did I do for the project? Various things, a lot of things that other people didn't want to do, write code that people had tried and didn't like it. But mainly I did whatever John asked me to do which could be all over the place. So I had a great time. I learned more than I could have learned any place else. And the greatest time I had was the people I worked with. Everybody was very congenial. They all helped each other out. And the cooperation was wonderful. I only came across competition with other projects at IBM when we got our project going. It was amazing how many people had other projects and they were the competition but it was within IBM. So we could live with that.

Fairbairn: Okay. Excellent.

V. Markstein: That's it.

Fairbairn: Thank you very much, Vicky. And Frank.

Frank Carrubba: I went to the University of New Haven for my undergraduate in electrical engineering and then later for my master's in operations management. In my junior year I was recruited by IBM to join the company [in Endicott, New York]. At Endicott I worked on the 1401 Model E follow-on for a short period of time. There were a number of problems with the already released1401 at that time. The indexing feature was having intermittent problems. They were also having intermittent problems with the memory and they asked me and one other person to help clean things up, which we did. After a short time on the 1401 program, I was sent to New Haven, Connecticut as an Area Specialist. About that time, the Atomic Energy Commission, IBM and Yale University were chatting about a project that would fully automate the world's first Tandem Van de Graff accelerator. I was approached to help design the computer system and interfaces that would accommodate the type of low energy physics experiments being envisioned. So anyway they pulled me off of the area specialist position and sent me to Poughkeepsie and stuck me into 7074 program, the computer system that was going to be used by Yale University to control the accelerator and gather data. Not too long after, they realized that the 360 model 44 was coming along and move me over to that program. When that was over the people at Yale, Dr. Allan Bromley and Dr. Joel Birnbaum in particular, asked IBM if I could be transferred to Yale for a period of time during the bring up and the debugging of the system. I met with Allan Bromley and Joel Birnbaum. At that time, Joel was a prior graduate student of Allan Bromley and he stayed on at Yale to work on this project. Joel and I hit it off very well. I liked Joel a lot and when he asked me to come work with him I did just that. Joel was responsible for the software effort together with Martin Michaelson, Phil Summers, and one other person. I, together with the Yale University engineering team, successfully automated the Tandem Van de Graaff and its state of the art data collection system. From that experience, when the project was ended Dr. Rolf Landauer and John Cocke from the IBM Watson

Research Center came to visit the Yale Project at, I believe, Joel's request. And when they showed up John Cocke pulled me off in a corner and couldn't stop talking about the custom hardware and the IBM mainframe and how we adapted each of them to do the kind of physics experiments that were being envisioned. John was particularly interested in how we interfaced to the IBM 1810. He was very, very interested in all of the details of the digital analog converters, and all of the various special interfaces that were designed specifically for that project. I had never met John Cocke before. He really impressed me as a man that was a true renaissance person. He could talk to you about almost anything. When the Yale project ended Joel Birnbaum went back to Yorktown and shortly after I finished up, Joel asked me if I wanted to come work for him at Yorktown. After many discussions, I decided to transfer to Watson Research and stayed there for a good number of years. From 1969 to 1981 to be exact. I actually joined IBM 1960. At the labs I first got involved with laboratory automation. We were going to automate all of Watson with a single lab automation and data collection platform very similar to what we developed at Yale. I worked very closely with a fellow by the name of Bruce Gavril. Bruce was a really great control guy and digital engineer. I probably did some of the more creative work during that period. That project took off and it wasn't long before we realized that if we had to keep that project going we needed optical links in order to move the data between the laboratory and a central computer system which was being installed. So I started working on optical fibers and glass fibers that were filled with trichloroethylene and that was a very interesting piece of work as well. And from there I got involved with a project called BOSS business oriented small systems. John Cocke found out that I was doing that and he said, "What in God's name are you doing business systems for?" And went to Joel Birnbaum and Herb Shore, who was the head of the Computer Science Department and he said, "Get Carrubba out of that project and have him report to me." So Herb and Joel agreed. They could not imagine anybody reporting to John. And he said, "Well, I'm an IBM Fellow and I deserve someone to work with me." So he got me to come over and we started on the telephone project. The telephone project was a very important project for IBM. The Company had already created a product that was to be announced called Rosebud that was done at the Los Gatos Labs in Silicon Valley. That project had a number of issues in terms of performance and cost and eventually was ended. But in the process we began to study how they were doing switching and how they were handling all of the dial signals and various other signals that are necessary in making phone calls. So John got a hold of Al Chang and John and Al Chang started analyzing the telephone networks and response times and so on that were necessary and a number of lines of code that would be necessary to deal with each line and each call. And they came to the realization that even if Rosebud was announced it wouldn't have been able to handle very many simultaneous calls. So we started looking at the main processor. And the main processor was a skinny down machine that was basically designed with load stores, add, subtracts, you know, very, very simple branches and testing branch instructions. And we were able to get the cost performance of that machine down to a very, very good level and it was able to handle a larger number of calls at a time and it was really a step in the right direction. So we started using that machine as a vehicle to talk more about why IBM should get into the telephone business. And Al Chang did a lot of analysis along with John and many other people that John corralled into doing the various pieces of work that were necessary. And then he convinced Joel Birnbaum that he should bring the project up to Ralph Gomory. They did that. Ralph liked it. He allowed us to continue working on it and then suddenly everything stopped. There were several rumors why they stopped the project. I don't want to go into them here because I don't know which of them were true and which were not. But the project stopped. At that point John said to me, "I really like this little computer. Let's keep going forward with this because I think it could possibly have general purpose application. So he got a hold of Bill Worley, who was out in Palo Alto at the time. Bill and John, together with Al Chang, and a few other people got involved with gathering up trace tapes from the IBM libraries and starting looking at the mix of instructions being generated by the various compilers, the frequency of execution and so on. From there, they decided that there was strong possibility for this little computer and, therefore, decided that we would start looking at the machine architecture needed to make it a general purpose machine. During that time John and I took a number of trips to California. We talked to people like John Backus about applying map coloring algorithms for use in compiler register allocation. We talked with Seymour Cray about high speed simple instruction machines and how signals, with fast rise times, were being sent around large configurations of hardware. We talked with Gene Amdahl about bipolar logic why he did what he did with the Amdahl machines. And then we talked with Dave Patterson a number of times. We also talked with John Hennessey a number of times. We talked with Ken Thompson from Bell Labs. And we would sometimes meet them either at Rosita's over in Berkeley or we would meet at the Oasis in Menlo Park. These people would come for lunch and we would have this really great discussion and John would basically be leading the orchestra. And I think John basically did this because he really wanted to have a sounding board for his ideas. He would put ideas on the table and everybody would either support them or criticize them and John would go away, get on the airplane, and we'd talk about his ideas all the way back home. And after returning to Watson Labs, we would either refine some things or John would say those people are not thinking clearly and let them do whatever they want to do. So, our last meeting in California was to see Bill Worley at his Page Mill Road office. The purpose of the meeting was to complete a first draft of the 801 Architecture. We started at about two thirty in the afternoon talking about the instruction set and we didn't finish until about three o'clock the next morning. From that meeting came a preliminary document that Bill produced, describing the architecture, the types of machine instructions and a first pass at a data flow that would execute every instruction in a single machine cycle. We wanted to make the configuration as simple as possible in order to achieve the fastest instruction execution time. We also wanted it to be the best cost performing computer we could build. The document was then sent to the Watson Labs team for review and discussion. The team then gave Bill its feedback, he made the necessary corrections, and sent a final document to John and to me and I think he may have sent one to Allen, what was her first name.

V. Markstein: Fran Allen.

Carrubba: Yeah, Fran Allen, I couldn't remember. But we each went through the final draft and started doing various pieces of work necessary to get a prototype complier and machine design underway. In parallel, John went to argue about money, budget money, schedule and so on. He convinced Birnbaum and Ralph Gomory that a prototype of this architecture should be built. During that time, I was busy with a first pass of the data flow. We got the budget approved and I started hiring. I brought in people like Paul Stuckart to do the system configuration design, including method of interconnect, power distribution, and a logic board cooling system that had to be very, very different than we had ever designed.

Fairbairn: Can we interrupt?

Carrubba: Sure.

Fairbairn: We're getting to the kick off the project and I think you've done a great job of summarizing. Before we got into the details I'd like to sort of finish this first step with getting each of the participants identified and what their background is and we have just Rich to go and then I'm going to turn it over to Robert and sort of finish off the kick off phase and then get into more details of the project itself. So Rich could you just give us a quick background? Rich is in Albany, New York and tell us how you arrived on this project.

Oehler: Sure. Let's see. I finished my undergraduate work in '62 from St. John's University, Queens, a city boy. And I was out looking for a job and I tried IBM and they said no. And I went to NSA. So I put on my white hat and went charging over the hill to try to write algorithms to help with crypto. That lasted about three years. And I was then moved into an R&D section inside of NSA to work on computers, everything from compilers to hardware and it was a team of about four people. And given that it was NSA there was a broad range of equipment that we could try ideas out including a lot of plug board computers. So that lasted until '70 and I reapplied to IBM and they invited me in. So now I think the reasons for that were that I really understood how the software worked with the hardware and did a lot of microcoding, did a lot of very level low level programming. I'll tell you about the simulator during the rest of this talk. But the bottom line was just getting warmed up in IBM and there was something called future systems that came along. And I was an IO manager in that group and I think I demonstrated my worth in IPLing and error handling and several other things that people were interested in. So they were paper studies but they got enough interest that when FS Collapsed, I was invited into the 801 team. And, again, I started out as IO man. So Rich and I worked quite closely and the higher level functions and Rich did a lot more of the low level function. So anyway from that early beginning I spent the next 20, 23 years or so working on the 801. And, again, I would go back and like to compliment John Cocke who's really the father of all of the ideas right here. And George Raydon who is unfortunately no longer with us where George did a great job of nurturing the team he had and keeping us on track and fending off the budget mavens and all of the other things he did. So without him I think we would never have gotten off the ground.

Fairbairn: Alright, thanks, Rich that's excellent. I'm going to turn it over to Robert Garner who is also an IBM veteran but not on this project but has a good perspective and he'll sort of further take the questions from here in terms of finishing off the startup of the project and what it was actually like to go on and work in that project.

Garner: Thank you, Doug. I guess I'm not sure I'd call myself a veteran compared to you guys. As some of you know I started off working at Xerox, Xerox PARC and then Sun Microsystems and then came to IBM in 2001. I was the architect of Sun's RISC effort. And so the early beginnings of the RISC work you guys did at IBM has been totally a fascinating bit of interest to myself and to Silicon Valley. So I'm very grateful you guys are available today for this interview. Frank, I'd like to, if I could since you had a lot of the early history and then I'll go around to everyone, I'd like to get a sense of the very beginning that you talked about. Frank, you mentioned the origins of the project and what you started as a telephone project and then there was the Rosebud project in Los Gatos that you guys reacted to. My first question on the Rosebud project was it a micro code based machine or was it-- why as it so slow at handling telecommunications machine?

Carrubba: It was a PBX system.

Garner: Okay. So it was all hard wired?

Carrubba: It was all hard wired.

Garner: Okay. And did you have a sense then of why it was so slow, just slow circuits?

Carrubba: The computer was one of the problems. The computer was a microcode driven system and it was bogged down a lot by churning through data path that wasn't necessary.

<crew talk>

Garner: Okay. I think he heard it. My question was, was the Rosebud system that you guys were looking at which was being done at the Los Gatos Lab because you were doing your telephone project in Yorktown, did it contain microcode and you said it was a PBX so there was certainly telephone switching electronics but it sounds like the control computer was microcode based.

Carrubba: Yeah. And there was a lot of special hardware as well that was the interface between the telephone circuitry and the lines, telephone lines coming in and leaving the system.

Garner: Sure. But it sounds like the computer itself was rather slow because it was microcoded. Why do you think they made the choice to do microcode? Back then there was a lot of talk of kind of merging-- well, first of all there's Maurice Wilkes and there was Fred Brooks and the 360 certainly took a microcode based approach. Did the guys in Los Gatos feel the microcode was the only way to build a computer? Was that kind of the general thinking at that time in IBM?

Carrubba: It's hard for me to answer that question because I really wasn't there when that decision was made. The system was up and running in the labs when we went there and it was under tests. So those are decisions that the group made long before I ever got there.

Garner: Right. Right. Now, let's talk about Stretch for a little bit, Stretch was obviously a hard wired control machine, an incredibly brilliant designed architecture. Actually, even though it was physically very large, very few circuits were actually used to implement it. And it sounded like one of you had experience on Stretch. I think that was…

P. Markstein: Time spent was me; that was Peter.

Garner: Peter. And Did John Cocke-- I mean my basic question for you guys is what really inspired him to go a simple load store architecture for your telecommunications system. You know, he had some experience with Stretch as well. We all knew about the early computers which are very simple load store architectures, the 700, the 7000 series. The mini computers that were coming out were also very simple load store architectures. What motivated him to go that way and not go towards microcode? Maurice Wilkes had been promoting microcode. Fred Brooks had been promoting microcode. Did he have any feelings about that use of microcode John Cocke?

P. Markstein: Well, yeah, he was very much against it. Certainly, he realized that microcode machines actually could do the most common instructions of the machines they were simulating in one instruction. And so for, say, a simple add register that would be one cycle. But then you had to run the simulator under the covers, that was another nine cycles. So even for the simplest instruction you were taking a ten to one degradation.

Garner: He had this realization from very early on then before 1974 on the telecommunications project?

P. Markstein: Yes.

Garner: Okay. And he obviously was going against the grain of the whole 360 juggernaut I assume.

P. Markstein: Yeah, he was saying in 360, except for the largest most expensive realizations they were all done with microcode. And he realized that a lot of performance was being thrown away and he thought well let's just have-- let's expose basically the microcomputer and make that the interface for programming. He realized perhaps the number of instructions you'd need to get a job done would be larger than for the common architectures of the day, 370. But that would be ameliorated by having a good optimizing compiler which he was a big advocate of. And it turned out he was absolutely right.

Garner: Yeah. Okay. Frank, your next step…

Carrubba: Could I just add to that for one moment?

Garner: Yes.

Carrubba: The other thing was that even though there were single instruction types of machines early on the data paths in those machines were encumbered. There were things in the data path that were used for higher level instruction execution that-- or lower level instructions that needed assists. And because of that, being able to pipeline the data flow, being able to execute a complete instruction in one single machine cycle, was not possible. With the 801 the way we designed the architecture and the data flow, we were able to pipeline. We were able to execute one instruction each machine cycle and we got incredibly good performance.

Garner: Yeah, certainly the microcode engines could do one micro instruction every cycle. And, you know, what was remarkable about the microcode machines was that the micro store was so slow. They typically occupied a half to a third or a quarter of a machine cycle time. But Frank I wanted to ask you about your early-- you said during that wonderful introduction to the 801 project you said you went out and talked to some other people and I wanted to try to get some dates on these if you remember the dates?

Carrubba: Would you repeat that, again? I'm sorry.

Garner: Well, you said that you went out and talked to some influential people outside of Yorktown. I wanted to get some of the dates. You said that you went…

Carrubba: In the early seventies, '72, '73 in that timeframe.

Garner: Okay. So you spoke to John Backus, Seymour Cray. You went to Chippewa Falls?

Carrubba: We went to John Backus. He had an apartment up on Twin Peaks. We went to his apartment. We also went to-- he had an office at Stanford at the time and he had an office at IBM. And we went down to the IBM office at least twice and up to his home once. And the discussions were always the same thing, you know, having to do with register optimization, register allocation, how many…

Garner: Did he give you a sense of how many registers he thought were required?

Carrubba: How many-- I was just going to say that, how many registers would be necessary? Sixteen, thirty-two, sixty-four, single bank, double bank; you know, we talked about all of those things. And the reason he got together with us on this topic was that John Backus had interfaced with a student who had, at that time, created a very clever algorithm for map coloring. And that's why we went to see Backus to begin with.

Garner: Who was that student?

Carrubba: I don't remember. I tried to remember over the weekend, I wasn't able to. But the discussion started in that way. And John Cocke explained to Backus exactly why he was interested in a map coloring algorithm. And since Backus was involved with optimization in his past it was very easy for him to pick up on what John was trying to do. And he believed strongly in graph coloring could apply.

Garner: Did John also suggest to have three operand fields for registers two sources and one destination? Or was that not on his list of…

P. Markstein: That came a little later.

Oehler: But it was on his list.

P. Markstein: Perhaps. But the first 801 was more traditional in that there were two operands a register, two registers. One of the registers got destroyed as a result of the operation. We realized that that, in fact, wasted some data that is the old data that was in the destroyed register. And when we did the second version of the 801 we went to nondestructive operations.

Garner: Why was John interested in these kind of compiler optimizations? And I tend to think of him as certainly Mr. FORTRAN but then APL was he studying the compiler optimizations for what language at the time?

P. Markstein: I think he was interested in a more abstract way. John Cocke – Cocke and Jack Schwartz at NYU wrote a book called "Higher Level Languages and Their Compilers." It was published by NYU. It must be 600 pages thick. And most of it, in fact, deals with optimization rather than high level languages.

Garner: You said that was John Cocke and who?

V. Markstein: Jack Schwartz.

P. Markstein: Jack Schwartz from NYU.

Garner: So I was asking my question of John Backus.

P. Markstein: Oh, John Backus.

Garner: Yeah.

P. Markstein: I just said John and I…

Garner: Yeah, I know we have two John's I do realize.

P. Markstein: But yes, of course, John Backus had done the original FORTRAN compiler and the objective there was to try to produce code that was comparable to hand coding. He came pretty close to meeting that objective although I'm not sure what the register allocation technique was. There were only three registers to allocate so that was…

Garner: Did he have his own paper design to compile for because certainly not for the 360. He must have had some machine inline.

P. Markstein: He compiled for the 704.

Garner: I see. Okay. That makes sense. So he was still working on that at the time in 1972.

P. Markstein: Well, I think he'd gone on from…

Carrubba: That was over.

P. Markstein: That was over and done with.

Garner: But you knew he had a lot of background from that work. Okay.

P. Markstein: Yeah.

Garner: And then Frank you said you visited Seymour Cray in Chippewa Falls?

Carrubba: No, here in Silicon Valley.

Garner: Oh, in Silicon Valley.

Carrubba: Yeah.

Garner: And the reason why was just to…

Carrubba: Well, we were talking about how we were going to implement this little machine and the path lengths and keeping it to a particular cycle time. And we realized that the transmission of signals from one board to the other was going to become a hindrance to us. So Seymour had that great machine approach that looked like a cocktail lounge.

Garner: This is the Cray 1 you're referring to.

Carrubba: And it was a round leather covered mainframe it looked like a banquette almost and it had this large back on it in the middle. And if you look at some of the end views of it looked like what people interpret as a flying saucer.

Garner: Yeah.

Carrubba: So the machine was built in a complete circle. And everything in the middle had to do with interconnect, but the interconnect was extremely short and consistent from board to board no matter where you were in the machine cycle, whether you were doing IO, CPU cycles, memory cycles. You know, the distance between the one or two boards was basically the same.

Garner: Yeah. So did that inspire your design for the 801 which was also circular?

Carrubba: It inspired me to think about that and discuss that approach with Paul Stuckart as a way of keeping the signal lines as short as possible. The problem was that in order to move those signals from board to board, and not lose rise time and cause delays as well, we would have had to use coaxial cable for all of the interconnect. And instead what we did was we discovered a company who wove cables on a Jacquard loom. They were able to weave the cables to the impedance, exact impedance that we needed to get the rise times correct and to have a minimum of delay. Plus, there were hardly any cross talk problems to speak of.

Garner: Do you remember the name of the company or where they were located?

Carrubba: No. Do you?

Oehler: No.

Garner: Okay.

Carrubba: You're giving me some things to think about. I'll think about them.

Garner: Alright. So next you said you also spoke with Dave Patterson. Why Dave Patterson?

Carrubba: As I said, I think, first of all, John Cocke respected Dave and I think as I said to you a lot of our trips had to do with John creating a sounding board for his ideas and to try to get people to punch holes in them or to propose alternatives to his ideas. And I don't remember the exact discussion we were having with Patterson but it's been a long time. And I think John went to see him because he wanted to have that type of discussion with him.

Garner: Do you remember what year it was? This must have been later than 1972.

Carrubba: That was just before we started. That was probably early '74.

Garner: Really? That early. Early 1974 because Dave Patterson and his own RISC project really didn't take off at Berkeley until 1980, 1981. So that's a whole six years earlier that you and John met with Dave.

P. Markstein: I think those discussions continued and after the 801 project got going there were, again, visits to the West Coast. I wasn't part of it also maybe you can say Vicky.

Oehler: Yeah, I was.

V. Markstein: Who was?

P. Markstein: Rich. There were discussions in which most of the aspects of the 801 as we then saw it and it was getting pretty firmed up were discussed it was interesting because IBM which is usually so secretive about new things said, "Yeah, you can talk about anything you want except the register allocation." That they realized was pretty key we were not allowed to talk about that but everything else was on the table.

Garner: Okay. So Vicky, you wanted to say something and also Vicky you worked so closely with John maybe you can tell me about his motivation on the 801 too.

V. Markstein: Repeat the question, please.

Garner: Well, you were about to say something. And then also if you could say more about John Cocke and his motivations and how he got you excited about the project.

V. Markstein: Oh, well, he had a very persuasive type of speech, I think. He talked about his machine it was like no other machine. And it was at NYU and my mentor at the time was Schwartz. And Schwartz had a lot of respect for John. So everybody I knew thought John was quite good. So when he offered me a job at IBM I jumped it. I said yeah I think I'd like to do that for a while. But I was going to talk about my trips to the West Coast. He sent me over to talk because I still-- and I met [John] Hennessy for the first time.

Garner: What year? About what year?

V. Markstein: About '80, it was the early eighties.

P. Markstein: About 1980. It wasn't much later.

V. Markstein: Yeah, it was about '80, 1980 and I brought a few people from the project with me. One was Greg Chaitin who was the big register allocation guy. They were the brains behind a lot of it. And we were not supposed to talk about register allocation. So my job was to be careful that Greg Chaitin didn't let anything out. We could say anything else we wanted but not register allocation which, by the way, I guess Sun were the people who went-- you might know this, never were really able to do it.

Oehler: That's correct.

Garner: Yeah, that's correct. Yeah, we did register windows instead.

V. Markstein: Register windows was quite difference.

P. Markstein: Windows was the…

Oehler: The work of the devil.

V. Markstein: A work around. <laughs>

Garner: It worked though.

V. Markstein: I guess IBM was right. We didn’t let that cat out of the bag. But IBM never thought they would do anything with this computer. As a matter of fact, it was becoming a sore on their side because they kept saying, well, this is great. You’re going to go out and put out a computer for $15,000 that’s faster than our $25,000 a month computer. How are we supposed to make money? And I said, that’s for you to know. And if you don’t know, you’re in the wrong business.

Garner: Thank you for that. Okay.

Garner: All righty. I think that covers my questions about Frank's timeline there. And I thought might be good would be to, frank, for you to maybe continue on the building of Data one, how you built the team, and then everyone-- but then after that we can go around and people can talk about their contributions. Does that sound like a good idea?

Carrubba: Yeah. After we did the initial data flow, and started to think about the I/O and the memory interface, the project basically started making some changes, because the hardware and software people actually started getting their teeth into the design. So we would sit down, even though we were trying to get to the point where the design was frozen, and talk about some of these possible tradeoffs, and either weed them out as something that would be nice to have, but not necessary, or something that we overlooked and absolutely needed to go into the machine. So after having a number of those discussions, and most of those took place between George Raydon, myself, Mark Oslander, and Marty Hopkins . And, because of these discussions, the data flow and/or interfaces would change somewhat. Since we had not yet committed the design to build, we were able to make some changes during that period of time. Which we did. And then after that, we started talking about how to package the hardware? What kind of logic do we use? What cycle time are we looking for? What logic could get off the shelf that would give us the performance we were looking for? What types of I/O do we anticipate would be necessary to, not only prove the machine's complete architecture, but also to have system that would do some real things that that could be measured. In addition to that, we were talking about, "How do we test the machine?" And I that’s when I hired Willard Borscious, who was on Service Free, I believe, to do all of the systems exercising programs, and fault locators, because we really wanted to be able to, once we had the hardware up and running, debug it as fast as we could so that the software people could get their hands on it. So in parallel with all the design work and everything, Willard was designing a set of programs to exercise every element in the machine, and to come up with a set of fault locators that would help us if we ended up having to debug during a very busy period. The mainframe was Paul Stuckert's design. We designed it around the ideas of the Cray machine, where everything pivots from a single point. And opens and closes like the pages of a book. That was the whole idea. Then we realized that once we selected off the shelf emitter coupled logic, chips placed on custom boards, the heat generated would have been really overwhelming. So we had to come up with a cooling system that would complement the package and at the same time give us the airflow we needed to cool the logic. Paul Stuckert, being a very inventive person, one of the most patent recognized people in IBM at the time, came up with a scheme for cooling that was very different than anything anybody had ever done before. And he fashioned it after how they moved oil through the pipelines that come from Alaska. And that cooling system was not only silent--unless you were a bat or something, but it also moved air in a very linear or turbulent way. And it depended upon where the logic chips were mounted on the board, and whether the air foil would be laminar, or if it would be turbulent. So we factored all of these things in when we did board layouts and everything. So the cooling system in the package was patented. It was something nobody had ever done before. And logic was off-the-shelf logic, but in order to package it in a large way, the way we had to with the 801 was something that was hardly ever done before. People just created a single board of this stuff and we had to connect a large number of boards using coaxial cables across a flexible backplane — along with miniature coaxial cables on the boards, especially on the faster parallel busses. And we had to do employ a lot of tricks in order to preserve rise time, and distribute the clocks. We had delay lines everywhere that we would tap in order to get clock alignments on a board and from board to another. It was quite a machine to design and to debug.

Garner: Can I ask a question right there?

Carrubba: Sure.

Garner: How many people did you have on the hardware design team?

Carrubba: Well, I had myself, Ken Case and we mainly did the CPU. Then we had Norman Kritzer who did the memory controller. He and I worked together, but he actually put it on paper. Then we had Paul Stuckert, as I said, who was the frame designer, the power system designer, the power distribution designer along with the cooling system. And then we had Rich Freitas and Peter Karnazes, who were working on the I/O.

Freitas: You didn't add in memory.

Carrubba: Memory, I'm sorry. Memory and I/O.

Garner: A pretty small team. So you know, I did see the cooling. You have a piece of plastic over the board and compressed air pushes air up over the ECL] chips.

Carrubba: Yeah, we wanted to create a control stream so it wouldn't interfere with the board next to it. And we wanted to have a source that would cool the board in an even way, from one end to the other, front to back. So we didn’t want to put-- we couldn't put any blower motors in it, because if we did, it would have been impossible to open and service the system. So what we did was we created a pipe and that was sealed at one end, and had a series of holes drilled every three-quarter-- oh, no-- every half-inch, by laser. And they were all drilled in a straight line, from front to back. At one end was the part where the air source came in through a fitting, and pipe slid into a phenolic funnel section that was shaped like the letter V, and this pipe was set right in the middle of the V. The bottom of it had an air filter that that allowed clean outside air to brought in from the outside from the outside. So we got a very high-end compressor from an aquarium in Manhattan, and we put the compressor way down inside the power bank, and we pumped air to each of these pipes. And the pipes when the air was released through the orifices that were laser-drilled and pointing straight up, would then entrain outside air to come up through the filter material, the phenolic V shaped funnel, across the pipe itself, and then up across the board's logic with that plastic with a plastic cover controlling where the airfoil spread out to.

Garner: Okay, Frank? Can I ask you--

Carrubba: And then the air would escape from the top, which was totally open.

Garner: Yep, yep. Can I ask you another question? About what year do you think the machine was fully available for the software folks?

Carrubba: Till the machine was full--

Garner: How long did it take from-- I guess. How long did it take from when you started to when the software guys were running a program?

Carrubba: Well, we started in late '74 to really talk about pieces of it, hardware pieces. We started actually building it in '75.

Freitas: Late '75.

P. Markstein: Yeah, but, in fact, while you were doing that, we put together a simulator.

Carrubba: Oh, yeah.

P. Markstein: And the simulator, in fact, was very fast in its own right. In modern terms, you'd call it a "just-in-time" compiler. And it looked at a cache line of code, compiled it, so that simple instructions were just executed in one <inaudible> 370 instruction. So even in late '75, we could simulate the entire machine. We could take outputs from the compiler and test it out on the simulator. So we had a pretty good idea of how things would go. We started to get a good idea of which instructions, perhaps, we should tweak the characteristics of so that they would plumb better. It was actually because of the good, the strong simulator, good interaction between the software folks and the hardware people. So while it was a number of years before we actually ran on the physical hardware, we were running major pieces of code on the simulator from the beginning.

Carrubba: When did you finish the I/O? That was the last piece? Do you remember? '78.

Freitas: Somewhat. Whenever we had that demo for Akers and Opel. That was about the time when basically it was done.

Carrubba: That was '78.

Freitas: Okay.

Carrubba: So we had a running machine in '78.

Garner: Okay, so we had-- based on the simulation, though, Rich, it sounds like-- and Peter-- it sounds like we already-- you already knew that for some benchmarks programs it was going to outperform the high-end 370s.

P. Markstein: Well, let me tell you how extreme it was. So the language we had was called PL.8, PL-point-8, because it kind of paired away from PL.1, the more arcane things you couldn't implement in an effective way. So I remember that part of the optimizer used bit factors, which it turned out PL.1 had a terrible time with, because it wasn't sure where the words would spin instruction boundaries, or byte boundaries. It was an unbelievable amount of code.

Garner: Yeah.

P. Markstein: With what we compiled with PL.8 was so streamlined that when you simulated it on the simulator, it ran those routines faster than the native 370 with its production compilers.

Garner: The one with PL.1, you mean? With this PL.1.

P. Markstein: With PL.1, right.

Garner: Yeah, yeah. So that's a statement both about PL.1 and PL.8 and your instruction set and the 370. All four of those come into play there.

P. Markstein: That's right. But I mean, it was a very good feeling that we were on the right track.

Garner: Did anyone ever think of using a-- I mean, you chose PL.1 and PL.8, was there any thought of choosing some other language to compile, Fortran, or god forbid, __________?

P. Markstein: Well, ultimately when we went on to do the RS/6000, we got a Fortran compiler, and we made a C-Compiler for that matter.

Garner: Yeah.

P. Markstein: But initially, we got a lot of the work done in PL.8, as we called it. We didn't say the "point." In fact, there were 801 architectures that were made into controllers for various IBM machines, and they were largely programmed in PL.8.

Garner: That was a good choice, yeah.

P. Markstein: And then we made on the backends for the PL.8 compiler, we made a 370 backend.

Garner: Oh.

P. Markstein: We made a--

Freitas: 68000.

P. Markstein: A Motorola 68000. I think we produced more 68000 code than anyone else for a number of years, because we had this compiler that put out very good code.

Garner: Was that to support some kind of internal workstation project or something? Or why was that?

P. Markstein: No, we used the, in some cases, 68000 as engines for controllers. I don't remember which product.

Garner: I see, okay.

P. Markstein: And they were coded in PL.8.

Carrubba: We were also looking for cost-performance points. So we wanted to look across a broad range of hardware. And including that particular piece.

Garner: Okay. Okay, Rich and Peter, certainly with your OS background as well, Doug reminded me, I don't remember reading about any-- not this, Rich-- I mean, Peter. Peter, I don't remember any operating system. You just had simple ways to load programs in the PL.1. Was there any tiny OS you put for it?

P. Markstein: No, Rich?

Oehler: No, we-- yeah, we-- when I did the simulator, I went back to the difference between privileged ops and regular ops and put all that mechanism in. I put in the mechanism for caching, including all the invalidates that were needed. After all, the 801 had a programmed cache for all the invalidations that were going on.

Garner: Well, you had a separate instruction data. So you had to. Is that what you mean?

Oehler: Yes, but I had to-- in order to run a piece of code on it, you had to recognize that that code was going to be multi-threaded or not. Whether it was going to have any of these problems or not. Then you had to program it against the standard architecture, which is another one of the things that I added in for the instruction set of the OS. And so we had even working on some of the more traditional early Linux kinds of machines. And some people took it for the-- I can't think of the word right now, sorry. But yes. We did-- it wasn't done until I was able to run hundreds of millions of lines of code and run them under an operating system, and use the hardware mechanisms simulated in software to do all the data capturing.

Garner: So what data capture are you referring to?

Oehler: Instruction chains.

Garner: Instruction chains, okay.

Oehler: We built up-- we were the first ones, I think, to recognize what was happening with the benchmarking activities, and I went full board after running the benchmarks on our simulated hardware, so that we could go back in and analyze that cache behavior, instruction mixes, all the things that went along with that.

Garner: Right. But I was just wondering in the 801 project itself, was-- did you guys have an OS, a small OS that ran an 801? An operating system kernel?

Oehler: Yes, yes.

Garner: Who wrote that? Or--

Oehler: I did with one other person helping.

Garner: So, there was no virtual memory in the first 801? So it didn't have to be looked at.

Oehler: That is correct. That is correct.

Garner: It just loaded programs and scheduled I/O basically and __________?

Oehler: Right.

Garner: Okay.

Oehler: But according, you know, the way it was, I got as close as I could to emulating the behavior of a cache. Okay, this is being able to deal with problems like it was multi-threaded in places and I had to be able to coordinate all that with the storage. So, yes.

Garner: When you use the term "multi-threaded," so you mean in the modern sense? The hardware did not support multiple threads.

Oehler: That's correct. It wasn't until we got to the RS/6000 that we fully put in a multi-processor environment.

Garner: Okay.

Oehler: But we had to deal with the same issues again about memory addressing and making sure that we had the coherency correct.

Garner: Okay. Since I have you on here, Rich, do you want to talk about more what you did on the 801 project? Just freeform? Anything else you want to add?

Oehler: For me?

Garner: Yeah.

Oehler: Oh, yeah, the thing I was trying to say before was that I'm the guy who likes to build things and make them work. And so that was a skill that helped me-- helped the team get the first machines married with the hardware, with the software that we already tested. And from that, I went on to be the-- not the keeper of the RISC architecture, but the lead architect in the process. I had all of the people in the beginning was IBM, and then over time that stretched out to include Motorola and Apple in the--

Garner: Oh, the RS/6000 project, you mean? PowerPC. It was a __________ PowerPC.

Oehler: PowerPC, that's correct.

Garner: So how many people-- so your group basically grew over time. It started pretty small on the 801 project; and then with the RS/6000, it grew; and then with PowerPC, it was even larger?

Oehler: That's correct, because we had more people helping us. <laughter>

Garner: Yeah.

Oehler: So to speak.

Garner: Okay, okay. Do you have any fond memories of the 801? Like something it did not do correctly that kept you up all night? Or you know, the hardware that didn't-- or was it a smooth sailing?

Oehler: I'd never say it was smooth sailing. It sounded like a struggle, right? It was-- yeah, sure. Just like any of these machines.

Garner: Okay.

Oehler: I don't have any clear vision of what happened. Sometimes I just don't remember some of the detail. But the thing about the 6000 going to PC was that was another revolution in the architecture in things that we had another opportunity to fix before we got to the full RIOS, the RS/6000 machine.

Garner: Okay, okay. All right, if you had no other thoughts, maybe Vicky, if I could ask you for some of your cherished memories, if you're there, from the 801 project?

Oehler: Before she starts, just let me mention that this came out once in the beginning, but the thing that-- and Frank said it, too, the thing that was most outstanding about the project was that at the end of the research aisle, 23 and 24, I believe it was...

Carrubba: Yes. Mm hm.

Oehler: We took over a part of the aisle and made bullpens out of it. And we had the compiler guys working in one bullpen, and the engineers working in another bullpen. And that was a tremendous environment. I'd never been in an environment like that before, where everybody was sort of, "Oh, let me fix that! Or let me go do this!" And everyone was-- they were not waiting. They were trying to implement and get as much done as they could. So that really speeded up the cycle, getting the machine right.

Garner: Wow.

V. Markstein: We were basically ahead of our time, because when you go to Facebook now and Google, that's exactly how they work. They all work on one table. Everybody communicates. So you knew right away where the problems were. And but we were 20 years ahead of our time!

P. Markstein: Yeah, actually in the bullpen, at least in the compiler bullpen, we had something like eight or ten workstations. And people would sit at their workstation and if you had a question about anything, something looked strange, you would just say it. You wouldn't even turn around. You'd just say what was on your mind, and somebody in the room would pick it up and say, "Oh, yeah! You need to look here or there." Or, "Oh, I screwed this up. I'll get it fixed in a minute. And then--"

<overlapping conversation>

Freitas: I was in the hardware line across the hall, and you could tell when there was a problem, because the volume would go up! <laughter> Bodies would come flying out of the room!

Carrubba: Vibrations!

V. Markstein: It wasn't always so nice. There were times when somebody would scream out, "Who's the idiot who did this?!" <laughs> Everybody would go under the table.

P. Markstein: We would have it fixed and then just go off and you'd go on from there.

Carrubba: And John Cocke would bounce back and forth between the lab where the machine was being built, and the software area that was developing the software that would be running on it. And it would be injecting a little of this and a little of that, into both groups. And I always found it really interesting, which because sometimes he would come in and we would be debugging the machine, and he would start talking about a subject completely different than what we were doing. And we had to try to keep concentrating on what we were doing, and he was telling us about something that the people down in the inkjet group was doing, and how silly it was. And you know, "If they only looked at it this other way, that would have been much better!" <laughter>

Garner: And you're thinking, "John, let us get the job done here!" <laughs> So was this--

Carrubba: And what's really interesting, because he loved the cooling system that we invented, that I just explained to you. And he would smoke continuously. John would really smoke continuously.

Garner: That's when he learned.

Carrubba: So I took him up to the machine, he would take a drag on his cigarette, and blow the smoke down underneath the grating the logic boards moved around on, and watch it come up through each of the board and out the top. <laughter>

Garner: That's a great story. So who was it that came up with this sort of creative way of working in bullpens?

Fairbairn: Was this something that just sort of happened, or did somebody say, "Hey, we need to do it differently?"

Carrubba: I think we were-- I know how I felt. I think we were both-- both sides were frustrated at the way systems had been designed in the past. And we really felt that there was a better way. And one of the better ways was to be able to sit together, understand what we were doing on both sides, to understand what trade-offs that needed to be made, and to be able to decide whether they had to be hardware assisted, or if they had to be just implemented in software, to understand if there was a problem, and how to solve that problem. A lot of the hardware problems were actually solved by the software people.

Garner: Yeah.

Carrubba: Because they were able to create something that would agitate the machine to the point where it would be easy for us to find the problem. So working hand-in-hand, rather than designing a machine, picking it up and delivering it to the software group and saying to them, "Now program this thing." We did it as a single unit. And it really worked well. There was incredible camaraderie.

Garner: Yep. So next, I kind of wanted to go to Rich, because he hasn't had much of a chance to talk. Rich, do you want to talk about your involvement in the 801 project? And maybe--

Freitas: Well, I'm sort of-- go ahead.

Garner: And the story about the Demo Day, I think you told me once.

Freitas: Okay, Frank's gonna like that. Well, I joined the project a little later, end of '76, and the architecture was basically-- first pass at any rate was done for both the processor and the I/O. And so what Frank asked me to do was work at the I/O. I had had experience at Berkeley in sort of running the design of the computer systems research project machine, and I did both the-- I led both the memory and I/O system designs and implementations. And so I was sort of familiar with that, so he asked me to do that. Mostly being the I/O, starting with the I/O. He gave me the architecture documents. And if you look at from the hardware point of view on the prototype the I/O device is broke up into sort of two kinds of devices. We call them adapters and I guess devices. The adapters with boards that were in with the processor and the other devices were out in the room or somewhere else. They might have a controller associated with them. So as far as adapters go, we did a bus serializing adapter, and a DMA adapter, an interrupt adapter, and a test adapter. __________ all of those. And those were at a high level, described in the architecture document, and I kept track of those documents and those devices from an architecture point of view for the I/O devices which were external, which we had a disk controller, a schedule, we had a disk controller, a PDA controller, which was to talk to the 370, and there was supposed to be a terminal control unit, which never got implemented. And in order to communicate with those from the system proper, the core part to the devices were serial links. So all the I/O communications was done as a serial connection in the devices that were in the room with the prototype proper, they were done over coaxial cable. And out the back of each card in the midline was a little coaxial cables, and those would go out with the devices, you'd convert it back into a parallel interface, and then that would be to the controller, which would then present it to the device. For the connection to the 370 downstairs, we had to do a 500-meter link, 600-meter link, and that was done optically. And we used lasers that were designed in the optical group a couple aisles down from us. Leon Cumberford designed and fabricated the lasers for us, and then refabricated and refabricated them, because we'd blow them out a fairly good clip. And Dennis Rogers designed the receiver end of it, which was a pin diode and then an amplifier built in MECL, since everything else was in MECL. And then that sort of provided our communications. We had two ends of that. And then we strung 500 meters of this thick optical cable through the building, which was an exciting Saturday afternoon, let me tell you, watching those guys lube them up and pull them through all these boxes we had put in. And it survived having these 300-pound gorillas come in and do the yanking on the cables. But that gave-- that's the basic environment we had for the prototype. Piccolo disk and disk controller that we did, Peter Karnazes designed the disk controller, and then we had the PDA controller, which I did, I believe. And those are sort of the I/O devices. And at a sort of high level, that's the kind, that what we had as I/O for the software to work with. In order to get to that point, to have devices to make, we had in essence, two hurdles. One was how to do design with a stone ax.

Carrubba: With a stone ax?

Freitas: What we did to do logic design was we drew on ALD sheets in pencil.

Carrubba: Okay.

Freitas: And they were shipped to Kingston, converted into

Carrubba: Punch cards?

Freitas: Punch cards, and then printed on the standard IBM ALDs and shipped back to us. That cycle took three weeks.

Garner: Ouch.

Freitas: And if you had to do a correction, the way you did it was you wrote in red on the ALD, the lines and boxes to take out of the design. And if you were adding something, you put it in green and then you shipped it back to Kingston, I believe?

Carrubba: Yeah, because--

Freitas: And they'd reprint-- they'd turn into punch cards, list it, and then ship it back to us.

Garner: Frank would tell you that was the same process in the late 1950s.

Carrubba: Yeah.

Freitas: If you-- well.

Garner: What did ALD stand for?

Carrubba: Automatic Logic--

Freitas: Automatic Logic Drawing.

Carrubba: -- Diagram, yeah. They were basically software written for the 700 series in the mid-1950s.

Garner: So Frank, no wonder it took a while to get the schematics done.

Freitas: So we used--

Carrubba: Yep. All the drawings were hand-drawn, and then run through this process that Rich described. And one of the main reasons, in addition to what Rich mentioned was the fact that we had to have the boards manufactured. You know? And we had to do our own chip placement on each of the board, and so on and so forth. So we had to have a way of sort of automating things so that we could send information to the board manufacturer.

Freitas: Yeah.

Garner: So Frank, there was also no static timing analysis tool available for use. Is that correct? No static time--

Freitas: No. We had no simulators, we had no analyzers.

Carrubba: There were no simulators other than what Rich talked about.

Freitas: Rich Oehler.

Carrubba: Rich Oehler. And you know, that really was a terrific piece of work that they did, because that helped us a lot. In fact, we made one change to the machine that was a result of what we saw when running the simulator. And that was to put a three-instruction fetch-ahead capability on the program cache.

Garner: Hm. Interesting. So Rich, going back-- Rich Freitas-- Rich, did you want to finish up any more thoughts on the I/O subsystem, or I think you actually had a story once about a the Demo Day that was kind of dicey.

Freitas: Yeah, I was going to finish with the drawings. We recognized that turnaround was a problem, so I started taking apart-- also we could ship a tape. We started taking apart the tape, look at the images of the cards, and see if we could shortcut it. And Peter and I looked at that, and ran around with it for a while. And Horst Henn, who was a German visitor, came in one day and he said, "I can do what you're doing in APL. I'll write you a program." So he wrote us a program that would put up on the 3277 screen the picture of the boxes we normally used, and we could just type in all this stuff we'd been drawing, and then relist it. So we cut the time, this three-week time that was there when I joined the project, down to less than a day. And so that whole cycle, that was three or four people working on it, recognizing a problem and then finding a way to deal with it, short-circuited that process. And it gave us the ability to build, as Frank was talking about, wire list we could ship to a vendor. So I eventually took those wire list to Santa Monica, and had the boards built by an automatic process.

Garner: In Santa--

Freitas: That thing did things like sorted wire lengths. We sorted wire lengths, so the wire lists would go, and we'd put long wires and short wires we'd put on the right way, and it would tell us whether there was something that was so long we had to put in coax.

Garner: Right. The wires were stitch-welded, I think?

Freitas: No, they were wire-wrapped.

Carrubba: Wire-wrapped.

Garner: Wire-wrapped, okay.

Freitas: And there was issues with wire-wrap on little pieces breaking off, which gets to the story you were mentioning.

Carrubba: And because of the on-board power requirements, we had to have special ground planes. Paul Stuckert did a fairly elegant simulation and found out that the ground plane was bouncing all over the darn place, and we couldn't stabilize it, so we needed two ground planes.

Garner: Okay. Rich, did you have that story you wanted to say?

Freitas: Yeah, one of the issues we had in debugging the machine down in the trenches was that since it was wire-wrapped, when you put wires on or took them off in making changes, you'd knock off a little piece of copper on the end of the wire. And god knows where it would go. One of the places it would go was into the wire one you already had, since it was about a half-inch deep or more, three-quarters of an inch deep on many of the boards. The machine would run, and would run Mersenne primes. We had a MIT meter, we called it, which was a counter put on the construction complete line going out of the CPU.

Carrubba: <inaudible>

Freitas: And that would run at 14 MEPS. So when things were going good, you could put this machine up and give demos and you could see we were getting 14 MEPS, and the 3330 downstairs was running around four? I think it was three or four times-- it was running a little bit faster. So they brought in-- the machine was running to some extent. And the CEO and the President, who were-- I don't remember the order, Akers and Opel-- Akers must have been the CEO--

Carrubba: Akers was uh...

Freitas: And Opel was the President.

Carrubba: Yeah.

Freitas: And they came to research for a demo, and they were in Ralph Gomery's office, who was the director of research. And I had the secretary down there call me when they left. Well, I could turn the machine on, and then get it warmed up. Because when it was cold, it didn't work.

Garner: Uh oh.

Freitas: If it warmed up, it would run. And so it would be running when they showed up. Its other characteristic was that once it got hot, it would stop working again. So there was an interval of time of about 20 minutes when the machine first came up that it would work.

Garner: Oh, no!

Freitas: And so we'd find it--

Carrubba: Was I gone at that time?

Freitas: I don't know if we told you. <laughter> He just told us, "Demo." I said, "Okay, boss!" We got up-- and it ran-- it did everything correctly, it was just-- and then they-- you and George started talking to them-- and I guess Joel was still there-- started talking to the executives. And they didn't notice the fact that the machine failed. And it wasn't going up and down as it cycled through the numbers, it stuck at 14, and that was it, because it had gone off into the wilderness, but it went-- so that was the demo. Two or three weeks later, we found the damn little piece of copper wire that was in there that was sitting between a ground pin and one of the pins on the gate, and as the temperature changed, we surmised that it was shorting and not shorting.

Carrubba: Oh, I remember now.

Freitas: And when we pulled it out, then everything worked fine from then on. But when we gave the demo, we had a 20-minute window.

Garner: So after you found that one problem, the machine was pretty reliable on the whole?

Freitas: Well, the CPU was stabilized at that point. We'd screwed around with the clock and could see some things were-- and they rechanged the-- you know, a lot of--

<overlapping conversation>

Carrubba: It was pretty stable. We were able to run all the fault locators and exercise programs on it. And it ran pretty well at that time.

Fairbairn: Let me just interrupt. I don't know if the people in the CDU can hear me? Do we have a hard cut-off at two o'clock? Or what's the?

Crew: Yes, we do.

Garner: Okay. So maybe the--

Fairbairn: So I'd like to-- I mean, unfortunately, we could easily go on for another couple of hours. I'd like to understand a couple of things. First next few minutes, how was this project viewed within IBM in the '74/'78 timeframe? And sort of what was the reaction to it? And when-- you know, you built a machine. How many machines did you build, and sort of give me a brief rundown of sort of how it was viewed, and then how the project wound down? And then secondly, sort of briefly, what were the follow-on things? We don't want to get into the details, but sort of what were the spinoffs, if you will, or follow-on projects that used this technology?

V. Markstein: Well, can I interject? As a research at IBM, there is no guarantee that they'll make a product out of it. As a matter of fact, they want to make sure you don't think they're going to make a product out of it. So there was no intention of making a product out of it, except as we kept going along, it became pretty clear that there was a need for a scientific computer that could go fast enough. But that's when the competition within IBM, other people who had projects making such computers, decided that our project was no good. And then the competition started. And but before that, it's a research project. It's pretty hard these days to realize IBM was that wealthy that they could have a hundred, two hundred, three hundred such projects, and then just say, "We're not going to do anything with it. We just want you to go and play, and learn some things, and that's it." So they didn't care.

Freitas: What was the project that Cullum was running with liquid helium, or--

Carrubba: Well, you know...

Freitas: Simple code one? Was pat-- running at the same time. A hundred people.

Garner: if I could ask-- sticking-- we only have five/six minutes left, on the theme of Doug's question, what came next? Was it ROMP? Was it your co-processor project, Rich, was after ROMP? So how did you segue into ROMP, and was that a disappointment for you guys, because it was so slow, no cache, type-- Office Products Division. You know? What was that all about?

P. Markstein: Well, it was a way to get started. Yeah, it was slow because by then they were sort of implementing Version 1.5 of the 801, and we were starting to work on a M--

V. Markstein: America.

P. Markstein: The RS/6000, which we named America, because John said nobody would dare kill a project named America. <laughter>

Oehler: And the other part of that story, Peter, it was also named for the--

V. Markstein: Race.

Oehler: The trip around England, and on a ship-- it was a yachting race, and--

V. Markstein: A race, right.

Oehler: And eventually they came along and the America ship was first.

V. Markstein: It won.

Oehler: And the Queen said, "Where's second?"

V. Markstein: There is no second.

Oehler: And he said, "Ma'am, there is no second." They were that far ahead. So.

Garner: So did you guys feel good about the ROMP? 'Cause it seemed like it was a step down in performance.

Oehler: At that same time, though, we were doing-- well, Peter said it. I was still in Austin at that time, and we were working on trying to get to a power-- excuse me, I shouldn't say that word-- get to a place where the-- damn, sorry. I'm losing the thread of my conversation now. But I was there, and we were working on Version 2 of the architecture. And very quickly people went off and were working on-- I remember when Filesta came and wanted to see all the pieces of the puzzle, and basically we took them through it and he went away and came back several weeks later, and said, "This is what I want to do. Basically lay it around, and the caches that were on that, and the I/O, and he was going to put it inside of a typewriter." And I was happy with that. He was keeping the architecture alive. In the meantime, we were spinning off ideas about architecture in Poughkeepsie and in Endicott. We actually had a machine up and running in Endicott that was a full-- talk about turning things on their head, it was a full 370 running on top of a RSC machine.

Garner: So can we just describe how did the specific first project that each of you made such a major contribution to, the 801, how did that ramp down, and what year did it ramp down, and was there one machine built, or multiple? How did that all happen?

V. Markstein: Went to __________.

P. Markstein: It didn't feel like it ramped down. It felt, you know, they were offspring that continued to advance the state of the art as we had it. So from the various 801s that we built at the lab. And I think there we only built one of each. Some divisions actually used that architecture to make controllers, or to make micro-engines for the 370s of all things. And then, of course, there was ultimately the RT, and at the same time, the development of the RS/6000. So for those of us who were there from the beginning, it never felt like it stopped. It felt like it was going on.

V. Markstein: And then it went to Austin where they built the machine. And we went to Austin with the machine. So it never wound up for us. We were just constantly there.

Freitas: There was Ole and I-32.

<overlapping conversation>

P. Markstein: That was part of Rios.

V. Markstein: Rios.

P. Markstein: Office 1.

V. Markstein: There was Rios.

P. Markstein: Over at Amazon actually. It was the wide.

Garner: So this architecture technology spread into many different divisions, many different aspects, you all, several of you followed one path of that, but in fact, it propagated elsewhere within the corporation, is that correct?

Carrubba: That's right.

P. Markstein: Yeah, in fact, all the really fast machines were based on those architectural principles. I think Big Blue basically had an 801 derivative in it, and probably Watson does, too. I can't say, because I haven't been at IBM.

Garner: No, that's true, yeah.

P. Markstein: Hm?

Garner: That's true.

P. Markstein: Yeah.

Garner: No, I think it's been very successfully deployed. I think some people who in Silicon Valley thought that maybe, you know, if there had been a, you know, the RS/6000 to come out sooner, IBM could have gotten a leap over some of the other firms here. But from your perspective, it was like a fine line. The RS/6000 came out at a reasonable timeframe?

P. Markstein: No, we though they should have done it way sooner.

V. Markstein: Way sooner.

Garner: I thought you would have, yeah. <laughs>

Fairbairn: But in fact, your project had a major influence on future architectures of all of IBM's equipment as well as, obviously, those outside of IBM.

V. Markstein: IBM, yes.

Carrubba: Yeah, outside as well. Because many, many people followed in our footsteps. And many of them followed rumors more than, you know, actual hard copy evidence that this thing actually existed. <laughter>

Fairbairn: All right, well, unfortunately, I think we have to wrap up due to constraints of television time. But it's certainly been very educational for me, and we'll have some-- just as with the 801 program, we'll have some follow-on oral history discussions to carry it further. So thank you each for your participation and appreciate it very much.

Garner: Thank you very much, guys!

P. Markstein: Thank you for putting this on.

Carrubba: You're welcome!

END OF INTERVIEW

