Oral History of George Michael

Interviewed by:
Eugene Miya

Edited by:
Dag Spicer

Recorded:
January 17, 2006
March 14, 2006
May 9, 2006
July 12, 2006
July 25, 2006
September 15, 2006

Mountain View, California

CHM Reference number: X3418.2006

Eugene Miya: Okay, we'll start off first with Anel's sort of standard pat questions and then from there we'll elaborate everything. So, CHM wants me to ask you, early years, project and than overall kinds of things. You went to a lot of projects obviously, so let's start off with your early years to keep things simple. Where did you grow up?

George Michael: Buffalo, New York.

Miya: Do you want to say some more about Buffalo?

Michael: Well, when we grew up there it was a blue collar town and it was – the biggest virtue from my point of view was it was right across the river from Canada, which I loved. We used to go up there all the time, swim, hike and bicycle and I liked it.

Miya: Okay, just to get things out of the way, what did your mother and father do?

Michael: Well my father was a train master and then station master for New York Central in Buffalo. My mother was a house maker or a housekeeper, or whatever you want to call it; a mother who stayed at home, unusual today.

Miya: How many siblings do you have?

Michael: One.

Miya: A boy or -

Michael: A sister.

Miya: A sister, younger or older?

Michael: Younger.

Miya: And her name was?

Michael: Dorothy.

Miya: Is she still alive?

Michael: Yes.

Miya: How many years younger?

Michael: Two.

Miya: Okay, and if we wanted to get in touch with her, where does she live?

Michael: In Hamburg, New York.

Miya: Okay, and -

Michael: Which is a suburb of Buffalo.

Miya: Okay, what's her last name now?

Michael: Cook.

Miya: Dorothy Cook, okay. What are your – were your youngest thoughts about what you wanted to do when you grew up? Provide some inspiration for young people here, ah see we have the George Michael humor here.

Michael: I don't think I had any thought except that I remember reading about the transmutation of one element into another by driving particles into the nucleus and I thought that would be a good thing, take some lead and make gold, but I never went any further then thinking about it. Other than that, I think I wanted to just be a good swimmer, a football player, runner, outdoors person in general.

Miya: How old were you when you had that series of thoughts about transmutation?

Michael: I don't know 12, 11 somewhere around there.

Miya: Sixth grade? Elementary school?

Michael: It was probably sixth or seventh grade, yeah.

Miya: Bear with me on the next question; I know how you feel about this.

Michael: All right.

Miya: Who were your idols when growing up?

Michael: I don't remember any.

Miya: Okay.

Michael: Well that's not strictly true. Actually, when I was – I took a job as a laborer in a feed mill and the guy who was my boss was kind of an idol in the sense he could pick up in each hand a 100 pound bag of feed and walk around with them and yelling at people to get busy and get their work done and stuff like that. He was quite a guy. He also taught me an enormous amount about how to survive strength

issues without wrecking your body. The word that he used was don't commit your weight and that's a good piece of advice as it turns out.

Miya: Do you remember what your boss's name was?

Michael: I don't remember his name. I know we called him Happy and had the interesting experience after the war of going with one of my uncles up to Port Colborne, Canada and meeting Happy there when we had supper in a little bar, and it was very nice. He remembered me, I remembered him, nice guy.

Miya: So you stayed in touch with him briefly after the war?

Michael: Nope.

Miya: That was it? Just that one day?

Michael: That was just this one time I saw him there and – the business of keeping in contact just wasn't very common for people.

Miya: Okay. This is going to be a long one for you. When and how were you first exposed to computers?

Michael: That would have to be in 1953 after I graduated from college and went to work at the – what's now called the Lawrence Livermore National Laboratory (LLNL). At the time it was called the University of California Radiation Laboratory in Livermore and I arrived on the 16th of April in 1953 and the Univac 1 arrived one week later. It was clearly the first experience with a computer that we had. It didn't dawn on me what was important about computers for perhaps a half a year. They were just adding machines if you will and we had the interesting experience that the calculations that we were going to do were carried out to 12 decimal digits and we had a crew of desk calculating people on whose hand calculators Monroe's and Marchants and Fridens, and every number that the Univac 1 produced was reproduced by a numerical calculation carried out by hand. When there was some doubt, people tended to trust the hand calculation more then the computer calculation and I think subsequently that's certainly been the wrong thing. Univac did make mistakes but it had this beautiful virtue of correcting them, and if it couldn't correct it, it would stop. So the Univac was a decimal machine and it was the first machine and to me, and I think to all the other people who were the using thing, decimal was natural. That concept of binary hadn't occurred to anybody then, and again, that's not quite true. One of the guys used to come regularly to the laboratory for consulting work with John von Neumann-

Miya: We'll get to him.

Michael: - the mathematician and he was pushing binary calculations. But other than that everybody else felt that if you entered a number like .1 it should come out as .1 and not as a repeating decimal as it was with binary notation. But we overcame that, we've outgrown decimal.

Miya: Okay.

Michael: Well you know -

Miya: Yeah; no, we'll get that at a further degree.

Michael: All right.

Miya: We talked about the value of decimal machines are for you and I in the past. We should probably say that we've known – you and I have known each other for almost 25 years I guess.

Michael: Has it been that long?

Miya: I hate to say it, yeah, I think so; something of that order. Do you want to say anything in terms of the when and how – do you want to say anything a little bit about the town of Livermore, by the way? This is where you received the Univac.

Michael: Well yeah.

Miya: How big was the laboratory at the time?

Michael: Livermore's – actual location was next to a naval air field that was chosen as one of the few places in the Bay Area that had few foggy nights, so people could land there all the time. Other than that, I felt that Livermore was a mean, pinched, little town and I've never seen much to change my mind about that. It's now a big mean, pinched town. At the time when we got there I think there were only three or four paved streets in Livermore and that was a virtue and I thought it was very charming to be able to go to a real general store and buy groceries, hardware, kerosene lamp chimneys, curing salts for meat, all of it there in one store. That store went out of business unfortunately very soon after we got there. We were not particularly welcomed when we got to Livermore. The people who lived there, I think rightly, were worried about all these stupid others who were coming to the rad lab, or they called it the atomic; they're a strange breed of people and all they could see was that the existence of the laboratory there would be a big source of causing them to have larger taxes. I remember the rotten gun club passed a resolution that said if you weren't a native Livermoron, you couldn't vote on financial issues in the rotten gun club. This was essentially to protect their little treasury which is a whole $8,000. But things have moved on from then and the rotten gun club is still there, Livermore is much bigger. It went from 3,200 people to close to 70,000, I do not consider that an improvement.

Miya: So the lab was there in part because there was a naval air station there with a field that could be -

Michael: Yeah, the background on that is that it's almost folklore. Edward Teller was unhappy with the aggressiveness of the people at Los Alamos on the issue of trying to build a hydrogen bomb, which they called then, the Super, and he agitated as only he could, at most levels of government in Washington. Finally, they gave him, if you will put it that way, the abandoned naval air station outside of Livermore. At the time, the only thing that was there, other then the abandoned station, was a thing called the materials testing accelerator that was run by California Research and Development, which was a part of Standard Oil of California. The accelerator was used to test the behavior of various incendiary materials when they're exposed to various incendiary radiation pellets and the laboratory started, I think, either July or September of 1952.

Miya: I think it was September.

Michael: September, probably a good time and most of the staff that came out to the laboratory came from Berkeley and there were some really good people among them. The laboratory had grown from the initial 200 now to about 10,000, and I'm not sure that's good either.

Miya: The physical location though was, in any way, the runway – you said it was selected because it was-

Michael: The runway was -

Miya: - fog-free nights?

Michael: - just accidental.

Miya: _____

Michael: We didn't care about the runways, what we cared about – what they cared about I supposed is the barracks and the BOQ for the naval officers that were there and things like that, the hangars; they built a [?] accelerator inside of one of the hangars and it just was a facility into which you could expand a lot, and it was close to Berkeley which was a virtue. So they poured a lot of money into that place I must say.

Miya: Okay, we'll revisit both Teller and von Neumann because they -

Michael: Yeah.

Miya: - they want to know that kind of thing. So where did you go to college and why there? Tell me about the University of San Francisco.

Michael: Well I went to the University of San Francisco -

Miya: After World War II?

Michael: - after World War II, yeah. I began what you might call an abortive university experience at the University of Niagara in Niagara Falls, New York. But I came out to the University of San Francisco, not because it was the University of San Francisco, but because it was close to the person, the woman that I finally married and while at the University of Niagara we used the book in one of the philosophy courses that had been produced as a translation from the Italian by a professor at USF and I thought that was pretty nice so I went to USF because it was close to Heidi.

Miya: Right, we'll get to Heidi as well too. You remember who the author of that book was?

Michael: Well, yeah, the translator was – it just went -

Miya: That's okay.

Michael: Fearon; Arthur Fearon, F-E-A-R-O-N, and don't ask me to tell you how to spell the Italian author.

Miya: Wasn't it Ferring?

Michael: No, it was Vivello but I don't know how to spell it.

Miya: Varvello; that's okay. So Heidi, your first and only true love -

Michael: Yes.

Miya: - was the big influence for the selection for your university?

Michael: Not directly, it was my choice to want to be near Heidi. So -

Miya: Yeah.

Michael: - it seemed like a good thing to go to USF. She had graduated in 1948 from Dominican College in San Rafael and she had a Chemistry and a Mathematics major and at the time she was going to go to work at The Cutter Laboratories over on the East Bay side of the bay and we committed marriage in 1948 as I recall, but I don't remember what month, and I started school in 1948 at USF and I didn't get out of there until 1952. It was a traumatic experience to have to work and go to school and try to do well in your studies and stuff like that, but I'm glad it's over with, we got through it okay.

Miya: I think actually I'll back up here a little bit, knowing you, and that before college I should probably ask, since I know you so well, say a little bit about your brief time during World War II, your service.

Michael: Well it was quite brief; it was, I suppose you would say, macho or something like that or enthusiastic enlistment and what was called the V5A Program, and persons who survived that became Naval Aviators.

Miya: So you went into the U.S. Navy?

Michael: Yeah, U.S. Navy as a cadet essentially. I only lasted two years and than the war ended and the Navy sent us all home essentially.

Miya: Did you learn to fly?

Michael: I was in what was called primary training and that was done at Memphis and they then offered – you could be discharged and they gave you some money and they said that you guys are not going to pass the flight test, so why don't you take it, so I did and by that time I had met Heidi and so I wanted to go back there. So then I got discharged I guess in 1946 and I went home – no I didn't, I was discharged

out here for some reason and so I went up to Eureka where Heidi lived and I spoke to her father, saying that I would like to commit marriage, and he said he wants her to finish school, and I said that sounds like a good idea, and I went home. I thumbed my way across the United States; it took only two rides, and made it back to Buffalo. Then Heidi came to meet my parents the following year, 1947, she met the guys I ran around with and we agreed that we wanted to get married so that happened the next year and it was a wonderful experience. I loved her very much, I miss her yet.

Miya: Yes, I know.

Michael: Huh?

Miya: I know.

Michael: Yeah.

Miya: Did Heidi provide an inspiration for you for selecting Physics as an area to study -

Michael: No.

Miya: - from your time at U.C. San Fran – or I should say U. San Francisco.

Michael: The reason I chose physics is because it looked to me like if you knew the things there, you couldn't be argued with about how nature worked and things like that. In the summer of the previous year I had worked as an electronics technician for Westinghouse out at the Buffalo airport and that was intriguing, but physics seemed to be the most basic science and that's the reason I chose it. Largely, because when we would have arguments you could say that you'd studied the stuff and this was what you had learned and people would shut up. The only thing I think I had wrong, interestingly, was the question of how is electricity conducted at high frequencies in a cable? I think I had that wrong but I learned differently as I went through physics and it was as we – this one guy from England said, do you want to study physics? The first thing you should do is study mathematics and than the second thing you should do is study mathematics, and than the third thing you should is repeat steps one and two. So that's why I ended up majoring in mathematics as well and I think – I'm not a pure mathematician, I'm not pure anything. I am applied mathematics and I must say it's been a fascinating tour and does make physics more sensible and as some people have said, it is amazing how well mathematics describes nature, I think Hamming said that.

Miya: Well just to interject here, one of my favorite little things that I did with another friend of ours, so most scientists refer to mathematics as the queen of the sciences?

Michael: Yeah.

Miya: So who's the king of the sciences?

Michael: Physics.

Miya: That's what Cliff would say; as opposed to other people I know who would say, like England, science is – or mathematics is a queen without a king kind of thing. But every physicist I know says that, so it's the king of sciences, I just wanted to make sure. So I think we've answered the question of what did you study and why. Were there any other topics that you studied besides physics and mathematics you want to put in a word for or anything like that?

Michael: Not really. I spent a lot of time studying various forms of philosophy because it seemed like you could win arguments by being very methodical about how you proceeded. But I've decided that the best thing I could say about philosophy was that if you start with a false assumption you can prove anything. So I prefer physics and mathematics which are testable and correctable.

Miya: I'll have to get a drink. Do you record him by the way as well his drinking, but as well as he could record mine?

Michael: It's inevitable.

Miya: Okay, the next question Anel wants me to ask you is when and what was the first program you ever wrote?

Michael: Well part of that borders on classified so I can't name the program, it was part of nuclear design; whatever you want to infer from that and it ran on the order of 40 hours on the Univac for each problem that it ran.

Miya: It did work the first time?

Michael: Oh no, but one of the things – remember I mentioned earlier that we had taken a whole cycle of the Univac calculation and reproduced it by hand. That helped us find enormous numbers of errors. Nobody was really concerned about the speed at that point because it was dramatically better than what a human could do, but I remember that one of the programmers that was helping me we had to rescale numbers because of the tremendous range that was required and instead of shifting, which was the division by ten, she would divide by ten, and that's very, very slow. So a lot of the time spent on this calculation I'm sure was due to the fact that it was not efficiently written. It produced good results, it produced voluminous results, but it was slow and from that in the process of writing that, you need some recreation now and then, so one of the guys that I worked with, Bob Price, who got to be the President of CDC and presided at its death, and we worked out a scheme for using Monte Carlo techniques to do an integral. That was very interesting. It filled in a lot of mystery that I had assumed before that. We had another program that was – stood off on the side--was one that was designed to have the Univac do its own inventory and there was such a ferocious competition for time that you really couldn't work very hard on anything that's off on the side. You had to do the main calculations and there were numbers of people trying to do the same thing. Univac was – it was decimal number one, number two it was checked, so that every second it took almost a tenth of a second to check all of its results and all of the contents of the memory. The memory was – well one way of saying it is a 1,000 words, another way to say is 12,000 bytes, these were 7-bit bytes and that in contrast of – for instance today I had sitting on my desk a Macintosh G5 I guess it's called, it has 4 Gigabytes of RAM that you could say is a half a billion words of memory and it has a 160 Gigabytes of disc and that's more then the lab dreamt of until most recently. So things have changed an awful but the need to write a good program has been – gone by the wayside almost. The people today do not produce nice tight code generally.

Miya: You mean an efficient program?

Michael: Huh?

Miya: You mean an efficient program?

Michael: It doesn't do you much good to make a tight physics program if the substrate of your program is full of editors and assemblers and stuff like that, that are sloppily written. It is almost through all of my career that business of chasing down one or two words to save those and shave off a few micro seconds here and there was a pretty standard operation for everybody.

Miya: Well I think some of those details we'll probably into further on, especially for all the other machines. I'm not boring you am I?

Michael: No, I'm amused.

Miya: I know you're amused and I know how you think about these particular questions here. How else – for that very first program, this is just the first program now, she has this general how did you do it? So one of the things I should probably ask is what programming language did you use for the first one, can you remember?

Michael: At the beginning there was no such thing as a programming language. It was quite natural to develop some sort of a shorthand which was the equivalent of a programming language, but to begin with we just had instructions and registers for operands and the registers had a number between zero and 999 as did the instructions. But it was natural to say, as you were writing something, to give a name to these various operands and refer to that name from a little table that you had written down and put it in the number. So we performed a function of an assembler, if you will, by hand. There were no programming aides when we started.

Miya: No operating system?

Michael: No operating system, it was just a -

Miya: A raw, bare bones machine?

Michael: It was a raw machine and I don't know what to say, it was just so natural to use the machine the way it was. When the assemblers and debuggers and compilers, and all that stuff came on, they moved the user one or more steps away from the actual program I was going to run. Given that, they lost a sort of a symbiotic relationship with that program. When my program was running on a machine I could stand next to the Univac and just watch the lights and tell what it was doing. Of course, that's partially because it was slow, but it was also part because I knew what was going on, and in some places we began, as our sophistication was improving, we threw in little display commands essentially to freeze the displays that were shown on the console. Just to tell you that they were at this part or were at that part and so forth, because you knew that register wasn't going to be used very much and so you set it up and there watched and when it changed, you knew he had left that section to the code program and gone

somewhere else. We had an audio amplifier just coupled near the high speed bus and you could listen to the machine and you could tell what it was doing also, and some of the people, whoever I quote, clever enough or interested enough, or devilish enough, whatever you want to make the machine sing or play. I have some tapes somewhere of the Univac singing Merry Christmas and its intelligible… I never really much cared about that kind of stuff but it's impressive that they could do it.

Miya: Was that something that Univac did or -

Michael: Univac, yeah.

Miya: Okay.

Michael: We had a really gifted young man when we had – much later in the CDC 3600, and he fixed – he wrote a program, if you want to call a program, such that all parts of the machine were used to play a band version of the Stars and Stripes Forever, and he had the tapes rattling in the vacuum columns, and different frequencies running over the high speed bus and I don't know what other things that he used in general, but when they found out about that he was doing this they were going – they were interested in firing him because he was wasting valuable machine time, but saner or more reasonable heads prevailed and actually the program was a big hit when they had a family day where uncleared people would come into the lab and they see this machine playing the Stars and Stripes Forever, it was quite impressive.

Miya: Okay, I want to at this point and time stick to the Univac, but we'll get back to the CDC 3600. Well, I can imagine a few of the things that need to be asked in terms of how you programmed the Univac 1 – for your first program, we want to concentrate just on your first program. Okay, so when you wrote your first program did you write in coding sheets, and were you the person who punched this up -

Michael: Nope.

Miya: - on punch cards?

Michael: I wrote on coding sheets, it was almost impossible to not do that, and then we passed these coding sheets to Cecelia Larson… on what's called her function in that place was a Uni-typist and she would type your instructions onto a piece of metal tape, a half-inch wide metal tape that had an oxide coating on it, at 20 pulses per inch; that was an input tape, there was also an output tape if you wanted to feed that tape to a typewriter printer, it would accept it. Then we would load that tape into the machine and hit start. There was no assembly needed, it was already in machine executable form and it ran.

Miya: You mentioned lights and switches. Did you have to also manipulate the switches as well to, like on the PDP-8's and stuff like that later?

Michael: Even more so.

Miya: More so; tell me about – tell the people about the switches that you got to – console switches.

Michael: I can't do that too well Gene, it's a – the Univac was a fairly solvent[?] machine, but the tape drives that it had on it were sort of Rube Goldberg kind of things and they gave a lot of trouble. They were literally symphonies of sealing wax and string, and springs, and stuff like that and every now and then there would be a mistake. It was either in the memory or on one of these tapes and the operator at that point would essentially, using toggle switches, back the calculation up to a last good point and start it over. So for 24 hours a day there was somebody servicing that machine, making sure that it was going to run okay, and when it didn't run, to fix it, so it did; that operation was called SCICR, but I don't recall what that meant, supervisory control or something or other. It was pretty interesting to watch and some of the operators were virtuosos at how fast they could do that. There were a whole bunch of switches that had to be toggled in the right order and then you could get the machine to work. One or two of the guys were quite good at that, their hands would fly, and you couldn't even see them they were moving so fast.

Miya: How many people did you work with in order to accomplish getting your first program? It sounds like besides Cecelia you had a whole bunch of other people as well too?

Michael: No, there was a senior physicist -

Miya: But you weren't him?

Michael: No.

Miya: Okay.

Michael: I was a junior physicist and there was a junior physicist, me; and there was a programmer who would handle some of the routine stuff, they didn't do the physics itself, and there was Cecelia, I mentioned Cecelia.

Miya: Cecelia, yeah.

Michael: That's it.

Miya: So there were just four of you working on this thing?

Michael: Well nobody worked on it except the junior continuously, the others would just drop in when they needed to have untangle something or something like that, they did it.

Miya: But do you recall for instance how many other Univac people were working just to maintain?

Michael: There were probably 20 Univac programmers, maybe more then that I don't know. The thing is you can imagine that there was a kind of a religious attitude about things. The physicists didn't trust a programmer to translate -

Miya: The word from God.

Michael: - his physics into a viable program. So many of the physicists learned programming on their own and they would write their programs and then a programmer, well they were called coders at the time, would help massage the thing into an executable program. There were always, maybe two or three engineers around the place to kick the Univac into life again when it ran into trouble.

Miya: Two or three hardware EE's?

Michael: Hardware guys, yeah.

Miya: These all came with the machine?

Michael: No, they were hired – we only had one guy, Lou Onofre, who was the world's first computer engineer at Livermore.

Miya: Lou Onofre like San Onofre?

Michael: I don't know, Lou Onofre is _____ and he had a crew of people that are probably eight or ten, that maintained the entire machine and ran it when people were trying to use it. To give you an idea of that, one night there was an electrical storm -

Miya: Hold that thought because we're running out of tape here, we're going to get ready to switch it--

END OF TAPE 1

Miya: At the end of the last tape, I asked- since you described the three-person, or the four-person hierarchy, on your first computation, I asked who your senior physicist was and you said Michael May.

Michael: Well it's more correct--. There were maybe two or three senior physicists who were doing it. Mike was one of them, Sid Fernbach was one of them.

Miya: Oh, we're going to get to Sid later, okay.

Michael: And Bob Boulivieux [ph?] was one of them- and as a matter of fact, Art Beale was one of them.

Miya: Okay, we'll check the spelling on those later.

Michael: So it wasn't that I had a single person- okay? This team developed the equations that we were going to use and when each section of the old process was being worked on, they would be involved.

Miya: Now were you the only junior programmer, or did you work with others?

Michael: I had a programmer to help me. Her name was--.

Miya: Oh excuse me, were you the only junior physicist?

Michael: There were these physicists I named, and then there was me.

Miya: Yes, but you were the junior physicist.

Michael: Yes.

Miya: Okay, so there were the senior physicists, Michael May amongst them, with Sid, who we'll get to, who was prominent in your career. You were the junior physicist.

Michael: Right. And then there was a programmer.

Miya: A programmer.

Michael: Leotta Barr, and--. That's sort of about it.

Miya: Leotta Barr, and then there was Cecilia, who did the coding.

Michael: Well Cecilia Larson was a resource for the Univac- the whole Univac, not just me. She typed input tapes for everybody who was producing them. She was a very remarkable woman who was more than just a typist. She was the soul of the Univac, in many respects. A great person.

Miya: Okay, we'll probably revisit some of this stuff but--.

Michael: Yes.

Miya: Okay, so I'll suspend the questions right now regarding your first program and go on to Anel's next ones. Who were the most important influences in your life in college?

Michael: I guess- I don't really know his first name but he was one of the math instructors, Mr. Quinlan- and one of the physics instructors, Joseph Rood.

Miya: Like in a rude awakening?

Michael: R-o-o-d.

Miya: Okay.

Michael: That's about it. There were other nice people- you know? Carl Waider, who was the chairman of the Physics Department, and I spent a lot of time with him too, for that matter.

Miya: Carl Waider?

Michael: W-a-i-d-e-r.

Miya: Okay.

Michael: And I actually spent some time thinking about those days and, you know, the best description I can cavil up is to say that I was unconscious. I went through that really important section of my life being unaware of anything.

Miya: Do you think that helped you?

Michael: No. It was more this nonsense of working, going to school, being a father and all that stuff. That's tough.

Miya: Life is tough.

Michael: Well, I was knowing a neat level of poverty that I'd never known before. Before I was- you know, I didn't have a lot of money but it didn't make any difference. You know what I mean? But when you have a wife and a child, it makes a difference.

Miya: Let's see, who was the first child- who was your first kid?

Michael: Karen.

Miya: Toot.

Michael: Yes. And the rule that says, you know, a child changes everything. That certainly was true- just a lot of grunt work.

Miya: Do you think it helped your programming at all?

Michael: No.

Miya: Or did you think about programming at the same time you were trying to raise Karen?

Michael: No. During the time when- you know, just before Karen was born and then during her first year, I guess it was, I worked as a water chemist for a company that sold boiler water treatment for- onto ships. And they would periodically draw out samples of the water, bring them to our office and I was supposed to test them to see if they're staying within the range of the chemicals that you put in there. That was interesting but I really didn't like chemistry- dirty.

Miya: How old was Karen when you started working at the lab?

Michael: Karen wasn't born when I started--. Let's see, yes, she was about one month old. Let's see. She was born in March, I think, and I went to work in April. So it was about a month.

Miya: Okay, because we didn't get your water chemistry background as well too. We're obviously not going to cover every job that you've had.

Michael: Oh.

Miya: But at least the parts that are leading up to the computation I think are important, kind of thing.

Michael: Well certainly the water chemistry did not lead up to computation.

Miya: Right. Well see the next question that she wants me to ask is, what was your early job history?

Michael: You just got it.

Miya: Right.

Michael: There was one other job, which was…

Miya: Before the--.

Michael: …pretty important to survival and that was I worked as a post office driver. And I had that job from 1950 to 1952.

Miya: That was where?

Michael: In San Francisco. And it was pretty important in the sense that it gave us enough money to survive. We had the GI Bill, at that time.

Miya: See, a lot of people are not going to know what the GI Bill is in the future. So.

Michael: Well the Congress passed a law that said anybody that served in the Second War had some credits which you could apply to an education, or whatever else you want to call it. Some of the people were using their credits to take dancing lessons and some were using their credits to open up a little store to sell whatever they wanted to sell. And most of them used their credits to go to school. It is, in my opinion, absolutely the most important investment Congress ever made. When the time came that "people were needed", there was an incredible army of trained people. And I think that the schools were helped by the GI Bill, and so were the- oh, the people who went there.

Miya: Well I think they're going to be more interested here in terms of your early job history was- essentially largely regarding computation. So when you were a junior programmer, on your first program, did you go from- for your next programs you were sort of the junior programmer and then you went up the

scale- excuse me, not a junior programmer, a junior physicist. You ultimately became a senior physicist or--?

Michael: Yes, in a sense- in the sense that the original senior physicists on the job that had guided the first program I wrote, were off on other things, in a sense. And I took the basic idea and just implemented it on other machines as they came in. In every case I would have a programmer helping me? And we added complexities to the problem.

Miya: Just one programmer, typically?

Michael: Yes. Yes, I don't think you should say "just one programmer, typically". I think it's- one programmer was more than enough? I mean, people can get together and trade ideas and see things work out right and have a good channel for communication when things don't work out right- see? The second guy that worked for me left the laboratory to become a taxi driver because he could make more money.

Miya: The times when programmers were cheaper than taxi drivers.

Michael: Apparently. Well, it probably isn't legal or anything like that but they were getting money from tips and stuff like that, and they don't have to report that- at least they didn't. So when the Univac was- oh, not finished, but when we finished with the Univac, our next machine was a [IBM] 701, and that was a symphony of errors, and it was just terrible. The 701 was a binary machine and it was parallel, as opposed to the so-called serial architecture for the Univac. So it was quite a bit faster. But that only meant it made errors much faster- really much faster.

Miya: You mean serial versus parallel, you mean parallel as in a parallel word or a parallel byte?

Michael: Yes, parallel word.

Miya: In registers- and the serial ones, they were bit serial or digit serial?

Michael: Yes. In the Univac it was bit-by-bit, character by character, number by number. And they'd go through. So if you had a 12-digit number, it had to go through the adder 12 times, and so forth. With the 701, all the bits went through to the adders at the same time. And they had the usual carry and overflow and other forms of signals that had to be taken care of. But it was faster. However, it made terrible errors because the memory was very flakey. It was I think the only machine we had experience with that used Williams tubes for memories? And they were very strange. I had the occasion one time- with the help of an IBM engineer, I ran one cycle of my calculation 15 times and never got the same answers. And he found the trouble, but that didn't change the fact that I couldn't trust the calculations? He found what he called microphonic tubes? You could tap on the tube housing and it'd shake and give you strange results. We found that if you had the memory bank oriented the wrong way so that when people came in and out- used the door to get into the room- sunlight would come into the room with them and it'd blow the memory, and you'd get a lot of memory errors from that. So we took to wrapping the machine in black cloth, as I recall, and that solved most of those kinds of light sensitive problems, but it didn't solve the microphonic-type problems, and we had a lot of trouble with that. I did not like the 701 at all, though it, in some sense, set the style for all subsequent machines. The 701 was unreliable- that's the best way to

say it- and it was faster. But I think that being right's more important than being fast, at least when you're beginning.

Miya: I was going to say, I've heard you say the opposite.

Michael: Not me.

Miya: Oh yes?

Michael: Yes.

Miya: Hum, okay.

Michael: Anyway, after the 701, we moved to the [IBM] 704 and we got away from Williams tubes and made use of the first memory supported by [magnetic] cores. Okay? And that was a dramatic improvement. So the 704 was a tube machine but the memory organ was a ferrite core. And the- just the hardware stability was much better. Then, adding to that, the software got to be much better, not the least of which was the introduction of floating point. We didn't have that on the 701, or the Univac, but we got floating point on the 704. And that made quite a bit of difference. We didn't have to worry at all about scaling, though some times were spent making sure that you didn't have to worry- and just as a good scientist would want you to do. The 704 had one other thing on it. It had a thing called the [IBM] 740 and the [IBM] 780. The 780 was a direct view display tube and the 740 was a tube, a five inch tube that was imaged by say the lenses in a camera- so you could photograph. The stuff was just lousy as heck. No one had done any real engineering to get decent photographs from a cathode ray tube.

Miya: The 740 was a scanner? You mentioned the 740 and the 780. The 740 is a scanner and the 780 was the output device- the input and output devices.

Michael: No, the 740 was a five inch tube that you photographed.

Miya: It was a photograph.

Michael: Yes. The 780 was a display tube that you could see direct view.

Miya: I should change it, I should say, the 740 was a camera.

Michael: All right, if you want to say it that way, sure.

Miya: Well we just want to make this--.

Michael: If that helps you to clarify it, that's good.

Miya: Well the thing is to put the right terminology so people kind of have--.

Michael: Well the point is that we didn't put just a camera in front of the stupid thing. We put other things there. So it wasn't just a camera. But anyway, that started me on the long adventure of developing lenses, films, processing styles and so forth so that we got very good film- exposures and images that were really useable.

Miya: Yes, we'll probably talk about Kodak and Polaroid a lot.

Michael: Well they both--. More than Kodak, it was Xerox, Kodak and it was our souping. It was the technical photography group at the lab, which was really a great group. And it was just some attention to details- okay?- that hadn't occurred initially, to anybody. But after awhile, that's what made the difference between good film recording and mediocre film recording. I remember much, much later, in the late '80's, showing a film that we had made in 196- say 2, just to pick a number, around that time, and it brought people up to their feet, saying that the film was so good. I mean, it was just very, very clear film- color film, better than, in lots or respects, than the stuff that people were routinely getting off their tubes now. Yes, it was really good stuff.

Miya: This is the Dicomed?

Michael: No, the films that we made were--. Just getting off into the real wild stuff, there was a so-called Oxberry Special Effects camera and projector and special films and filters, so that the output of the cathode ray tube was photographed on film and that output was transferred from that film to another film, in order to have color output. We had color output before anybody else did- I guess is the way to put it.

Miya: Right. Now I assume a lot of the stuff you covered actually in the very, very first of Jeannie Treacle Bay Area Computer Perspectives when you had a total audience of about three of us, at Xerox PARC- do you remember that?

Michael: Yes.

Miya: So I think you covered a whole bunch of that in that--.

Michael: I expect that's true- yes.

Miya: So that's something--. This is an annotation on the audio portion of this.

Michael: There were more than three people there.

Miya: No, there was only three- [Donald] Knuth, Peter Nurkse [ph?], you, me- because we all went to- we did the precedent of going to dinner afterwards. Remember that?

Michael: Yes.

Miya: But I was surprised that Knuth was there, actually. It's quite an honor that Knuth--.

Michael: It was great. I was pleased that Donald came though.

Miya: Yes, I thought it was neat. That was a good time, and I had my first of many meals with him, to subsequently follow. So see, there's Don Knuth- you're name dropping another name. Everybody knows George Michael.

Michael: No.

Miya: So we've got to tease out all these people. Oh yes.

Michael: Well we'd had a lot of experience with Donald Knuth, prior to that, because I asked- had invited him, and he accepted, to come over and give a distinguished lecture talk at the lab.

Miya: When was that?

Michael: I don't remember.

Miya: The 80s?

Michael: We had money that the Department of Energy- well it was then called The Atomic Energy Commission- gave us, and we invited people from all over the United States, essentially, to come and give talks? Hart Maness came, Minsky came, Knuth came, McCarthy came- there was just lots of people. It was kind of a cultural raising for the people at the lab who were off in this applications stuff.

Miya: In Computation?

Michael: The whole lab. When distinguished lecturers come, the auditorium would fill, with 400-plus people. And they'd take the overflow and put it on video and people would sit in other rooms, watch the video.

Miya: Let's see. Which McArthur in computing?

Michael: McCarthy- John.

Miya: Oh, John McCarthy.

Michael: Yes.

Miya: Oh, our friend, John.

Michael: Our friend, right. A great person.

Miya: Yes, I saw him on Friday. I was with Gio [Weiderhold]. So you guys started having these lectures when- in the sixties?

Michael: Gee, I think we've had those things probably from 1965 on. They're still having them. You know?

Miya: Yes, I know.

Michael: All kinds of people come to a lab and talk about their persuasion. I tried to get [Richard] Feynman to come to the lab but he wouldn't do it. That's all right, I understand. We had Ed Fredkin come to the lab and he upset the local physicists quite a bit. He was talking about how computers are going to be so pervasive. He used the example of they'll even be in light bulbs. And some of the local scientific staff were just sort of up in arms and saying, "This isn't physics, blah-blah-blah." How wrong they were.

Miya: Okay, well the next section here are various projects of which I know you've done many. Are there specific projects that you want me to interview you about as time goes on, or just the ones that I'm certainly aware of, in my memory?

Michael: Do what you want. Okay, I'll--.

Miya: Well see, what I'm going to do is--.

Michael: If I think of something, I'll inter--.

Miya: No, here's what I'm going to do, because I have a rough idea from what Dag wants and will like. But let me tell you the three kinds of questions that- for each of these projects. The first is, when did you first think about doing something, what made you think of it, why did you do it? The second thing is what is the most challenging problem you faced doing that project? And then why didn't you do a project another way? Okay, so what I want to do right now for you is an outline, just very briefly. You consent as a nod. But these topics I'm going to cover on, later on.

Michael: All right.

Miya: So first will come the various machines. We started off with the Univac, the 701, the 704.

Michael: Yes.

Miya: We should probably go over some of the other ones that are on the first form. Okay?

Michael: As I said earlier, the experience of trying to make the Benson-Lerner plotter a useful tool in analyzing programs soured, at least me, on complexity. As far as I'm concerned, simple is far, far better. And now let's see. It was 1956 that I was introduced to the idea of diagnostic data. When you make a

test, you have to take your data and then you have to analyze that data and produce results and reports and all that kind of stuff.

Miya: When you say test here- I just want to be explicit on this- you mean--.

Michael: Nuclear tests.

Miya: Right.

Michael: Okay.

Miya: Not a computer program test? Right. No shock.

Michael: Not a computer program.

Miya: The terminology you guys used is 'shocks.'

Michael: And again, one of the principal problems with the things, the way they were being done, is that it was very slow. So the criterion that you have to have -- first and foremost it's got to work. Secondly, it's got to be faster and simpler, otherwise it's not worth it. Now given that, Norman Hardy and I were sitting around ruminating about the displays and stuff like that, and he pointed out that it'd be neat to hold up a piece of film that was exposed already, and developed, in front of the cathode ray tube and plot points to see if they got through the film. That comes- that got to be called a flying spot scanner or a flying spot digitizer- we called it the eyeball. It was one of two inventions at the time. The other one was done by Phil Pederson [ph?] at the TX-2 at Lincoln Lab. I didn't know about him then. I learned about him subsequently, and he became a very dear friend. But he produced that famous Cal Comp drawing of the Mona Lisa that everybody collected. He did that by scanning a slide of the Mona Lisa and then designing a character set that would correlate with density, and he drew out the Mona Lisa with that character set. He and a few others at Lincoln Lab built a thing which they called a photo-digital digitizer, or something like that. We called ours the eyeball, and the idea was to put a holder that held film in front of the cathode ray tube, with the proper lenses, and project the image onto the photo cathode of a, say, some sort of image sensing tube- photo multiplier, photo diode, whatever it might- whatever it needed.

Now, whenever you plot a point, you know where that point is, and if you get a signal from the photo multiplier, it means that the film was transparent at that point. So you could build a mosaic of transparencies, various amounts of density or whatever else you would like to correlate it with, and get an idea of what the image is on the film. It is considerably more difficult to get all these 0- these clear and not clear spots organized into an image. That's very difficult. But your eye does it very nicely. And we built an eyeball. The first one, the first model, was on the 704. We took the camera off the 740 and put in a film holder, which also held a photo multiplier behind it, and then would plot points, and if they got through, they would signal my trigger unit photomultiplier, and we could build up a mosaic of responses. And you could maybe build a library of such responses, and say, if you have a cluster of them this way, that's a spot; if you have a cluster of them this other way, it's a line. Here's another place where it's an intersection. And there was just almost an unlimited variety of ways to characterize images.

But for us, we quickly left the 704 because it was "needed for big, important physics calculations" and moved our operation, which was a vast improvement too, over to the PDP-1, which by then we had. And you would read- though we had designed the whole thing to read film, with the PDP-1. I had a programmer that was working for me then by the name of David Mapes- quite a gifted young man, no formal training at all, as far as I could tell, but very smart. He thought like a computer. And with his inspiration and his inputs and stuff like that, we developed a set of programs which we called the eyeball reader. And you could put a piece of film into the film reader and read that film and produce essentially a set of data points that were the signal that you wanted. And then you could take that thing over to a bigger machine and run it through all kinds of mathematical filtering and things of that sort, so that you could report out- associate it with other things that were around it and so forth. And that was done- well, probably three years- and in those three years, during those three years, Ed Fredkin and Ben Gurley established a thing called Information International Incorporated, and they were building film readers, and they were excellent- really superior things. And we helped EG&G [a U.S. national defense contractor] order one, which was specially built for EG&G. And one of the measures of this kind of machine is how addressable is the raster? So on the PFR-2, as it was called, that went to EG&G, the addressable raster was 2-18th by 2-18th points. That's never been exceeded. It's still, far and away, more than anybody else ever had. And it had some sort of a PDP-1- not 1, maybe it was a 4 or a 5 or 7, I don't remember what. But it read- we gave the job of reading film to EG&G.

That removed the need for us to do it any further. And EG&G read films for years on their PFR-2. So a nice little example of a transition from a freaky laboratory critter to a viable commercial product. EG. Information or Astral build a PFR-1 and a PFR- I call it the PFR a half- PFR-1, PFR-2. And their workhorse was the PFR-3, and people all over the government in this country bought that film reader and used it to read film of various sorts. We found that you could also read paper with the thing. Instead of using transmitted light, it used reflected light. But it worked. So you could read the chart, you could read a map. We had a demo which was not really very good but you could read music on the PDP-1, held up in front of the eyeball, and then play it. And I was trying to hook up the program that Peter Sampson had written at MIT. His thesis was, using his notation, compile what he called 'voices' in the sense of Bach music, written in multiple voice. And then it would play up to three different voices at a time. So it was a natural to play the Bach trio Sonatas on that thing. And they were beautiful. I could never imagine hearing anything better. Norman did that with a guy by the name of Ted Ross.

Miya: He did that with Norman Hardy.

Michael: Norman Hardy, yes. And he did that- they used a 7000-series machine- I think it was a 704- and cut a record in Berkeley, playing flute sonatas- that's how it sounded like, beautiful stuff. So it was sort of a neat hack to play the- read the music and then translate it to play. And it never really worked very well. But, what the heck- it was just something to try. And [the] PDP-1 was called our Romper Room, where little children could play, work out ideas that had never seen the light of day before that. And it was fun. We tested the typewriters that were ultimately not used on the Octopus [the Livermore internal data network]. We tested printers and card readers and tapes. It was just a great machine. I suppose that some of the machines we have today are nice, but they've been- so much crap has put on the thing, which prevents the programmer from getting close to the actual hardware, and it's just impossible to use them. We always have an intermediary like Word or Mac or some of these other things that--. The manufacturers are a bunch of- almost criminals. They don't even let you see a programmer's reference manual, how the thing works. It would've been nice if they had persisted in that. At the time in Livermore, it was routine that with the help of these programmer's reference manuals and some other engineering details that came along with the machine, or proposed machine, that we knew more than the

salesman who came to sell the things. And it was essentially a really exciting time of being exposed to these new machines and to be exposed to what I think are four important things. Okay, I had really interesting problems, not the baloney that you find in the academic field. There's some nice problems there, but typically they're going to prove this is Fibonacci number or it's a new random number or a prime or something like that. Our problems were "real". We had some very, very good people who were helping to do these problems. We had a really important thing- money enough to do them. And lastly--.

Miya: So the first thing is interesting problems. The second was, money enough? Hold on a second. I'm trying to keep up with you.

Michael: The second one was people who are good.

Miya: Good people.

Michael: Yes. And I mean not just one person, I mean a team- engineers, technicians, programmers, physicists, the whole bit. They were really good. The third thing was money. We had enough money to do stuff. And the last thing was we had a management that was smart enough to stay out of the way, so you could- once you got a sanction of a sorts that said, "Yes, that's a nice thing to try. Don't take too much time to do it but go ahead and try it", we could go, and they didn't bother us. That was how the Octopus was conceived and I don't think the long-term experience with the Octopus has been all that bad.

Miya: I'll raise the Octopus network- because I only know a little bit about it- later on. I think what we do is detailed Octopus.

I want to get, in addition to that, CHM's other four questions here as well too, which is the overall- sort of a wrap-up.

Michael: Okay.

Miya: So we'll use the next 40 minutes for that as well too. I know that Dag is going to want me to detail all of the notable people that George Michael has encountered.

Michael: Ouw.

Miya: It's true. Well, you have to give Sid his due, as an example. And I'm lucky enough to have met Sid and Heidi and several of these other people, as well as your lovely daughters.

Michael: Yes, they're really great.

Miya: But I know Dag wants to know anything he can about von Neumann and stuff. And then--.

END OF TAPE 2

Miya: We should also talk a little bit about too, because not so many people are going to be familiar either in computing or even in the outside world, about testing and stuff like that. You are one of the few people in the world who's ever seen at atmospheric test as an example and programmers with the application-- These things you and I take for granted to a certain degree when we talk about this stuff. These are some of the things of the oral history I think that have to be captured. Let's wrap the last four questions in the next 40 minutes and then we can detail other things along these projects and other things like that. Overall, what were you saying just a moment ago about exciting people and exciting times or stuff like that?

Michael: We were lucky we were at the beginning of a new era in a sense and there were a lot of really, really wonderful people around, okay, I think much more. There are certain kinds of people around now who are more flinty eyed and hard commercialists and stuff like that but at the time commercialism wasn't quite that important to the people that I dealt with in any event and it was just a thrill to work with them, to hear what they had to say, to learn from them.

Miya: Do you think that was because of the Cold War in part?

Michael: Huh?

Miya: Do you think that's in part because of the Cold War?

Michael: No. I think it's because it's a new field, computation.

Miya: It was the beginning of the field.

Michael: Well, it was getting to be a new field and what I hear the thoughtful people saying still is that computers have added a new kind of science to what we had. Okay? I know you don't agree with that but that's fine. I think it's true. I think that computers allowing you to control reality so to say as much as you can it gives you a visceral feeling for what's going on in nature. So we have experimental physics, we have theoretical physics and now we have computational physics.

Miya: I'm just being devil's advocate but you know that.

Michael: Huh?

Miya: I'm just being devil's advocate but you know that.

Michael: Look. I don't know who's the devil here but <laughs>.

Miya: We're going to have to expend some of the time in these interviews with your humor as well too. The first question of Anel's which we're sort of in the middle of is what was the most exciting period project during your entire career? What was the most interesting--

Michael: Oh, Lord. I don't have a most exciting thing. To me going to work every day was a high. Every day was something new or could be new. The problems that we worked on were fresh, they had importance to the nation and we were making progress. That's very nice. It must be something because then there was a sort of a little kernel of people, maybe five or six, who would go back to work every night and work until 2 or 3 in the morning because it was so interesting. They didn't have to do that and just to be part of that was just-- Well, it was actually wonderful. Okay. I enjoyed it. So that was a big high for me but it wasn't just a single thing. It was a continual thing.

Miya: Were those all just men or were there men and women?

Michael: Huh?

Miya: In the five to six who would go back to work every night.

Michael: Men.

Miya: They were all men.

Michael: I don't remember any women there. I don't know that it has any of- much of a comment or anything. It's not wrong to be interested in more than one thing, and if the people that I was working with were compelled by their interest in one thing they came back to work at night. It was nice at night. There were no telephone calls. There were not a lot of people around fighting to get time on a machine, it was easier to get time on a machine at night and the administrative policy of the lab in part was that as long as you don't severely disturb production you have the right to stop a process and do your own thing, to use a machine when you want it. That's not important anymore because now the machines that you're going to use are on your desk. However, the stuff that's on the desk is essentially Mickey Mouse. It's okay for a guy who never wants to do anything with a machine, just find a word in a paragraph or something like that, and if he wants to find two words he has to do it twice, that kind of thing. At the time that wasn't the way it was. We had better control of our tools than people today, I think. It may be that I haven't kept up as thoroughly as I should but I hear these same expressions, judgmental things, from lots of people. I have never heard very much complimentary stuff about the product that's being marketing today. It is a marketing wonder but a technological blah. Okay. It just doesn't know how to handle a user. Anyway, when it comes to highs, there were several that I think is interesting. One of them was making the PDP-1 perform, all the various things. That was really fun, okay, number one. Number two was to bring film recording from a mishmash to a beautiful science. Okay. And the third was participating in a test series out in the Pacific. That's as close to Eden as I think I'll ever get. They took care of your what would you call it, routine needs to live. Okay. You had plenty of good food and the climate was just superb. There was enough time to go swimming and the weather was just friendly and even when it rained it was friendly and it was very nice to be out there. I liked that.

Miya: We should probably detail that.

Michael: Well, there's not much to detail there--

Miya: No, later. You don't have to do that now specifically. I've heard a lot of these stories not only from you but also from other people as well too. You laughed. It's true.

Michael: Well—

Miya: You go over some of the stuff but always something new comes out and that's actually what Dag is hoping for is to bring all of this kind of stuff out—

Michael: All right.

Miya: We'll do the same thing a little bit for Nevada as well. We've only come close to one thing actually that you left out as a classified event. That's because we're being easy on you.

Michael: Well, I could push in some cases but I just don't think it's worth it.

Miya: That's not for you or me to decide. That's for the historians who—

Michael: Yeah. Well, I suppose that's sure true.

Miya: You're saying very, very context sensitive words and I only have to know some of this because I've lived with you this long. If you could change or redo an event or decision in your career, what would it be?

Michael: I think I would devote even more attention to the ambiance of the laboratory. As it was, I had kind of a bad reputation. I wandered around like Miya, going all over the place saying, "What are you doing? What are you doing it that way for?" And that got me into technical photography, got me into electrical engineering, got me into machining, got me into physics, chemistry, computing, specialized programming, compilers, just all kinds of stuff, and it was largely because I was just wandering around asking questions. I would like to have done more of that if I had the chance and the people were quite open at that time and so it was just nice. For instance, one time I happened to be at Bell Labs sitting with some friends there and they were talking about this new thing called UNIX. So I said well, that might be interested- that might be interesting to use, let me- how do I get a copy? They gave me a copy. Okay. Can you imagine that happening at AT&T today? Not likely. When I was there I showed some of the films we made and they were pleased with them and they showed me some of the films they made and we swapped films without having to have any administrative approval or any of that nonsense. Well, you don't see that anymore. We're mature. We lost something. That's what I would like to change.

Miya: The lab or—

Michael: Just dealing with your colleagues wherever they were, Bell Labs. My two highs at the time where I really enjoyed going was Bell Labs and MIT and the third place that I enjoyed going was DEC and after that "bah." Then I add a place. After you knew the right people, you could enjoy yourself at T.J. Watson Research Labs, okay, but that was much, much more rigid or something than these other places. I think Bell Labs and MIT were just very, very good places. I wandered all around those places, okay, and it was great.

Miya: I know. I agree with you.

Michael: You don't?

Miya: No. I agree with you. Wandering around as you pointed out, you were like me in that regard. This is one of the reasons why I pegged you. You would basically do the ambiance is the one thing—

Michael: Yeah. I liked to savor it more.

Miya: Like a fine wine or food.

Michael: What?

Miya: Like a fine wine or food.

Michael: Oh, yeah. Right.

Miya: What advice would you give to young engineers today? If you need to take a moment to think about it, we could—

Michael: Don't specialize, number one, and study more mathematics. Mathematics looks to me like the key to everything. It-- In addition to giving you good guidance, it also puts the right kind of limits on you so you don't go off into phony science and things like that.

Miya: Phony science is like what?

Michael: Well, I can't think of anything but like the guy who says that I have equations to prove that the universe is expanding, everybody is expanding at the same rate and you can't detect it. That's weird. I have a guy saying here's a bottle and I put this electrode in it and I can get cold fusion. Those are bad examples but the only ones I could- that I can come up with at the moment.

Miya: Something that's not necessarily applicable? How does that sound?

Michael: That's fine.

Miya: Too theoretical perhaps?

Michael: No, not applicable. The theory is not bad. I am probably more than anything else a theorist. I can't help it. That's what I-- I don't feel comfortable going forward anywhere unless there's some theory to support what I'm doing but the question of how do you get that theory is what I- my advice for a young engineer or something like that is that hey, get well grounded in mathematics.

Miya: Yeah. That's what I would say too.

Michael: And you can't- you won't go wrong.

Miya: Is there anything you want to add about Anel's particular questions and then we could get into the more fun things that you would like to talk about?

Michael: Go on.

Miya: I was figuring about trying to take 10 year periods of your life and trying to get you to talk about some of these things or people, specific machines and stuff like that, and I'll try to do it based on what I think an audience who doesn't have any exposure to some of these topics would go to. For fun we can do it with some of the people first that you've mentioned and I know that there are a variety of people that both Dag and the rest of the museum want to know about. You mentioned for instance John von Neumann. Say everything you can about John von Neumann.

Michael: Everything you heard about is good or true and he's an interesting cat I must say. In fact, the first year or two at the lab there were all kinds of very, very interesting persons who visited the thing and they weren't always sequestered within the director's office. They actually got out where- into the trenches so to say and you got to talk with them. I remember in the case of von Neumann we as the junior physicists who were doing the work here were supposed to give him a briefing about what we were doing and I remember one of the guys that I worked with, Ed Leshan,[ph?] was going to give his briefing. It was on the use of Monte Carlo techniques in neutronics. So he started to give his talk and von Neumann went up to the blackboard and he took the chalk away from Ed and he gave the talk, okay, not that he was bored or anything like that. He was excited and he just went fast as he could and then he said to Ed, I hope that didn't upset you. So that wasn't exactly what he said but—

Miya: He gave Ed's talk or he just—

Michael: Huh?

Miya: He gave Ed's talk or he gave his own talk?

Michael: He gave Ed's talk. Ed was talking about some of the computational traps that can occur if you don't use the right kind of random number generator, if you don't refresh it the right way, if you let round off get to you in places, if you don't understand the difference between a direction cosine and something else. It's just those kind of things, the traps that occur when you misuse neutronic, when you use- misuse Monte Carlo techniques and it was something I guess that John von Neumann had been thinking about and it was just fun. Everybody enjoyed it. It wasn't tense or anything like that. It was good to be there. I remember very early too that Bob Jastrow, the astronomer who's now in charge of Mount Wilson I guess, one day we- he had this powder blue convertible, Oldsmobile convertible, and it was parked out there and so we went out there and put a sign on the thing saying- say "I'm sorry I bumped into your car but I didn't sign it," okay, we didn't sign it. So when he went out and he found the note he went around the car about three times before it began to suspect that he was being joshed. That was fun. One of the guys who visited the first summer there was George Gamov, interesting guy. He was a big, tall fellow with a high-pitched voice and every time he got excited in an argument he would want to bet you a case of whiskey that he was right, and it was just- again everything was so natural and free flowing nobody worried about the administrative blah blah thing. It just very nice stuff.

I think I've told you once before-- I've forgotten where but we had the DD-80 display at that point, the world's fastest display, and I had brought back what it turns out that genealogy is. Warren Teitelman at MIT had written a thesis describing handwritten character recognition where you could use time sequences to recognize a character and I got that from Cliff Shaw down at Rand Corporation and I brought it back and gave it to Bob and he wrote it and put it into his Yonka[ph?] and then we had Ivan Sutherland there visiting us. He was then head of IPTO or whatever it is at ARPA and Norman was demonstrating the thing. Norman had this characteristic that he would print the two backwards and he also would print a 'B' backwards or upside down. So he printed the two and the machine flashed at him and said you print like Norman Hardy. Well, he stopped, he wouldn't go on, and they finally talked him into going on. So he went through the alphabet and he got to the 'B' and he printed the 'B' upside down and the machine said "By God, you are Norman Hardy," and it blew Norman away. He still laughs about it. It was just really a great, great hack, okay, and we had that kind of stuff going on all the time.

Miya: Wait a second. You jumped from von Neumann to Jastrow to Gamow—

Michael: Gamow, yeah.

Miya: --to Warren Teitelman, to Cliff Shaw, to Sutherland, to Norm Hardy. I'm trying to serialize you here. Let's stay back on John von Neumann. You mentioned that Gamow had a high-pitched voice. Did—

Michael: Yeah.

Miya: --von Neumann also have a high pitched voice or—

Michael: I don't remember that. It was more normal than Gamow's was. I don't remember. We didn't get to interact too much with von Neumann. After all, he was on the general advisory committee and they usually sucked up his time in the director's office and we weren't invited into that but this particular time as we were giving the briefing about what kind of calculations we were doing, which he was interested in, he was also interested in the work that subsequently Chuck Leith did on weather forecasting or just perhaps giving the general circulation model for the Northern Hemisphere if nothing else. And some of the people at his Institute for Advanced Study worked on the first- one of the first weather models that a computer could calculate with. Chuck's great- I think the great achievement included that his would run all the time, it didn't crash because of the introduction of noise, and it would improve the more inputs it could get from reporting stations. So Chuck had to write a big program that was trying to understand the Teletype messages that came in from these various stations they'd be always garbled and stuff like that. Well, Chuck did a beautiful job on that, okay, and I don't know that von Neumann ever interacted directly with Chuck on that but that's what he was interested in. Okay.

Miya: Chuck's computation was robust.

Michael: Huh?

Miya: Chuck's computation was robust—

Michael: Yes.

Miya: --and von Neumann was interested in that.

Michael: I don't think the initial computation that von Neumann sponsored was as robust but it's also probably true that he was addressing a different aspect of general circulation. Chuck began with a set of equations that were written up by Louis Richardson and published in his little book and- but Chuck had the advantage of lots of the numerical techniques that had been evolved at the laboratory, and see, to me, I remember one of the things I was shocked at is that somebody told me that 'A' times 'B' is not the same as 'B' times 'A' on a computer and I thought "What was going on here?" Somebody was telling me that this funny thing called equal sign is a real misnomer. It isn't. Assign it,[ph?] okay, we could maybe stretch it for that, but not an equal sign. Well, all of this had to be learned and the first 10 plus years is- that's what we were doing. So it was a very, very exciting time.

Miya: Numerical methods.

Michael: Uh huh. When we started I was given a book published by Scarborough in the last century- in the 1800s, okay, on numerical methods but he had developed a set of methods that were useful on a desk calculator, not on a computer so- and the business of round off was just completely different for him. So we had to learn all that and I think we did. There were lots of problems that we wanted to do which couldn't be done because the machines weren't big enough, okay, but that sort of gives you a nice horizon, a nice set of goals to work toward, and I think we did.

Miya: One of the things I know about von Neumann you wanted me to harp on, which you harped on me because I had already read them, was von Neumann's Silliman lectures on the brain and computation. Do you remember that?

Michael: I didn't attend those lectures.

Miya: No, but I actually read the book but you told me to go off and read them.

Michael: Yeah. That's a nice book.

Miya: Yeah, but I seem to recall you said that you actually saw von Neumann at the hospital on his deathbed. Is that correct?

Michael: Well, that's a piece of apocrypha. The story is that Teller and a couple others went to see von Neumann on his deathbed.

Miya: Were you part of that group?

Michael: No. And he got them to promise that they would push computation at Livermore whereas it was not being pushed at Los Alamos barely, okay, and then before we did anything like committing stuff to hardware it would be thoroughly modeled by computation. And so Livermore from the very beginning so to say, von Neumann died in '57 or so- from the very beginning was targeted at doing models before we did anything else and everything is modeled, not just the grosser features of something. Everything is modeled.

Miya: Can you say what constitutes everything?

Michael: No. Just think about when you engineer a complicated part or a complicated thing that has many parts in it. Everything was modeled is what I said. Don't be too literal with that in the sense that a guy's got to have a square piece of metal in such and such a dimension. You don't have to model that necessarily but you may have to model how it fits into this chamber or something like that and these are just guesses. I have no idea in particular where the constraints are, where the limitations are and things like that. It's just that the machines were there, they were an incredible resource for the laboratory which had- beginning to have almost 1,000 people and a great number of them were using the machine to model physical processes.

Miya: In thermonuclear reactions so--

Michael: No, not just thermonuclear reactions .

Miya: And the physical things that go with those reactions? Well, photons, neutrons, blast effect.

Michael: There are all kinds of computations which relate to quantum physics, have nothing to do with weapon physics or modern warfare techniques or anything like that. Okay. Those are being studied. Some of the models for defining the physical parameters for an equation of state or some strange cath[ph?] or other material, okay, those have to be roughed out and then smoothed over to make sure they give the observed results as well. It hasn't much to do with thermonuclear stuff per se.

Miya: Again that's a world that most people are foreign to. When you use a term like equation of state, most people don't know what an equation of state is.

Michael: Now you do.

Miya: No, we haven't gotten into equation of state. You simply used the term. Right?

Miya: Michael: Look—

Miya: We'll get to it later. The question is what constitutes everything. What people are going to want to kind of figure out is where--

Michael: Think of an automobile Gene.

Miya: No, it's not me. I have to ask the questions for the people who aren't here.

Michael: Tell people to think of an automobile and all the parts that are in it and imagine that many of those parts are being modeled, okay. This is stupid, a little thing like here is this seat made out of mohair or plastic or something. How many times can you sit on it before you wear it out? Okay. How many times can you step on the brake pedal before you warp the brake pedal? Those are things that are

modeled. They have a bearing on the overall performance of the instrument that you're building. That's what modeling everything means.

Miya: Did you interact when you were with von Neumann and Teller with other guys like Nick Metropolis and the like?

George Michael: Well, I knew Nick because he was on the staff there.

Miya: Nick was on staff at Livermore.

Michael: Livermore, yeah. I'm not sure that he was on the staff at Livermore now that I think of it. Isn't Nick down at Los Alamos?

Miya: I think he's University of Texas now or something—

Michael: I did interact with him wherever he was. I interacted with Hamming wherever he was and certain other people at the lab, Harlow down at Los Alamos, but it's a natural thing because you're there on business and you're dealing with these people. That's fine. I don't remember ever really going out of my way to interact with somebody. If it happened naturally because of assignments or just target of opportunity, swell, we enjoyed those, but I don't remember chasing down-- Yeah, that's about true and when we went to MIT I had sort of a regular tour in which I visited people, find out what's going on and what is what but I don't think that was searching them out for this. That's who I went to see.

Miya: The community was so much smaller back then.

Michael: Huh?

Miya: The community of physicists and scientists was smaller back then in a way.

Michael: And they were collegial. Actually, I think that there- this collegiality among physicists extends just to their own little set. It's that way today. There-- There's more to talk about and many, many more people to do the talking so they restrict things by just dealing within their own specialized interest area and I don't think that's bad. It's just oh, should we say sad that people can't all be generalists.

Miya: So you were the butterfly who—

Michael: I was the butterfly. Right.

Miya: That's your—

Michael: But I learned an awful lot that way though. God.

Miya: That's your word you used for me once.

Michael: I'm not able to talk about a lot of that stuff.

Miya: We'll talk as much as we can. How does that sound? We still have a few more minutes here. Do you want to say anything else about von Neumann? He's important to the Museum especially.

Michael: Well, no, I don't think there's anything else to say about him. He was a party guy and he liked jokes but I didn't interact with him at that level.

Miya: Is there somebody that the museum should talk to you about him further?

Michael: I don't know. I'm sure you could ask one or other of the former directors at the lab but I don't know who. Ask for Mike May[ph?] and- if you can find him. He spends a lot of his time at the campus- UC campus in San Diego and—

Miya: What I'm thinking is the fact that he was also at the IAS as well too at Princeton, New Jersey, Institute for Advanced Study. There's a whole bunch of other names here we should probably go over but we don't really have the time. Is there something in your past that you want to make sure that we—

Michael: Say it again?

Miya: Is there something in your past that you want to make sure that we cover in these interviews or do you think we've pretty much covered your entire life as you think—

Michael: No. I'm-- I probably need to ruminate about a lot of this stuff for a while but you're being thorough enough. You're driving me nuts.

Miya: I have to talk to you about the computer research group and our work with dataflow and—

Michael: That's a long ways off yet—

Miya: I know. That's into the latter 40% of your career.

Michael: Now what year-- You're actually in the 1960s right now. Okay.

Miya: Late '70s actually.

Michael: All right.

Miya: But the thing is the Museum is going to want to know about the meetings.

Michael: I'll cooperate whatever way I can—

Miya: Right, and the computer, so what I'm going to do is I'm going to e-mail you these notes. We have five minutes left. Why don't we do this? Talk about a shot. Talk about an experience that most people will never have because we've stopped surface testing. What was a shot like, the preparations? In five minutes so you're going to talk about-- Go ahead.

Michael: A shot involves hundreds of people, okay, each doing his little thing. When I was out in the Pacific for some tests, I spent part of my time with the Scripps Institute divers. They were implanting pressure sensors down at 150 feet on the ocean side of the lagoon. I spent time there. I spent time in the technical photography trailer. They were perfecting things like if somehow the films get contaminated with radioactive material have we lost the signal? No, you can get it back. An 18th- 19th century book described how to do it. I don't remember the name of that book but I think the author was Mees, M-e-e-s. Anyway, so the tech photo would make sure that their juices are fresh and the person who's going to be running the master timer makes sure there is no glitches, no loose cables. The guy who's running the oscilloscopes that they want to photograph signals with-- These are very special, cantankerous kind of animals and he has to make sure that they work. I spent one time with Otto Krause[ph?] who is now dead. We were measuring what does the photomultiplier's signal look like at the very instant of dawn. The only way to do that is to measure it, okay, so we did and it was interesting. We got caught in a three day storm out on this remote island. There was nothing to eat but number 10 cans of chocolate ice cream in the kitchen freezer. There were no helicopters that could fly so we didn't get any food brought over to us but we spent I think a little more than a day and a half there until the storm broke and they could get a helicopter in. They brought food and we left with the helicopter. That was fun. It was very serious and—

Miya: When you say 'dawn,' do you mean—

Michael: Huh?

Miya: --do you mean the reaction or do you mean dawn as in the sun comes up?

Michael: The sun comes up. They wanted to see what does the photomultipler's signal look like. These photomultipliers are not normal. They are many, many, many generations deeper into the reaction time, because there's a lot more going on.

Miya: We have one minute- two minutes? One minute. Talk about the event. You have one minute to talk about the event.

Michael: I can't. I can tell you that usually the events were done in the morning, and in order for that to happen everybody had to be accounted for, and you could tell whether or not it was going to happen because they had a bank of lights up on top of the water tower and if the light was green the test was on. Then when the test was going to happen everybody was mustered down on the beach. All had glasses. It's like Feynman said, you don't need glasses, okay, you just turn your back on the thing and everything disappears, you can't see anything, and then when you can see something again it's safe to turn around. You're looking at the rising sun effectively and fine.

Miya: That's it. All right?

Michael: Whatever you say.

Miya: When I get back from Europe we can continue all this sort of stuff.

END OF SESSION 1

START OF SESSION 2, March 14, 2006

Miya: This is March 14th, 2006. This is the second taping session for George Michael's oral history and George what we'll do is I had a set of notes that I emailed you and at least 3 other historians. These will be somewhat disconnected in some ways but I think they're going to be things that are important for history and for the Museum and the like okay. So like the last set of tapes we talked a little bit about you going out for a shot in the Pacific, we'll get back to that at some point in time. But you worked at a comparatively unusual place for computing out at Lawrence Livermore [National Laboratory]. Do you want to sort of state your historic stance for working out there that you tend to give people who are not familiar with nuclear history?

Michael: That's a large order in a way. I got there in April of 1953 and I got there a week before the first computer got there, the Univac I and over the next 20 years, 25 years, the laboratory was fortunate in acquiring serial [Number] One of the best computers that were available and we had the freedom to employ these things as efficiently as one knew how. Not much was known about computing when we started so a lot of it was discovery and it was all very exciting to do things the first time. I said the last time we talked, we had four features or four components that made computing very useful. We had very good people who are pioneering in a lot of this stuff. We had really interesting problems that had national importance in a sense but even more on their own they were very interesting problems. When we had enough money to investigate in a thorough manner, and lastly we had a management that wasn't meddling, it stood aside and let us do our work. That combination led to what I think of as the laboratory participating in the Golden Age of computing.

I think since 1980, things have in some sense gone backwards in that the number of people who can be very comfortable with computers is changing. I don't think it's the numbers so much as the content of what they're learning. We have young people now who are very familiar with laptops and things like that. Any one of those laptops was in a sense, more capable than the machines we had at the laboratory which were servicing everybody at the lab but I think it's true for only a limited number of things. Particular, one of the important things in computing is doing I/O and the laptop regime just doesn't have much I/O that's very--- it doesn't have very fast I/O is the best way to put it. So familiarity that the people developed in the years gone by was supported by very, very fast I/O on the one hand. Our problem, perhaps the only real problem which has now been solved, is there was never enough storage. Now storage is not a problem, just learning how to use is it a problem.

Miya: The question I was trying to pose actually is that at this point in time, with revisionist history, let me try to yank your chain a little bit. There are people who would have said to you 'oh well wasn't working on the hydrogen bomb an awful thing?' and you would answer them. What if you ever got in shouting matches?

Michael: Is awful what?

Miya: Oh wasn't working on the hydrogen bomb an awful thing?

Michael: Oh yes, I think that in a way the rationalization that seemed most influential was to say someone's going to work on it and I don't trust anybody else to be a decent human being. I think we are barely, were barely back then. So if somebody had to work on the bomb it should be us, since we had a morality behind us. I'm not so sure that that argument will continue to hold water but then it made sense. Then we had a real enemy, at least that's what the American government tried to tell everybody, the communists were dangerous and I guess I don't really think that was the case so much as Russia was dangerous. But we've always had a dangerous opponent, however, their record with espionage and subversion and so forth is a thing that makes you think that having to work on these devices was useful and it was more useful because it had to be done to balance what the Russians we thought were going to do and I think history more or less bears that out.

Miya: Normally I use Soviets as opposed to the Russians 'cause the Russians are now.

Michael: The what?

Miya: We use the word Soviets for that era to distinguish them from the Russians but that's okay. I want to just jump briefly to a different topic, which has to do with a museum and that's, you're a founding member of the Computer History Museum and you once told me, history does not record it's alternatives, so what's the message that you want to get across to the Museum, what you'd like to see them do in terms of the recording of history from your point of view and _____?

Michael: The development of information technology, computers and things like that, was in a sense a regime-changing kind of adventure. It has clearly changed the direction in this country and I think that it's become more, what I want to say, it's become more routine now, okay? It's said that all the miracles and the marvelous technological advances that we went through are nothing but wallpaper to the kids of today. However, learning about their past and how we sort of got from where we were to what we have now is an important part of putting together the proper background for an American or for any other person who wants to study. So history, the Computer History Museum in particular is making sure that we don't lose the stuff that was done. A lot of it was unprecedented, a lot of it has been superseded by things that have been improved since then, but there had to be a start that was it. As I think I also said, if you don't learn the lessons of history, you're doomed to repeat the failures at least. Now that's not me, that was I think Mankin who said that and it figured very true, so that's another reason for learning--- for saving the past and making sure that it's available to moderns.

Miya: Do you want to make any comments about the lecture series the Museum holds?

Michael: Well I think it started out very well and… it's proceeding now in a slightly ambiguous ways. In the past there was sort of an effort to make sure that the speakers were technically astute and they were speaking from a depth of real technology. Now we're hearing, you know, the founding of this and the financing of that and it's all very important but it's not quite the same as knowing the technology and how it was used, how it was developed in the past. It's fine though I think we want to try and continue getting technological kinds of talks here, maybe a special series of speakers. But I think that their talks don't have enough technical content to be sustainably interesting.

Miya: Well the Museum has to balance its education kiddie component like it did in Boston and it's historic stuff.

Michael: Well, no that's an assumption you're making, you know.

Miya: Well it's a pull, that's what it is.

Michael: It may be true again but I'm not sure that the lecture series or these memory series things _____ acknowledge that. If you're going to talk about technology and let's talk about technology and not everybody can follow it as closely as you think they ought to, they'll get there. It's like going to school, you didn't know much when you started and you know a little bit more when you finish.

Miya: Well do you want to make any comments about the Museum's directions for things like volunteers and for women and minorities and business, come on let's say them, let's get them out on the table George.

Michael: I don't have any problem with any of those things, I think volunteers are the life blood of things like museums and special conferences and so forth. Volunteers are very important and I think the museum is headed in _____ mostly the directions it should and can go in. It has a remarkable collection and it's still open in the sense that you can bring things to it and if they can't put it up on exhibit right away, that's fine, it'll have its place later on. I think that as well if I was in charge, maybe that's the way to put it, I might have named the thing not the visible storage thing. I think that's kind of a misnomer and secondly I'm not sure that the chess exhibit was how we got there and so forth. I'm not sure that really cuts to the meat of the matter there, it's probably too complex a subject to treat as an exhibit the way it. I've gone through and looked at it and there's lots of neat signs but I don't think I learned much about the nuts and bolt development. Why was Blitz such an important chess program? Why is IBM's Deep Blue kind of in a shadow among people who follow this kind of thing? That doesn't show up in the exhibit, so in a sense it gives you a slanted look at the past reality, probably try to fix that. For a first exhibit I think it would have been more interesting to show the achievements of miniaturization for instance. That's something that you can grasp pretty well I think and you know, <inaudible> implicit in some of the exhibits that are out but some of the achievements are actually quite startling and need to be stressed more with the people who come here to look at the gee whiz stuff. I think you're just leading me down the path here, you know.

Miya: Of course, all the questions are leading. Do you want to say anything about the Museum in terms of special emphasis on women and minorities as an example?

Michael: I dislike those kinds of things intensely. I think that one stands on one's merit period. I don't care if it's a woman or a minority, its great okay and I think that the stressing of those things if anything enhances this concept of victimhood, which I think we don't need.

Miya: Similarly with an emphasis on business and computing, you mentioned a little bit of that, get it out.

Michael: Not it's not a--- I'm not pro-business anywhere but I think it's a bit ironic that we have the start of the computer business in the Univac I with help as I remember it from the Franklin Insurance Company, which is strictly business okay? In a sense, and perhaps all through the 50's decade, people in some of

the traditionalist people in the sciences were denigrating the computer okay? I've heard senior physicist's saying they're nothing but slide rules. I've heard some say that if there is such a thing as computer science it isn't and all that's kind of got rings of truth in it but it's ironic I think that on top of that, the people who had the shall we say foresight to want to use computers, were largely centered in business, that's unfortunate but we have to face it.

Miya: I want to stick with the Museum by the way, but I want to ask you this question. What do you regard as perhaps the most amazing development that you've seen in the time that you've been in computers?

Michael: I don't understand.

Miya: What was the most impression-forming invention or event in your time in computing for you?

Michael: Oh gee, I don't know that I could answer that question. To me the only difference that was important between large and small computers was the amount of memory at that time. Now it's not the case, my desktop has 4 gigabytes of memory, that's _____ that was more than all the computers at the lab ever had then.

Miya: Yeah I'll get in the story because I think that's important _____.

Michael: Well the development of storage follows the exact rules or the exact predictions that people like Ed Fredkin were making okay? That withstanding it's still in impressive thing. The rest of the computing developments also follow very, very understandable paths okay in the 40's and 50's the business of miniaturization was known, was predicted that it happened is swell okay but it's not earth shattering. I think that maybe the development of the transistor to begin with was earth shattering, that's fine. I think that the software developments that accompanied all these computers were more important, they were more important than the hardware. When I started not much was commonly known about numerical processes and the ones that had been worked out were largely attuned to calculations while you're sitting at a desk, if maybe you have more than a slide rule you may have an adding machine and so learning about the idiosyncrasies of the difference between continuous mathematics and discrete mathematics as it appears on a computer was an important set of steps that were taken, they willy nilly had to be taken. I think one of the interesting things was and sort of amusing was that on the first computer, the Univac, people generally did not trust it, yet it was a checked machine but in the fact that they didn't trust it, they employed a bank of desk calculator persons to calculate each number that was going to be produced by the machine to 12 decimal digits and when there was a disagreement they tended to trust the human done calculations rather than the machine. Now of course that's changed, I think properly since then but that maybe speaks a lot about the attitude that people had to begin with in computing.

Miya: In terms of the Museum, in terms of the history _____ do you want to say anything about Gwen and Gordon Bell as the founders of the museum basically from the period of early on when they were with DEC?

Michael: I don't remember the years exactly but both Gwen and Gordon had an abiding interest in historical things related to the computer and they have a lot of that stuff in the display cabinets in their house and it was after DEC took over the RCA building in Marlboro that they took over a part of that

building and put the museum in there. It was then called the Digital Computer Museum and that turned out to be not too good because people who didn't work for DEC didn't want to have anything to do with it. So after a while the thing was changed the name from Digital Computer Museum to something interim, I think it was The Computer Museum, and they eventually moved into a whole story that had originally been occupied by the Department of Transportation on a wharf in Boston. Down below them was a children's museum that was being run by Michael Spock, the son of Dr. Spock, Benjamin Spock, who wrote the book with which a whole generation of children were raised. So they had this thing there and they had a great deal of support from Ken Olsen, then the president of DEC and it was a bunch of grown people who were revisiting almost childhood awe and adventure, collecting things together just because they thought they were important. They got a curator and got some subscriptions and things like this sort but that computer museum there gave way to the one out here and I don't know too much about that story. In a sense I'm glad it's out here but we had a lot of serial number one computers at the lab and my effort was aimed at trying to get them over here. Not only just computers, we had serial number one actually of the IBM photostore and wanted that here too. So we had a collection of computers that we were going to have in our own museum at the laboratory and one of the directors of the lab decided to give everything over here so everything that we had collected pretty much ended up here and I think that's a good thing.

Miya: It's good you mentioned Ken Olsen because he was the next topic so that's fine.

Michael: He's a great person.

Miya: So with that context for the start of the Museum, what do you think people should be reading if they want to read about computer history, you gave me a couple of books, a few good men from Univac by Lundgren and Emerson Pugh's "Memories that shaped an industry." So I mean you want to say something about those two books?

Michael: *The Computer from Pascal to von Neumann* [by Herman Goldstine] gives an excellent early history. A lot of people don't like it because they think that the author was self-serving but I don't agree with that, I think he did a very good job of getting the history into print, that particular section. Now there are all kinds of additional books, one of the ones that I thought was pretty good was the psychology of computer programming.

Miya: Weinberg?

Michael: Yes and I think that Donald Knuth's unfinished series is absolutely superb, it maybe the touchstone of the whole field. It certainly caused a lot of things to change in the field.

Miya: What's an example of that book that changed made it more respectable or something?

Michael: No I think Donald [Knuth] regularized the steps to get from an idea to a computer program, understanding relations of mathematics in general, the development of algorithms in particular and finding a reasonable way to state that--that's what Donald did, is still doing. Everybody I know that was involved one way or another with computing was captivated by that book, just volume one.

Miya: Alright George, I wanna move onto a couple of other topics okay, tell me a joke.

Michael: I don't know any jokes.

Miya: I'm sure you will, we'll get back to that question. One of the things that you mentioned in the last tape was the importance of people staying in touch and one of the things that you did throughout your career was you helped set up meetings right in terms of organizing people, people with common interests and the like and jokes are important I know to that because you tell these jokes, you get people to attend these meetings. So let's see what I had here in my notes is meetings and user groups and for you to say a few comments about scientific and other policy communication and then the thing I have here in depth is Salishan [Conference] So why don't you say a little bit about the Salishan meetings that you've put together in the early 80's?

Michael: The year is 1980. I had just been appointed to be in charge of the computer research group at the laboratory. One of the guys that I talk with fairly regularly is Timothy Rudy who was a programmer in A division, one of the divisions, and he said to me 'Well, now that you're in a position where you can throw an oar, why don't you do something about getting the manufacturers to understand the kind of computers we need because we're not getting them anymore?' So that was the genesis of the Salishan idea, initially the idea was that we would have people who build the machines and design the machines, their bosses in some cases come to hear what some of the people at the laboratory are using their machines for. How this very large scale partial differential equations system fits onto a computer, has something to do with the speed that it can run and so forth and we were noticed increasingly that the people in the design and building of them weren't doing that, they were building machines that were easy for them to build in some respects.

So with the help of Garry Rodrigue and Tim and I guess that was all then, we proposed to the management at the lab that we sponsor an invitation-only meeting where this kind of dialogue could go on. And initially I suggested the rules are that we want a place which is comfortable and not in the middle of a huge city where there are all sorts of distractions. So the initial idea was that we would try a place called Bluebeard's Castle on one of the American Virgin Islands, St Thomas as I recall. I sent Gary there to look it over and he came back saying it was okay and so we started planning to hold the first meeting there. We'd invite a number of people and go there. Somebody from Los Alamos suggested that that was too dangerous to have all the computer designers that are important in one place, that could be wiped out by one bomb and so they said that the people who supply the funds, the Washington class, would not approve people going there for a meeting. So at that time I was talking with a guy by the name of Ray Chin who was a mathematician at the laboratory and he told me about this place up in Oregon in a town called Glen Eden Beach, it was a fancy motel and he thought it was very nice because he'd been there for a math meeting and it was sequestered, it was quiet, the food was good, why don't we go there?

I should back up a little bit. One of my philosophies was that if these things are good ideas, they need to have the support of more than just one person. So Gary and I went down to Los Alamos and I invited the Los Alamos people to join with us in sponsoring this meeting and the person I dealt with was well there were several, Norman Morris, Jack Wharton and Bill Buzby. And Bill Buzby and Gary Rodrigue were sent up to look at the Salishan lodge and they came back and said it's kinda remote being about 90 miles south of Portland, Oregon, but it's nice as every room has a fire place in it, the food's very good. So we made an arrangement with Salishan to hold a meeting there, that meeting took place as I recall in March, the first meeting, it took place in March of 1981. I think it included around St. Patrick's Day too and again my feeling was that if we could get the people to talk to each other it would be very good and one way to do that is to supply reasons for standing around talking to each other, hallway conferences so to say. So the lodge permitted us--- they gave us a room that was called the Sunset Suite and they permitted us to

bring our own wines and things like that. Went up to the Napa Valley and got a whole $600 worth of wine and we motored that up to the Salishan lodge and the staff of the Salishan really very good people. They helped us supply the wines and finger food, you know, hot hors d'ouvres and things of this sort and in general the meeting was a smashing success.

I recall sort of anecdotally Neil Lincoln who was one of the designers of the STAR-100 for CDC, sitting in a special meeting that was where Ernie Pecatti talked about the use of a Monte Carlo method for neutronics and after he finished I heard Neil say if I'd known that when I started I never would have built the STAR the way I did. That to me was the complete justification for having the Salishan meeting and at the time I hadn't any intention of making it into an institution but there were all the people, not all, but a great number of the people who came by and explicitly said you've got to do this again. The only charter I had from the lab was this is the lab sponsorship, it goes first cabin, fine, so we went and made an enduring relationship with the management of the Salishan, every year before their regular season started we went up there and by invitation hosted something between 60 and 100 people, we served them very good wines and the talking was exceptionally good. It's still fairly good but again they have this problem of talking more about management and what we wanted to do was talk about the technology. However both are needed and the Salishan is now hosting some where's between 140 and 160 people, the wine bill's gone up to about $6,000 and that money is supplied by the private companies that build machines, IBM, CDC, Cray, Sun, Sequent, I don't remember all of them.

Miya: Some people called it a boondoggle, don't you think?

Michael: No I don't think, it could be a boondoggle if it's just a big social affair but the technology that was being talked about there is very good and it goes beyond just weapons physics if you will.

END OF TAPE 2.1

START OF TAPE 2.2

Miya: Answer the question of what's the relationship of fine wines and computing so?

Michael: The way we did it?

Michael: The way we did it was to make sort of a game out of the- the wine thing, for instance I produced a sheet of paper that said what the wines were and what was the reason for getting it. In some cases, no wine costs more than $6 a bottle, in other cases the wines all had to come from a certain region in Napa or in Oregon okay or in addition, wines that were came by a varietal name. And so especially the guys from Los Alamos were very pleased that they could come in there and sample a whole bunch of wines and also have some very good food, finger foods and also excellent technical discussion. We put some blackboards into the room and they would, you know, people get up there and argue with each other at the blackboard. It was a good place. I think we justified the idea that the invitation only was the way that people got there. It would have a full day of meeting on Tuesdays and Thursday's and a half a day meeting on Wednesday and Friday and on a Wednesday afternoon was called free time or random access time where people would get together to go play golf or in the case of the talk that Ernie gave, he gave this talk during that free time and people who wanted to hear it came to hear him, it was very good. We had presidents and vice presidents of companies and stuff like that, a lot of computer designers.

Miya: Seymour Cray?

Michael: No Seymour didn't come to that meeting.

Miya: One of them though, Seymour came to _____ right?

Michael: Yeah I think so but yeah not that first one, he was tied up, when I recall, when he was tied up with making the sure that the Cray-1 was going to work right and stuff like that.

Miya: In 1981?

Michael: I think so. No, maybe it was the Cray-2 or Cray, some machine that he was working on okay. Maybe wasn't the Cray-1.

Miya: But he ultimately did attend?

Michael: I don't have any clear memory of that so I don't know.

Miya: Steve Chen attended as I recall right?

Michael: I don't have a clear memory of that either, George Bicks, they claim he was there but I don't remember.

Miya: I know George was there I saw him.

Michael: Yeah things tend to munge together when you get old and time is going on.

Miya: Well ageing is like another threshold.

Michael: It's a terrible thing.

Miya: Well no it's another session we want to extract stuff out of you. So people went and played golf, sounds like _____ to me.

Michael: No I don't think so, I think you notice that among high quality technical people they tend to take their brains with them no matter what they're doing and the venue in which their brain sits is not your worry as long as it's working.

Miya: And there was an airstrip there too.

Michael: Well I notice one other thing too, several problems that at that time weren't, you know, technical problems were not solved, got worked on by people from industry, academia _____ who were invited. First all it was the architects of the computers, secondly their technical management to the extent

that it needed to be there, third, there was some academics who were training people to replace us, fourth, there were a few government people there who supplied the funds for the purchasing of these machines and so forth okay and fifth there was staff from the laboratories, Livermore and Los Alamos. It's been broadened since then, I don't think properly and I don't think to its value, to its benefit but things change and you can't do much about that. It was then and still is difficult to contain that number of different _____ in a minimum number of uh… invitees.

One other class of person that was included in the invitation but wasn't counted was the business of graduate students who were the students of some of the academics that we invited. I was particularly enthralled with a guy from Northwestern, I can't remember his name now but he brought a couple of lady graduate students who had been trying to get jobs as they were finishing their studies and one of them had sent out according to what she said something like 200 resumes and not gotten a single bounce that she wanted and on the other hand, after she gave her little talk, she had 3 job offers and she did take one of them and the other one was named tenured professor of computing at the University of Colorado in Boulder and what they were doing was considering the experimental and theoretical evaluation of where very, very high frequency and short term pulses wander around on printed circuit boards. They don't go where you think they're going to go and that explains lots of the instabilities that computers were showing up then and that's what these two students were studying. And it was just very nice, I didn't think of us as being a body shop but it was good to see people who recognized that these students were very, very good and interesting and that there was a place for them and I liked that. I think that we have to nurture students much, much more but not the way that a lot of people are doing it. I think that the venue at the Salishan is good, where they get to talk to the guy, not some abstract business man or something like that, personnel thing but talk to a person who's actually building machines.

Miya: Did you guys learn how to hold this meeting from prior meetings or did you guys wing it, did you guys figure out all the stuff on your own?

Michael: I don't know what you mean.

Miya: Well actually there are a small number of papers by a variety of people on how to actually hold a technical meeting that I'm aware of, Barbara Simons, Dave Ridell _____.

Michael: The only thing that I think is important about any of those is the one, I think it came from Stanford which said if you want the meeting to have any kind of an enduring effect it has to last more than two days. And they gave a bunch of good reasons that's why we ran the Salishan from Tuesday to Friday.

Miya: But you had other things besides, you mentioned the Sunset Room, which is what I had notes on here, do you want to say something about the triangular logo. Who came up with that?

Michael: Me.

Miya: Algorithms, architectures and languages, which have now been changed to _____.

Michael: Well look there are all kinds of statements by very, very reputable people.

Miya: How did you come up with that logo, do you remember?

Michael: Well I'm trying to say, for instance Littlewood says if you have the right language and you can express your ideas and that and it sets the mind free to think about important things, that's the value of language. Architecture is to be properly influenced by the algorithmic structure that you're trying to implement, fine. The algorithms themselves are influenced by language and architecture, they should be. So it looked like that there was this three-way thing, you know, you could say under architecture we want to put the operating system or maybe you want to put that under language, whatever, but those things neutrally support each other in a very stable kind of structure as a triangle ergo.

Miya: You just threw it together?

Michael: Yeah I mean it's interesting that others copied it a lot, but it's also interesting that it survived this long.

Miya: I know, it's amazing, but I know that there are other similar kinds of meetings besides Salishan. You and I attend the Hackers Conference, I interact with the people who run the Asilomar Microcomputer Workshop, there's another one, there are other fancy $2,000 attendance workshops _____ Esther Dyon's Release 1.0, but you guys didn't attend or weren't influenced by any of those, you guys just did this all yourself.

Michael: Yeah. What do you imply from that?

Miya: Well no it's just a matter of, you know, extract from you how to run a good meeting, like after dinner speakers.

Michael: In our case the after dinner speaker was to have deal--- after the first meeting was to deal with something that's more or less or orthogonal to the theme of the meeting. So one year we had a guy by the name of Dr. Zemba.

Miya: Right, fun guy. Who talked about horse racing and the use of mathematics to handicap, not just horse racing but even give yourself a slight edge in a lottery that's important to some people. He was the guy the State of California consulted, when the State of California decided to start a lottery, he was from the University of British Columbia, right.

Michael: One year we had a guy talking about the infrastructure for supporting the space things that NASA was doing okay, that was very well received. One year we had Russell Brand, he's a very young guy and his exuberance on the one hand and the topic of computer security on the other was a bomb, okay? You know, not a failure it was wild success, whereas he was supposed to just talk for about 30 minutes or so, they had him up there for over a couple of hours asking questions, he was a banquet speaker. The first year we had Jack Worlton talking about the philosophy, I guess I asked him to put this together and he did it. The philosophy of supercomputing and Jack has done a marvelous job on that and he's continued doing that even until today, he's a very good person. I think the benefits to the typical attendee were just obvious, I think that people who say how to run a meeting don't know anything about high-speed arithmetic, about the proper way to preserve significant numbers and things of this sort. The people that were speaking at our meetings knew that stuff. They lived with it all the time. One year we

had a guy, Fred Housere, who was getting into the process of designing drugs from first principles and he was using a certain class of particle physics, a certain type of particle physics. He met a guy by the name of Jim Belac at the Salishan meeting, Jim Belac is a molecular dynamist, computational league, molecular dynamics and it was a nice transfer of a technology. Fred is doing his stuff now as far as I know and Belac's methods permitted larger aggregates of particles. When we started, it was difficult to calculate a gas or something like that using just 32 particles okay and now they're up to the several million, well using methods that Jim Belac had done.

Miya: Particle and cell methods?

Michael: That's one of them, that's not his way. I think, you know, I'm so fuzzy about this but the basis for this computation is called _____ molecule and among others, Bernie Alder and Tom Wainwright pioneered some of those things. See after the first test series, October 1954 as I recall, Edward Teller called. He said we had a regular Friday afternoon meeting _____ group and he said now that we've got these tests out of the way, it's time to do some physics so we want to make some time on a Univac available to anybody who has an idea that wants to explore something. So Bernie Alder and Tom Wainwright came up with this molecular dynamics thing, they were studying things like the formation of bubbles in boiling, the transfer of energy from one particle to another and things of this sort. And they were doing that as I said to begin with, with 32 particles. Chuck Leith as I recall was investigating, I don't know what to say, neurons and their evolution by a form called breeding and there were other people who came up with techniques for doing this and doing that, doing heat balance, I didn't pay too much attention to all that but I was there. I worked on my thing, was working on graphics if you will; when we started the only graphics [group] that was there was a pretty crude and we were using a CRT, we started using CRT's and the techniques for photographing were very crude and we improved on that, we got films that were very good. Now it's routine to take color photographs in movie form, all that stuff. All that sort of bubbled up from this adventuring with the Univac and yeah a little free time, and for many years thereafter whatever machines we had were taken--- they took some time to look at basic questions. Chuck Leith studied the general circulation of the atmosphere okay, he was using a set of equations that were called the primitives, first proposed by Lewis Richardson in England but Chuck was using numerical methods that weren't generally known then and now it's very common but then nothing was _____. So it was a very creative period and that's why I think and contributes to what I call the Golden Age.

Miya: Stop a moment about Chuck [Leith], you mentioned that he was interested in did you say neurons or neutrons _____?

Michael: Neurons.

Miya: Neurons is in the brain _____ the brain. Okay not neutron evolution?

Michael: Right.

Miya: You're diverging a bit from Salishan, I want to run Salishan up, I mean we'll get to these other people as well too and we'll fix the typos. I came to the fourth Salishan and stayed for the fifth and then I took off, you were a little bit disillusioned by Salishan about that time and so we were talking about doing a successor, <inaudible> the supercomputer meetings and we looked at _____, which we moved the hackers conference to right?

Michael: Yeah I suppose that's true.

Miya: So we were actually seriously thinking about a Salishan replacement.

Michael: Well I ran it for five years as I recall, something like that, then turned it over to others and it's natural for the person who's running it to impose his own taste on the thing and I felt there was too much diverging toward management and not enough attention being paid to the technology. I think it's great to know here's a guy who comes to the meeting who's a banker okay, he knows banking theory and all that sort of stuff, but it didn't have much to do with the technology we were coping with and things like that and it was very interesting but I felt that we were off track. However the people still liked the Salishan meetings so I guess it's filling a need. I tried not to influence the content, I would make suggestions to the person who was running it but that's it. Hard to stop.

Miya: Did you guys discuss from the late 80's the development of the concept of the grand challenges, who came up with that phrase?

Michael: Well I don't remember exactly, it may have been Peter Lax, the point is that after the Russians beat the US into space, people were getting to look at the question of what's wrong with our technology here and there are a variety of people who might assign this or that thing, my bias was that I think that when people called businessmen, multinational businessmen and stuff like that, get to control the content of technology, it goes to hell. There were numerous meetings about how to put the technological base back on track in this country, there was a meeting called the Lax report, there were other reports that went on this way. The grand challenges I think occurred during the administration of Bob Cooper and Bob Kahn at _____, they wanted to use big problems to stimulate technology and it may have for that matter, I'm not sure, they wanted to have a situation war room and that was put on some carrier so that they could see what's happening in the theater. They wanted to have an autonomous vehicle that I think Carnegie Melon worked on that for a while, would follow a road without people driving it.

Miya: They're still working on that.

Michael: They're still working, well, you know, let me just tell you about another prejudice that I have, I think that with various exceptions the stuff that goes on in the schools is beautiful but thank heavens it isn't highly focused. The stuff that goes in the national laboratories on the other hand tends to be more highly focused and the two should work together, but they don't too often. The Salishan was an attempt to try that to change, the business of the supercomputer meetings that we're still having was also an attempt to blend those two things together. That's why you want to have students coming to these technical meetings. That's why you want to have projects in the schools that are being influenced by the tastes in the national laboratories. Now if you don't do that then you get stuff that what we used to call Harvard MBA mentality. They know the cost of everything and the value of nothing, that's unfair in a way probably, but that was the general theme. All these special meetings that were held at the national level, there was even an abortive attempt to have a general meeting called the futures meeting. They had very well-meaning people running them but it was too political and not technical enough to generate really new results and stuff like that, that would be useful. I think the grand challenge is one of those things. It may now be starting to show some fruition, the last time we had automated vehicle travel in a desert somewhere down south, actually the big sponsored ones didn't win. I mean it was some bunch of kids from Stanford right?

Miya: Yeah but they had Volkswagen, they were sponsored.

Michael: They had Volkswagen support?

Miya: And Intel and a bunch of other computer companies.

Michael: Well okay you can have support all you want but unless you have some sort of a guiding intellect, just cut the fluff away and you can still concentrate on what's "quote" important. That's a great art form knowing what's important and what isn't.

Miya: Do you want to say some stuff generally about managing computer people and leadership and dealing with bean counters and small minds?

Michael: There's only one thing that seems to always come up is and I think I saw this from John Bentley but I can't remember for sure, managing 6 programmers is like trying to herd 200 cats. I think, you know, you take the corollary is just get good people okay that are well trained and take the time to tell them what the problems are that you're interested in then get the hell out of the way and let them do it. That's all it takes and I insisted, there was plenty of evidence around that shows that with that style of management is used, they get great results. I think at one point Bell Labs was that way, at one point the General Electric Research Labs were that way, at one point the Livermore Lab was that way. Get good problems. Get good people. Get out of the way. Well I should actually say also, make sure that there is enough support so they don't have to waste their time seeking out support.

Miya: We talked a little bit about high-speed computing but a lot of the work you did was actually in graphics, especially on smaller machines and it was a surprise to me when you knew so many people in graphics that I knew like Lou Katz and Kelly Booth and a whole bunch of other people. So tell me about the early days of Siggraph. You were one of the founders of Siggraph _____?

Michael: Yes, Siggraph is special interest group okay, in early days it was just a bunch of guys getting together and to share ideas à la the Salishan model and it grew and grew and grew and I think that the break over point was 1977, the meeting was held in San Jose and they formalized the exhibit area, it kind of got out of hand right away and as a result technology tends to take a back seat. We have a lot of this art stuff, I remember one person there saying I don't want to know anything Octal, I want to know art. Well that's fine for somebody but I think when you're in a pioneering industry you better know something about Octal if that's what it takes. So the first Siggraph meetings were just small intimate things where some really good people were getting together and some actual stuff got taught that way. I think the last of those was at Bowling Green and then the next one we had was at San Jose, Steve Levine was in charge of that one as I recall. He and Ray Desasure both from Livermore and they turned it into an extravaganza yeah, it was very well received, Siggraph got to be a real cash cow and for a while they were publishing fantastic stuff, books showing how movies were made and things like that. I think it missed the point a little bit but that's the direction it went and I don't know where it is today because I quit going to Siggraph because it just was too big. I couldn't meet with technical people as well as I'd like to.

Miya: Well since I was from that era I know that when we were working, we could barely afford essentially black and white graphics, but out of Livermore came color graphics and the most visible person was Nelson Max, tell me about the hiring of Nelson and were you involved in that?

Michael: I didn't have much to do with that but for color graphics I recall one meeting which was held at JILA, the Joint Institute for Laboratory Astrology or something like that at Boulder and there were people talking about using color and stuff like that. I showed a movie there that had been made about ten years before then--it was better than anything they had done. So the business of solving color graphics, again just get the people who know what they're doing and let them do it. We had superb, superb film processing, it was a film processing problem, once you learn that what they call the modulation transfer function for a recording, you build in the response from the phosphor, the response from the time on blank, the response from the processing temperatures in the soups and things like that. Everything built in, a response from the lens and what you get is an absolutely excellent piece of color film. People in Hollywood have been making color film long before these guys were in even knee pants and they did very well. Did you ever see any of John Whitney's films?

Miya: Yes.

Michael: You know, this is before there were computers, he was using a gun sight calculator to move his little trays around and stuff like that. Well we had the help, Dave Dickson at the lab, in charge of technical photography and there was Claude Ditmoor and Bob Ready, these guys are masters of their technology and all we did was to get that together with computers, this you say, we want to do this, they show how. We couldn't get decent pictures off the CRTs that were attached to the computers, Dave came in and he redesigned things so he could use a different lens. That made a difference between night and day, we had a guy from Kodak, at least one. Chris Zeutz who used to bring test emulsions to the lab and we test these things in actual things okay and it would be--- then he'd go back and they'd do their densitometry and come back with another emulsion and that was the way we evolved films and phosphors and unblank times and unblank, you know, the question of how you send an unblank signal, the ZX signal, you just don't send an unblank signal, it's got to have a shape and that shape will be influenced by how the phosphor responds to stimulation, blah, blah, blah, blah. The book I used to begin with was solar star and valley that was of a Meridation Laboratory Series at MIT from the war. Talked about phosphors and talked about CRT's and talked about radar and things like that. Anyway color photography followed a very well defined technological development path and it was mostly sitting there when these people came in through Siggraph and could use it. They could go and buy films, we don't even need film anymore right?

Miya: I don't disagree. Do you want to say something about all of your other friends, you seemed to know everybody in this field, Alvy Ray Smith.

Michael: I know Alvie yeah he's a great guy.

Miya: Who else do you know?

Michael: Look when I was--- I don't remember what year it was but it was the year that Steve Levine came to work at the laboratory and he has an incredible amount of energy and he wanted to work on graphics and moviemaking and stuff like that and he arranged a trip for several of us to take with him and we went to NYIT, we met at [Ed] Catmull and his crew there and Alvy Ray and the guy founded Silicon Graphics, Jim Clark. Well they were all there. Bill Ectra was there and a lot of the friendships that were formed there endured for a long time see. These are the people who put graphics so to say in the big map, so you get to meet them that's fine.

Miya: Should we kill the Siggraph topic to death?

Michael: We're finished with it, you know, see one of the things that I remember when Kelly Booth was on the board, he got funds from Siggraph to do a history project.

Miya: It's still alive actually.

Michael: I'm not aware of that anymore, I know I was asked to help and I did and Steve Coons was one of the persons he had left MIT by then and I guess he was at Colorado, I don't remember the woman's name who was running the thing.

Miya: Could have been Sarah.

Michael: I'm not sure I don't remember, anyway her particular goal at that time was to record in various forms the very early.

END OF TAPE 2.2

START OF TAPE 2.3

Miya: So we just ended on Kelly Booth. I seem to recall that you had hired Kelly Booth.

Michael: I don't think I hired him so much as I recognized that he was an absolute gem and I worked with him. Same with John Beatty. Same with Steve Levine. We were real fortunate in the sense that we were able to gather these people together in the so-called graphics group. Bob Lee was put in charge of that. And graphics suddenly had a status at the lab. It had a budget and manpower and stuff like that. Up to then it was on its own so to say. It did not have any support. I remember we made movies, because they were easy, not easy to do, but you know, because they did show things about partial differential equations that you do not otherwise see.

Miya: Why?

Michael: Well, there is information in the motion of fields, things in the fields. And you do not see those when you are looking at a printout. I remember that Chuck was always showing that, equally as a musials [ph?].

Miya: Chuck Lee.

Michael: Yeah. A projection of a north hemisphere, with weather patterns moving around it. And he had been running it for quite awhile. Suddenly he sees that he has got something. He has got a bug in there, even though it was running, and it is stable. He has a bug in there which is causing the zone flow, zone flow at the North Pole, to move in the wrong direction. He would have never seen that without the motion. Anyway, we made movies of this. We made movies of interesting things, classified things, and so forth. And at one point a guy by the name of Eugene Fubini, who was the Assistant Secretary of Defense under

Harold Brown, came to the lab for a briefing and we were asked to show the movies. I just casually made the remark that we had made these movies by just jumping in and using time that otherwise would not be used, because we did not have an account or anything like that. And he turned to the director of the lab at that point, at some point later, he said, "You know those movies that are bootlegged in here, make sure that's the way it stays." So in other words, you know, he was treating us as "true believers." We are going to work on what we think is right in spite of obstacles. And well, that is when the graphics group was formally inaugurated, that bootlegging, sort of that went away.

Miya: This is all before PowerPoint. You are talking to generations which

Michael: Oh, yeah.

Miya: Use PowerPoint.

Michael: Way, way before that.

Miya: Like the media, multimedia. You mentioned actually earlier, we were talking about get away a little bit. You pointed out somebody who was in A Division. You probably should describe for historians on the tape a little bit about the organization of the laboratory. You mentioned the Livermore Multi Megaton Group. You mentioned the research group. So you should say probably a little bit about, up to the point, you know, do not get into anything classified.

Michael: Yeah. That's right.

Miya: We will get into the next topic of security next.

Michael: Well, as far as the Lab is concerned, I suppose everybody has a different vision of what it is, but the way I saw it, I am not sure I can dredge this all up correctly, but there was a contract called Whitney, which was, you might say, nuclear engineering. There was a contract called Sherwood, which was the controlled release of thermal nuclear energy. There was a contract called Bio-med, which was to study the radiation effects of living tissue. And then there was a support thing that supported all these other projects with machine shops and chemistry and hardware generating it, things of this sort. We were in the weapon division, Weapons Division. And there was A Division, and B Division, and L division, which has the job of managing things in the field, you know, for tests and stuff like that. And there were other divisions that I did not know about even, but that is basically the way it was. And the lion's share of the budget was going into the Weapons Division. But the others had a healthy dollop of money and they were able to do very interesting things. I do not know what more you could say, that things could have been better, clearly, but that is the way it started out.

Miya: So was computation initially in the Weapons Division, A Division?

Michael: Well, theoretically computation was in the support area, right? So it supplied computer time and computer expertise and personnel and so forth, to anybody who had the money to buy it, pay for the head count and stuff like that. And since the weapon division had the lion's share of the budget, it had the lion's share of time, lion's share of computer resources. It was the justification when we went for a new machine, things of this sort. We used to use the needs of A and B Division specifically to justify the

acquisition of a new, faster machine. That was the principal characteristic of all the machines, so-called large machines that we got. They were the fastest things available at the time we bought them.

Miya: And that was heavily motivated by the cessation of above-ground testing?

Michael: No. I think I told you this story once before, that von Neumann, who was probably the most interesting 20th century mathematician. But anyway, he was in his role on the General Advisory Board of Atomic Energy Commission, was dying. I think it was about 1957 or so, and he allegedly called together the people who were important in running the lab. That meant Teller, E. O. Lawrence, and I do not know who else. And he got them to promise that at the Livermore Lab computation would precede testing. And so everything that the lab was going to do was modeled in computing programs before you tried to build it. So that is sort of how things got started in there. When you are wanting to model things, you have to continually improve the model, so to say. And the way you do that is get faster machines. And inside the models themselves you can run them longer. That is one way you want faster machines. You can add detail, zoning may increased so you have better structure. And you can insert more modern or more advanced physics into the models.

Miya: Algorithms you mean.

Michael: Well, I do not know if you want to call physics an algorithm, but it is alright if you want to.

Miya: For a particular process, an algorithmic process.

Michael: Well, algorithmically is the way you do it, and you are guided by the algorithm itself, plus maybe some constraints that come from complexity analysis. But complexity is done at many levels, and it is not clear which one is the most influential for this thing. However, when you say you put in new physics, there are a variety of ways to do that. You can put in a recipe. You can put a table in that says if the pressure is this, the temperature is that. What will the velocity be, etc, like that, or things like that. Instead of the table, you put in the actual equation model to describe that physical process and let it run, and use its output as input to a next step, and so forth. Those are the ways that you improve the calculations. The next step is we have to get faster machines. We have to get them to be bigger. So the characteristic up to the time that I think stopped calling it the Golden Age, they characteristic of each machine was that it was quote bigger and faster in all respects than its predecessor.

Miya: When did the Golden Age end, by the way?

Michael: I tend to say it is 1978, but you could argue a year one way or the other there. To me, when we got the Univac or the 704, the 701, or the 709, 7090, the Stretch, the LARC, any of the Seymour [Cray] machines, they were always characterized by the simple thing that they were faster than the thing we had. Then when we got the [Control Data Corporation] STAR, suddenly that was not the case. In addition to being improperly designed for the kind of stuff that we were going to do, it also ushered in the new idea of vector computation. And given that, no one knew how to do that. And so we went backwards, and that ended the Golden Age.

Miya: Never resumed.

Michael: It is a matter of opinion whether or not we are better or not. I think the stuff that they are getting out there is not exciting. I think it is just a mute statement of that is what the manufacturer knows how to build. He cannot build anything that is really good. And this is particularly true for IBM, I suppose. Their marketing model dominates on what they sell, rather than what is faster. Is it cheaper, easier to produce? Could I produce like a cookie cutter, produce many of them? And in the process of doing that, two things are lost. The advanced ideas about architecture disappear, so to say. And the software that is needed to control this machine is either non-existent or gets to be so baroque and so turgid that it cannot really serve you in the way you want to be served. I could say anecdotally there are all kinds of demonstrations of that. I do not think I want to go into any detail, but it is just- well, I will mention one thing. People who are trying to use these machines had the criticism that, at the time, they would get, and have to run on end machines, which were a subset of all the machines that were available. The only problem is that if the machine has a fault or if it gets into an overflow or underflow condition that causes it to stop, it does not report that. So now its role in supplying upper ends to the neighbor computers disappears. Then they stop. So at the end of your time shot, you do not have anything except stopped machines. You never found out about that until the end. Now that is the story that several of the designers have told me.

Miya: That is why IEEE floating point was developed for underflow and overflow.

Michael: Well, yeah, but this is far past underflow and overflow. I gave those just as examples.

Miya: Right. Those are examples. We will get more into IBM. I have more here about IBM, as well, too.

Michael: Anyway.

Miya: Let me ask you this question here, though. Getting back to the fact that you have the Univac, 704, 709, 7094, the Stretch, LARC, Seymour machines, what was Los Alamos doing during this time? Weren't the people at Los Alamos doing computation? Didn't Los Alamos get the first Cray I, as an example?

Michael: That was a punishment for Livermore. Livermore did not get the first Cray I because it had Stars.

Miya: It had how many Stars? One, two.

Michael: Two.

Miya: Four, three?

Michael: Just two.

Miya: Do you remember how many Stars were made in total?

Michael: No. I do not know that. It is not very many. The Star, how do I put it? If you have an architecture in the problem, the problem architecture matches the computer architecture suitably. The Star will run reasonably fast. But the kinds of things that you are interested in doing in physics or in

modeling in general, which is based on physics and mathematics, do not lend themselves too well to the Star architecture. So there is an inherent mismatch there and that slows the machine down.

Miya: Is that due to hydrodynamics algorithms, where you are having phase transitions and things like that?

Michael: Partly. But it is also just a simple thing like at the end of a run, at the end of a cycle of calculation, you want to adjust the Delta T, the time step. That is not a parallel operation. So all of that parallelism sits there and is drained away while you calculate a new Delta T. It is a one process thing. And then now you can start up again, but now it takes an awful long time to get going. And that is, you have lost the speed advantage that you had. The Star had a basic design flaw, in that sense that there was a disparity in the operating speeds between vector and scalar calculations. That was corrected to some extent in Seymour [Cray's] designs.

Miya: Well, that was short vector versus long vector, wasn't it?

Michael: Well, that is part of it, but it is just a simple thing of saying if you have to go into a scalar mode to compute something, all the speed advantages that you might have had from parallelism disappear.

Miya: Amdahl's law basically.

Michael: Yeah. And there is nothing you can do about that, you know. It is inherent in the mathematics.

Miya: Well, algorithms are by their nature sequential. It is the data that is parallel.

Michael: No. If algorithms by their nature are developed very easily around from layman architecture. There is a question, can you find a way of designing a machine that can be made parallel? I think you can, but I do not know that they have done it yet. There are all kinds of things, you know, simple little tricks. When the memory was on the drum, you could calculate that after you get this operand or this operator, the drum will have rotated a certain amount, and that is where you want to look for the next operator or operand, not in between. Those kinds of things, that is called optimized programming or something like that. That is just a trivial thing, but it illustrates that you do violence to the algorithm itself if you want to try to exploit the architecture that is being given to you. I have argued for years that what we should do is turn the tables around. We have plenty of good design automation, so let's just start with the algorithm and see what kind of architecture will make it the most efficient thing you can. And then build that. But that is not the way engineers or IBMers, whatever they are, work. Just does not work that way. Maybe not. Maybe impossible. I do not know.

Miya: It might be. The infrastructure for building machines is very expensive these days.

Michael: If you took and put in nearly the effort on that approach, compared to what we have got now, I do not believe it would be expensive. I think that there was a time when we were looking at design automation to help design the dataflow computer. But what is a dataflow computer? That is a Jack Dennis machine.

Miya: Right. I will get to that.

Michael: Well, I am just trying to say that that is.

Miya: Jack is on my list of people we will get to.

Michael: Alright. It is a thing that works. You know, it is a first approach to start with the architecture.

Miya: We will get to the S-1 in a little, too. So you laugh.

Michael: Right.

Miya: I can remember these things. Let's stay on the STAR-100 for a moment, though.

George Michael: The who?

Miya: Let's stay on the STAR-100 for a moment. Now Frank McMahon, as I seem to recall, was heavily involved in that?

Michael: Among other things.

Miya: Right. And I seem to recall actually that Frank paid a serious psychological toll for that. But the reason why I bring this up is the fact that from Frank McMahon is where the Livermore loop's benchmarks come from. Livermore forced _____.

Michael: Chuck Leith noticed that with some little effort you could arrange to keep all the arithmetic elements in the [CDC] 6600, the 7600 and so forth busy. When you start to multiply, that multiplier is out of the loop for awhile until it finishes it. So there is some bookkeeping kind of things you could do in the interim. Then you can start an add. By the time that is finished, the multiplier is ready to accept its next job. So that was formalized in the stack loop.

Miya: We were talking about Frank McMahon in the context of the STAR-100's when he developed.

Michael: Well, I was telling you about

Miya: Chuck Leith.

Michael: Chuck came up with this thing and the programmer he used was Fred Andrews. And they produced a thing called stack lib ['stack library'].

Miya: Right. I remember that.

Michael: And that is something that Frank McMahon took over. Now, the stack loops, or not loops, but the Livermore test problems then were a set of things that McMahon and, I think, Lansing Sloan had something to do with. And that was to just evaluate potential computers that we might get. Turns out that the kind of stuff that Jack Dongarra's linear algebra stuff produces is not very useful to predicting the behavior of a computer when it is trying to run a partial differential equation test. So the Livermore loops are. And they are not the best in the world, but they are a step in the right direction. We used our own version of those to show the designers of computers that structure of the kinds of design calculations that we are interested in. And those things went all over the world. People were learning they could test their own languages on this kind of thing. They could test their machine designs on that kind of thing. One was a Lagrangian hydrodynamics model. I do not think that was called simple. There was one that was Monte Carlo for neutronics. That was called TART NP. And it was one that was Eulerian hydrodynamics called MUD, Mud, MHD, but it was Mud. And that one did not get completely flowered up, but it was useful. The difference between the Lagrangian and Eulerian hydrodynamics is enough that one will run well on one machine, but it will not run well on another. And it does have big implications in round off and stability in accuracy. All those things figured in somehow. So that is the genesis of that. Now, I brought over one of their final reports and left it here for the Livermore loops. And you could argue that they did not receive as much testing as Jack Dongarra's. But by the same token, you could also argue that there is no point in trying to run it on a VAX's or something like that, because that is never going to be a major production facility for these very, very large problems.

Miya: Right. We will get to that when I bring up Sid Fernbach again, that whole class.

Michael: Oh, Sid Fernbach, yeah.

Miya: We will get to Sid. You have good memories of Sid.

Michael: Well, yeah. We will get to him.

Miya: The whole performance issue, there is a whole lot of other stuff. Horst Simon wants me to ask you, for instance, what the origins as you recall, of floating point operations. And I seem to recall you telling me there were two guys who were working on the 6600 having to justify and whatever the two guys were, I do not believe Frank McMahon was one of them. Lansing Sloan might have been one of them. But do you seem to recall? Where did the megaflop come?

Michael: Long before the [CDC] 6600.

Miya: Right. Go ahead.

Michael: In the case of the 6600, though, one of the things was that as a result of, in Seymour's design, the result of A minus A, or A operand A, yielding zero. He did not clear the register. So there was some, the mantissa would be zero, but the characteristic would not be. And since zero was a very important number in hydrodynamics, it was making the 6600 run very slow because there was no way to test that thing to see when you actually had a zero. So Seymour fussed about it a little bit, but he eventually bowed to the request that he add a cycle and clear. When the number produced was a zero, clear the whole register, and then the 6600 all of a sudden picked up factor, a magnitude and speed or something like that, just because it did not have to calculate in cold zones, which were characterized by yielding zero

for shocks or whatever else it might have been. So now on the business of classes of machines and so forth.

Miya: No, no. We'll get to that later. I am just trying to find out where did the floating point operation become a metric? And what I recall.

Michael: Become what?

Miya: Why did the floating point operation become a metric? Why are we saddled with megaflops and terraflops and petaflops as our justification for buying machines these days?

Michael: Well, you know, a simple example is ask yourself the question. If you have a machine that will run your problem, doing a petaflop as opposed to doing a megaflop, clearly you want to pick the petaflop thing, all other things being equal. So the number of flops or petaflops, or anything else like that, is a first class measure of the machine's performance capability on your problem.

Miya: But who thought of it? I seem to recall you mentioning two names with the CDC 6600.

Michael: I do not remember that. It is sort of like in the case of the word "supercomputer", I do not know who thought of that, either. But I always try to blame Jack Worlton, but he will not accept it.

Miya: A lot of people blame you.

Michael: No, it is not me, either. I think it was some marketing guy.

Miya: I think it is distributed syn, you know, when they had synegers.

Michael: Well, it is a distributed blame in any event, o.k.? The case of megaflops and stuff like that, if you are sitting in a position having to choose the machine, you would like to know that on your problems machine A runs ten megaflops. Machine B runs 12 megaflops, whatever it might be. Those are numbers, now, that are put into the other equations that help you make a decision. The 10 megaflop machine costs $2.00. The 12 mega machine costs two million. So maybe you have a lot more of these 10 megaflop machines, rather than two million on one machine. But it is a thing. It became important when you had that kind of a metric to be assured that the metric was based on your kind of problems. That is very important. If you are going to run Jack Dongarra-type problems, that will not tell you anything really about your problems. So that is why the Livermore loops got pushed, because that was, when you go someplace and you are going to test the guy's machine, you run those problems and take the numbers that come out of that, and convert them to megaflops or petaflops, whatever else it might be.

Miya: But you do not remember anything about discussing this about something going in for the Congress, or that era in the early '60s or maybe even '50s now?

Michael: Louder.

Miya: You don't recall when it sort of took off.

Michael: It is in the late '60s or early '70s that these things got codified. And then to make it even simpler, we had these classes, a class-6 with such and such, a class-4 with such and such. I cannot remember now the spread of performance in each class.

Miya: A class 6 machine, the canonical class-6 machine, was a Cray I. But the Iliac IV could be there. A 7600 I seem to recall was a class-4 machine, which would put a 6600 as a class-3 machine.

Michael: Right. The 6600 was the first machine that we had that entered the megaflop range.

Miya: But you did not go in front of Congress and say, "We have a class-3 supercomputer" back in 1964. Since you have come up with that class.

Michael: It was once Congress, but it was to go to the headquarters of the AEC and say, "We have devised this scheme for evaluating computers." And to make it easy to talk about, they are called classes.

Miya: Was Sid the only one who worked on that, though?

Michael: We were the only ones that did it then, yeah.

Miya: Yeah. And Sid produced a book about that. And people were always asking, well, what's a class 7 machine or a class 8 machine?

Michael: Well, those things did not exist when he produced the book, I think.

Miya: No. They were on the fringes, the ETA-10.

Michael: Well, yeah. I mean, yeah.

Miya: But lots of people were asking, well, what's a VAX's, as an example. And Sid's answer was, "Oh, it's a class one-half."

Michael: O.k. I did not ever hear that. All those smaller machines were very, very important, but they were not going to do the big physics problems that people want them to do.

Miya: I seem to recall you called the VAX's marijuana or drugs or something like that.

Michael: Well, the marijuana of the academic crowd, yeah, yeah. Well, I mean, you know, they sell everything in terms of that. But there is another way to say the same thing. If you only have a hammer, everything looks like a nail. That is the kind of thing I feel that the government or whoever else was funding the academic world, was shortchanging those people by not letting them have adequate machinery. But I also admit that you can develop a lot of stuff on a VAX, which will apply very well to a

6600 or a Cray, or whatever else follows that. Software is software, except there is this crucible of facts. It is this, how fast it must be. And you find out that it is not that fast, then no amount of polishing or blah blahing on a smaller machine is going to make a difference. Norman used to say, Norman Hardy, used to say, "If you are in trouble with one small machine, getting more small machines won't help." I think he was almost uniquely responsible for seeing that the vision at the lab on large machines stayed aimed at the right direction. And he saw long before anybody else did there timesharing, file management. He still fools around with stuff like that. And it is very entertaining.

Miya: I would say Norman then probably prevented the lab from getting a connection machine.

Michael: Would have what?

Miya: That Norman probably prevented the lab from getting a Connection Machine.

Michael: No. I think the reason we did not get a Connection Machine is because it really was nothing more than Iliac IV in one package that was smaller.

Miya: But Los Alamos got two of them.

Michael: Who?

Miya: Los Alamos got two of them.

Michael: Los Alamos did a lot of strange things. I told you once before while we were busy timesharing, those guys were using these big machines as stand alone batch processors. And it was a Renaissance there when Sid conned them into using our timesharing system, and they used our mass file system. And in every case they did a superb job of documenting which the lab never did. We had junkie stuff for documentation compared to what the Los Alamos people had. They were documenting our stuff and they did a very good job. And they actually were kind of like advance men publicizing this stuff all over the world.

Miya: Was this the LTSS operating system, or the CTSS operating system?

Michael: Well, it was LTSS. CTSS is the same thing essentially.

Miya: Right, the Cray.

Michael: It is just with the Cray, yeah. And the LTSS, they never got into that too much.

Miya: Right. The compiler was called.

Michael: LRL Trend.

Miya: LRL Trend. That is right.

Michael: That, to my mind, was not a very elegant sort of thing.

Miya: Then they got Civic. The big problem it seems, in the inadequacy of the floating point, was the evolution of application codes from one-dimensional code to two-dimensional code, and in turn the three-dimensional codes. And the effect, or I should say the percentage of time spent doing floating point calculations successively decreased as you added dimensionality. So that by the time you are getting to three-dimensional code, not only are you dealing with more data, but you are spending more time looking up the data, finding the data, where you want to put the data. And it would seem more logical, I would have thought, that just simply an instruction would have been sufficient as opposed to a floating point operation, which is a mere fraction of the performance. Wouldn't you say?

Michael: No.

Miya: No. You disagree.

Michael: Well, yeah. It is a large order. I am not sure where to cut in on it.

Miya: We will get back. Do not worry. You mentioned with Norman, as an example, the operating systems. Why not use instructions as opposed to floating point operations, especially as you skip.

Michael: Let me give some numbers. These are very approximate, but they are order magnitude correct. The Univac time, the calculations we were doing, maybe amounted to having 200, or the equivalent of floating point operations per zone point, per cycle. By the time we got up to the Cray-1 or Cray-2, or the X-MP, there were closer to 200,000 floating point operations per max point, per cycle. That comes about by having a refined model, by having more advanced and more articulated physics representations in those models. And so when you see that much of a spread, the other problems in general are hidden in the overlap. In other words, you cannot do anything when the multiplier is running or the adder is running. You just have to wait for them to finish. Not quite. You can do the indexing. You can do the address change for the store. Whatever it might be, a lot of bookkeeping type things. Yes. There were billions of calculations, or billions of operations, but you do not see them, because they are buffered, you might say. Does that make sense to you?

Miya: Yes.

Michael: Okay.

Miya: But it is just a problem. You are just simply hiding it away.

Michael: They do not pay for them in the sense of time. The wall clock to finish the thing was determined by the floating-point operations, because you do not. The other things are buffered. It is the same thing when you are bringing data in from a disk or putting it out onto a disk, you do not see that time at all because it is hidden by the floating-point operation. Yeah. You need it. But it is hidden.

Miya: Well, it is like a floating-point operation done on graphics does not count toward a floating-point operation.

Michael: Well, I do not know. That may be an entirely different thing. But this I am talking about is you know that you are going to need something.

END OF TAPE 2.3

START OF TAPE 2.4

Miya: Okay, well, we're going to say a lot more about numerical solutions to problems in the future. But I do know that one of the other major areas that you're interested in was early computer security. And you kind of collared me, and I ended up writing a paper about our group between Ames and Livermore working together in early areas of computer security.

Michael: Yeah.

Miya: Now we had a whole bunch of urchins. You have what's referred to as a Q clearance. You might state a little bit for people who don't know what the Department of Energy Q clearance is like. So go ahead.

Michael: It's just a clearance for which it determines what level of classified information you're allowed to see.

Miya: Department of Energy, AEC.

Michael: Yes.

Miya: As opposed to other.

Michael: But in general, it permeates the whole government structure. There's a level that you're allowed to see something, and whether you will see it or not depends on another evaluation of whether you need to know this or not. My philosophy's always been, the more you know, the better you can be. The security people like to keep everything in the dark.

Miya: Compartmentalized.

Michael: That's because they don't understand the problem; they just know how to keep this as a secret. Ben Franklin says, "You can keep a secret with even three men, provided two of them are dead," right? <laughs> And the reason I collared you is it follows the overall philosophy I developed years and years before that essentially, work together. It makes sense. It also give you more leverage. So the idea of having a collaboration between the laboratory and us and NASA was a very clean, neat thing to do, you know? The only two areas in this Northern California that were trying to do big problems. NASA can't run on a VAX either, you know? <laughs>

Miya: Who did we collar? We grabbed a whole bunch of...

Michael: Well, I don't remember all the people. I remember Peter Neumann from SRI. And I remember Eric.

Miya: Eric Fair from-- I think he was part of TIS, Technical Information Services, right? And we got Milo Madean [ph?].

Michael: No, Eric wasn't, I don't think so.

Miya: I thought he was. We got Milo Madean.

Michael: Yeah, we had Milo.

Miya: We had Peter Yee.

Michael: I don't remember him, but yeah, I can believe it.

Miya: Yeah, Peter's one of my neighbors right now.

Michael: Good!

Miya: And a whole bunch of other people. But the big thing that you tried to do, though, was keep productivity going. That security should not be obnoxious, right? It shouldn't obstruct work.

Michael: It should be unobtrusive.

Miya: Yeah.

Michael: Yeah. I'm sorry that it's necessary at all, but I've agreed that we need security. So the question is can we get it into a form where it won't impinge on your work too much. Because you maybe have to go through a preliminary setup everyday, but then while you're in that, you can do whatever you want. At the lab there are some issues of security which I find difficult to understand, like when you have a classified document opened up in your office and you want to go to the bathroom, you have to lock the office, and before that, you had to put the document away in a file cabinet, filing safe. That seemed to be a little heavy-handed, I suppose is the way to put it. But again, it comes from people who are involved with security who know how to stop a spy from getting this and that. But they don't seem to have any idea what it is they're protecting. And what is the relevance of that to the overall picture and stuff like that. You know, I find practically all the time that they're mule-headedness just makes them lose respect in my eyes. They can't tell what is to be protected, or why it's to be protected. I have some anecdotes, but I don't think we'll bother with them.

Miya: Just say they're classified.

Michael: Well, no, it just seems picayune. Let me tell you one of them. In the '70s, as I recall, we were evaluating special keyboards for graphics. And we had this one, and it wasn't working quite right, so we

had a technician come in from the factory, and he was fixing it. And he fixed it, and then one would test it. So where it constrained where we could have this guy, because of the classification. So here we are in this office, and there was nothing in the office except a piece of dirty paper on the floor. And a side of the paper says, "Protect this restricted data." But there's nothing on the paper. So we grabbed that sheet of paper and put it into the carriage of the typewriter, and run the sequential alphabet test. Okay? This guy from security comes down, and he goes Nova, bananas, whatever you want to call it, and, "There's an unclassified person looking at a classified document." I said, "Where is that?" He points at this piece of dirty paper with the alphabet typed on it. I said, "It's the alphabet." And he insisted it was classified. I told him, "You don't know what security is anything about." And that got me in trouble, but I don't care much about that. But he confiscated that sheet of paper with the alphabet typed on it. I contend that his behavior brings discredit on the idea of security. Bam. You get more scornful than ever. End of anecdote.

Miya: I recall some of the other people we had. We had Cliff Stoll. He was tracking his spy.

Michael: Talk louder, Gene.

Miya: We had Cliff Stoll. We were tracking Cliff's spy.

Michael: Yeah.

Miya: We had Russell Brand, who I got into role playing to try to at least get people to...

Michael: You know, let me tell you a story about Russell and I. I was consulting for Frank Bailey here at that point. And he was doing something with the Amdahl machine, which maintained all your data stores. He had a timesharing system on it. I told him, "That isn't a very good idea. You should separate those two things." "Nah, that's okay." So I said, "I'll make a deal. I'll try to break into it, and I'll tell you about it." So from my office over at Livermore, Russell sat down and he typed a few words in, and got into the Amdahl system over here at Ames. <laughs> And I took the data to Ron, and I said, "Look, we could do it. We're easily-- it was no effort whatsoever. Why-- don't you think you should move the timesharing system away from your data storage, which really are the family jewels." I don't know what he did about the whole thing, but he did say that our invasion was embarrassing. <laughs> Well, but you know, I think that comes more from being unfamiliar with security than it does with being concerned about security. You know? How do you be secure?

Miya: Well, yeah, that's some of the-- we'll probably have to come back a little bit to security.

Michael: I hope now.

Miya: I do, too. Well, we mentioned Russell as an example, and so, you know, in terms of going back to that era in time, it was-- I believe it was Marcy Smith who told me to invite you to give a talk, a seminar at Ames, and that was, I think the first time that I had met you. Although, that's roughly the same time that my friend Steve Skedzielewski also started to work for you. Now that's a name that the person transcribing it is going to have to figure out how to spell Skedzielewski, which I'll do, because I learned how to do it from Steve. And...

Michael: Great person.

Miya: And we haven't even mentioned him with regard to the history of the Computing Research Group. We'll get more into the CRG. One of the other things that was sort of big in the '80s, but people think of less, was one of the reasons why we invited you to come to Ames, which was the fact that at the time, my honorable ancestors, the Japanese, were starting to come on strong with regard to, first consumer electronics and consumer goods, with steel tankers and the like. And you, with four others, went from Livermore-Los Alamos, went to Japan.

Michael: Five others.

Miya: Five others. I thought you guys were five in total.

Michael: Norman Morris.

Miya: Norm Morris, you, Jack.

Michael: Jack Worlton, Bill Busby from Los Alamos. John Raneletti, Harry Nelson and I from Livermore. And we visited three computer companies, and three user companies in Japan. And we learned about the plans for their-- they had a ten-year project to build a fast computer, which is very sensible. They were talking about a billion operations per second as I recall. Now we go past that now, but not at the time. And they had some really neat ideas about using the components at lower temperatures. HEMT, high electron mobility transistor, and the packaging was exquisite, And we had...

Miya: Let's say a science paper was produced. You were one of the authors. I think Busby was one of the authors.

Michael: The three guys from Los Alamos rushed home and wrote this thing, and never even bothered to consult with us. I had invited them to come along, because we were going to go to Japan, but again, I believed that we'd get much further ahead if we'd cooperate. And it was typical Los Alamos-- well, I don't know what to call the other...

Miya: Canter [ph?].

Michael: Well, they're very public relations-minded, okay? They ran ahead and published this thing. Nothing wrong with the publishing of it, but since there were six of us, I don't see why we weren't included in the author list. They correctly got it. You know, they said in a footnote that I had organized the thing and invited them to come along. Just seems a little bit cheeky.

Miya: But you forgave them.

Michael: I don't care about it. I mean, we didn't do it in order to write a paper. We did it in order to find out what was coming from Japan.

Miya: What was the climate of that era like?

Michael: What do you mean?

Miya: Well, I heard my honorable ancestors characterized as being ten-feet tall, and stuff like that. I mean, people of this country were worried about the American electronics industry.

Michael: Look, the relationship between businessmen and technology over in Japan seems to be cleaner than the relationship here, okay? Here, the technology doesn't have an equal footing. I'm not sure it does in Japan, but it seems more that way. I mean, we have-- I've heard it also said that the big problem is that our government structure doesn't know how to plan, okay? They can only deal with tomorrow. They can deal with a year from now or ten years from now. Whereas the Japanese are very good at that. And they respect each other more than I think is done here. And as things have turned out, you know, they've survived, whereas the people here haven't. That's another, you know, facts are okay. They're no match for belief, though. <laughs>

Miya: So say some words about your trip to Japan. Like where did you go?

Michael: I said we visited three.

Miya: Name the companies.

Michael: Hitachi, Nippon Electric and Fujitsu. I can't remember the names of the big companies that we've dealt with, but I think one was Toshiba.

Miya: Nikon was one, I think? For cameras.

Michael: No, that was just me. That's another story all the way. I wanted to meet the guy who took the specifications I gave, and built a lens, which we used on our graphics equipment-- it's a superb lens. He's almost blind. His name is Mr. Oikimoto [ph?], and he was a student of the physics department at the University of Tokyo, and he built that lens. I had to write a special specification to buy two of them, okay? Then the AEC turned around and bought 600 of them from-- and they used them in Nevada. And it was a very good lens. And it, by itself, was the most influential component in making photography of a CRT really useful. And in Nevada, it allowed them to photograph events that were not visible in ordinary lenses. A few ordinary lenses. So it was a great thing. And I went to one of the-- I wanted to meet Mr. Oikimoto, and I did. We had a nice time. He doesn't speak English, I don't speak Japanese. But we had an interpreter there. And we went to tempura bar in the middle of Tokyo, I got to see that they're going to have shrimp. Now he, with the cook, reaches into this bucket and pulls out a live shrimp, kills it, and cooks it. Good.

Miya: I wonder if I ate at that one. I'm wondering if I ate at that restaurant actually.

Michael: Well, all the tempura bars are that way, from what I was told. And then after you're through, you go to another room and drink green tea. I enjoyed very much that trip to Japan. One of the things that was an awful lot of fun was accompanying Harry Nelson, who's like a father-figure to the Japanese

amateur gamesmen over there, okay? And we went to a special meeting that they held in his honor. And had a big banquet, and it was great! You know, they'd be showing all these new games that these guys are developing. Fantastic stuff! They're really very good. So but that was extracurricular, you might say. We did that on a Sunday. During the week, we go to Nippon, Kogaku [ph?], that's-- no, not that one, no.

Miya: NEC?

Michael: NEC. Nippon Electric Corporation, yeah. And we were looking there at the unmarried mothers [?] of the FX...

Miya: The FX-2, yeah, I ran on it.

Michael: <laughs> Very impressive machine, except that I don't like Honeywell, and that's really what that thing looked like.

Miya: Yeah.

Michael: Then we went to Hitachi, and that was like an IBM clone. Then we went to Fujitsu, and it was a little better than average. And also the most-- that's the largest selling computer in Japan from what I hear. And they showed us what they were doing. And both at the component level. I didn't think too much of their software level, but it's a way of working. Which, again, I think it comes mostly from inexperience with dealing with large problems. They, for instance, had a special machine where you would try out your software before you'd put it on the big machine. Like the big machine is sacred or something. Nonsense! Who cares if a waste a little bit of machine time? I don't want to waste people time. That's the valuable commodity. So, and also their-- the environment in which you would check out your programs was very primitive then. And you know, but it maybe suited them okay. I think they had some interesting problems just getting the various alphabets to be representable on a printer. Hiragana, Katakana, one other one, too.

Miya: Kanji.

Michael: Yeah. And I got to meet Mr. Eiichi Goto also. He was the guy who invented the Parametron, as I recall. Very nice person.

Miya: So because of Fujitsu and our relationship, we should note that-- Ken Mura, I bet is the person who hosted you at Fujitsu. You want to say something about Ken?

Michael: I think he's a remarkable young man. He is a University of Illinois graduate, he has his PhD from there. He's a mathematician. A mathematical physicist, if you will. And his job in a sense could've been very frustrating. He was trying to sell Fujitsu machines over here. And it wasn't too well received by our government.

Miya: You want to explain why? From that period of time. That's 20 years ago. People watching us in the future, they're not going to understand that period of time.

Michael: Well, I don't understand it yet. I don't see how I can say anything about it. The point is that there was this "Buy America" sentiment. And for any machine that the government was going to pay money for, it had to be American-made. So even though when we came back from our Japanese trip, we had explicit permission from the Department of Energy, or AEC, whatever it was then, that we did not have to worry about where the machine came from. If it was the fastest that we could get, get it. Then they pulled their horns in on that. MIT tried to buy a Fujitsu machine.

Miya: I think it was an NEC machine.

Michael: Was it NEC?

Miya: I think so, yeah.

Michael: Yeah, I guess you're right. Yes, that's correct. And they were told, "Well, all the other funds you guys are getting will disappear." <laughs> and Bill Busby, who had by that time moved over to being the Director of the Computation-- the science laboratory department at NCAR [National Center for Atmospheric Research], tried to buy a Japanese machine, and was stopped also. So the only Japanese machines that we were then aware of, was down at Houston...

Miya: HARC. Houston Area Research Center.

Michael: Yeah, Houston Area Research Center, yeah.

Miya: Yeah, that was the FX-2, and then there was a VP-200 over here in Sunnyvale, Fujitsu at Amdahl, I used that one.

Michael: Yeah, well...

Miya: You guys probably used that one, too.

Michael: I don't remember. I wasn't involved in it maybe, I don't know. I don't remember it in any case.

Miya: Yeah, software was always a problem. We had to learn whatever operating system and tools they had to have.

Michael: Well, I think that the software problem was workable, in the sense that you, you know, by proper training and proper environment, I think could've been handled okay. There's something about-- because of the crowdedness perhaps over in Japan, there's something about teamwork taking precedence over individual stuff. But it may be that you need the individual to do the creation of a new idea. You don't get that out of a team, typically, so the style of software development out of-- maybe have-- needed to be fundamentally changed. And I don't know if they were ever going to get to do that.

Miya: But it wasn't working at Livermore, as well, too? Or was there always a lead, senior head physicist or something?

Michael: I think there was never any of this teamwork, but the idea of stuff, that was individuals. LNTSS was the final crowning point of a timesharing system development at the lab. It was two people. Not a team, <laughs> even Watson, you know.

Miya: Janitor. And said...

Michael: And he can't get done. Well, naturally! <laughs> Why it's so difficult for Russian to absorb is something I don't understand. One of those things.

Miya: So in the end, Japan didn't turn out to be a threat, tight?

Michael: Well, I don't know what you mean by threat.

Miya: Well, people were worried economically, that was the "Buy American" thing. But at the same time, you pointed out that when Seymour [Cray] got the Cray-3, the quality of the memory of the Cray-3, the Toshiba, the memory was better than the way the American memory...

Michael: Yeah.

Miya: So you want to say something about that?

Michael: Well, there's not much to say, you know. He couldn't keep the Cray-3 running at NCAR until he swapped memories. He got rid of the American-made memory, and he put in Japanese made memory, and I stopped having errors. Seymour's problem in practically every machine he ever built was memory. I've heard him called the Evel Knievel of computer design. Because he's always pushing way up on the power curve compared to where most designers go. And I know we had trouble on the 6600, getting memory errors all the time. Chuck Leith solved that problem in a sense by just running everything twice. At that the machine was dramatically faster than anything else that we had. So yeah, we give away a factor of two, but we are making great progress in running these large models. So I don't know that the Japanese threat has evaporated yet. Today Japan is in the same situation that we were. They're now frightened of the Chinese. To me, that's businessmen. And they are the crabgrass in the lawn of life. That's Linus Van Pelt that said that.

Miya: Ah, Linus Van Pelt of the...

Michael: Charlie Brown, Peanuts, you know?

Miya: Right, right, the great philosopher.

Michael: Yeah, he's good.

Miya: Next time we meet I want to talk in detail more about storage. But tell me more about system balance.

Michael: It was an idea in a sense that originated, I think, with Chuck Leith. And it was at the micro level of balance, you know. Chuck is very good, and he noticed that say when you're running the multiplier, nothing else can run in the arithmetic unit. So in order to feed that multiplier at its maximum speed, where are we going to get the operands? So in the case of the LARC [Livermore Advanced Research Computer], it came off a drum. And it turned out that, if I can remember it correctly, it takes a certain amount of time to get the operands off the brands and fed to the multiplier. By the time it gets there, the multiplier is just finished with a previous multiply. So that was nicely balanced as you say. If anything happens untoward, the balance disappears. Now you can take that to another level, which is what Chuck talked about. You balance I/O times against the operator times, adding, multiplying. Nobody paid much attention to the divide, because it's a slow thing, and it's never going to get fast. And it's done very infrequently. In fact, Seymour built a reciprocator, rather than do multiply. So, at the upper level, though, you could say, "Well, I have N words of memory. How many words of disk do I need to feed that?" Well, it's clearly going to be a function of time. And that kind of balance feeds into the operation of the arithmetic units as well. So by the time you're all finished, you're looking at what we'll call a modulation transfer function. Of all the components that support your calculation, and you find out how to tune those so that you have enough memory, you have enough disk, you know how fast the multiplier is running, you know how fast it takes to transfer, what is the bandwidth to get stuff out of memory, and up to an arithmetic unit and things of this sort. All that figures in. And there were simple little rules like you need to deliver a word from I/O within a multiplied time, or a tenth of a multiplied time. Whatever it might have been. I don't remember the numbers anymore. I'm lucky I can remember where I live.

Miya: Where was that? Oh, a pinched little town? Remember you said that last time?

Michael: What now?

Miya: You called Livermore a pinched little town.

Michael: Oh, yeah.

Miya: And it's actually transcribed that way in the proceedings.

Michael: Well.

Miya: Now did balance come before you did the little triangle for Salishan? Algorithms, architectures, language.

Michael: No, the triangle was developed for the Salishan.

Miya: Yes, but you'd thought of balance before 1980.

Michael: Oh, yeah. I had, you might say, a tremendous advantage. I could talk with Chuck Leith, Norman Hardy, and my officemate Dana Warren. Those are the three guys that had an enormous amount of influence on me. Made me think. It was such an exciting thing. You know, so we talked about this kind of balance. Remember, we were in a place where the morality was in chiseled microseconds. Want to go as fast as we can. We want to go as fast as we can. That means find a way to chisel the microseconds down. And so there were all these little rules, you know? You could sort of say, "I just

computed that average out, and I want to store it." No, I don't have to store it. I'm going to leave it there, because in three instructions I'm going to need it again. That's just a trivial little thing, but it was what people thought. So, Gene, Norman and Chuck, and I, we had an enormous amount of dialogue about performance. About complexity. And balance, you want to call it. Those were good days, I'll tell you. We used to run home, have supper with the children, get them to bed, then run back to the lab. And work there till about two or three in the morning, and we'd go out to a place called the Hungry Truck out on Highway 50, and have some breakfast. Norman's breakfast always consisted of orange juice. No, there was two breakfast. Orange juice, French toast, and coffee. Ice cream, too. Bill Lindley always liked to have cottage cheese with mustard. Revolting. <laughs> Chuck ate, I think a waffle.

Miya: Is there anything else you want to consider?

Michael: I don't think of anything, but I'm totally discombobulated.

Miya: Tell me about Sid Fernbach. We'll ask more about Sid Fernbach in the next time.

Michael: Sid grew up in Philadelphia. When the war came, the Second War, he was ready for graduate school, but instead was assigned to Japanese language study in Norman, Oklahoma. The war ended, he decided to pursue his PhD studies in physics. He went to Berkeley. He was an Oppenheimer student till Oppenheimer left and went to head up the Institute for Advanced Study. At that point, Sid was transferred over to the Institute. He finished his PhD, and embarked on a post-doctoral year with the Stanford Linear Accelerator. At that point, Edward Teller called him and said, "We're getting this thing farmed out in Livermore." It's going to be another laboratory, part of the university, and we want you to come to work." So Sid was seduced, I guess. And he did. And when he got there, Edward told him, "I want you to be in charge of the computer." "What's a computer?" <laughs> So it turns out that couple of people-- I'm not sure exactly how accurate this is-- but I think it was Herb York-- not Herb York,-- but Bob Jastrow and one other person who had needed a computer to run some calculations. And they went to New York City, and they came upon the decided that's what's needed at the lab. So they had ordered that. And that was when Sid was going to take charge of. So he did.

Miya: The Univac was at NYU?

Michael: Yes.

Miya: At the Courant Institute?

Michael: Yeah. He did that. And he was the director of the computation department until quite late, in the 1980s. In that time he got to be known as a world leading computer scientist. There're a lot of people-- a prophet is not honored in own country. So a lot of people at the lab thought that was just baloney. They said, "You give me that kind of budget and watch what I can do." Now Sid would-- Sid's theme song was "That isn't fast enough." And I think he was getting that from people who're trying to run these big problems, and he was very good at going to Washington to get money, and to convince the people who made "decisions" that a bigger machine was needed, and then he was fairly hospitable to all the salesmen, that sort of person who would come to Livermore to tell us about their machine. Okay? And they would always leave a programmer's reference manual, hardware reference manual, and people like Chuck Leith or Norman Hardy, you know, Harry Nelson, would go through those manuals and learn

what the machine was all about. And I think it was safe to say that we knew more than the salesmen did. That's okay. But it did lead to getting machines that we're not going to buy-- they got to be refined, okay? And Sid made a lot of enemies, and I think his downfall was getting two [CDC] Stars. 'Cause at that point he was removed as head of the computation department, and he was attached to some AD staff, and he stood that for a while, and then he resigned.

END OF SESSION 2

START OF SESSION 3, May 9, 2006

Miya: Time stamp is May 9, 2006, tape [interview] number 3 with George Michael. We skipped a month so this'll be the third of perhaps four tapes [interviews]. Dag wanted me to ask you a question relating back to working at Livermore and life in Livermore. He wanted to know why or why not-- Did you have a bomb shelter from that era in the '50s and '60s in your backyard. If so, why or why not?

Michael: No, I don't have—

Miya: You didn't have a bomb shelter. Why not?

Michael: There was a big one built to the north of town but I didn't belong to it.

Miya: Did you feel that it was useless to have a bomb shelter or—

Michael: It was a vogue-y thing and I must admit I lost some sleep trying to figure out how to do it but we never did anything. We looked into questions of buying boxcars and burying them in the dirt and equipping them with all kinds of stuff for survival but then somebody said to me the living will envy the dead so that was enough to turn me off so to say and I decided there was no likelihood that we were going to do this kind of stupidity so I just didn't do anything.

Miya: Today we'll concentrate principally on two sections that you and I have agreed on e-mail and that's a list of the computers that you used in your career on some of which you've said a little bit already like the Univac; and then the second section will be a list of people that you and I have come up with and we'll probably add to over time. I don't think we're going to be able to get to that entire list but let me iterate first the computers that you and I have agreed on. We start with UNIVAC I. We'll do some redundant stuff and then the IBM 701, the IBM 650, the IBM 704, the IBM 709, the PDP-1, the PDP-5, the PDP-6, the PDP-7, the PDP-9, the PDP-11, the PDP-15, the IBM 7090, the LARC and Chuck Leith, L-e-i-t-h, the Stretch IBM 7030, the IBM 7094, the CDC 6600, 7600, with its operating systems and software, the 8600 which didn't go anyplace, the Octopus system, the Radiation printer, STAR-100, Cray-1 and the Illiac IV stuff, Cray-2, Cray-4, Cray-3, the SRC-6 and then we'll get to people. Okay?

Michael: Whatever.

Miya: One of these times you should do a joke. You want to tell a joke?

Michael: You missed one.

Miya: What did I miss?

Michael: The 6800.

Miya: I did have that in here I thought.

Michael: Did you list it?

Miya: I'll put the 6800 in here.

Michael: Between the 66 and the 76.

Miya: I will put the 6800—

Michael: But it just keeps it off[ph?] of the general rule. Seymour had a stillborn computer between each computer that was commercially delivered. Between the 6600 and 7600 he had a 6800. Between the 7600 and the Cray-1 he had an 8600. Okay. Maybe even more than one but at least that much and in all cases the thing that he was looking into was how to make the componentry of that period do what he wanted to do and he did a fine job I think. I think he was a genius and an interesting guy and fun to talk with and certainly a good person to learn from. I think we lost a tremendous amount when he died.

Miya: Do you want to save it or do you want to just jump into the Control Data machines?

Michael: No, I don't want to do anything except answer your questions.

Miya: Let's start with the computers. Okay?

Michael: All right.

Miya: The first computer you've actually said a little bit about and the thing is you don't want to say anything more—

Michael: Say what?

Miya: The first computer you want to say something about-- You've already said a few words of the Univac I. Is there anything you want to add to that?

Michael: Yeah.

Miya: What additional things do you want to say about the Univac I?

Michael: Well, Univac I gave us a jump start and I would have to say in somewhat the wrong direction because since it was decimal-based, everybody thought that was natural and so we had a tough time in

some intellectual sense leaving the decimal notation and using binary. Everybody in the physics department thought that 0.1 was a terminating decimal machine- decimal number but in binary it isn't. That was a problem. The other thing that I- the Univac conditioned us on was that if the machine wasn't reliable, the system [still] could be so we were accustomed to trusting the Univac and its reliability once we learned how to do it; whereas when we got the IBM stuff, the principal word was 'unreliable.' It would make mistakes all the time and we had a tough time correcting those and it- I suspect it colored the opinion of a lot of people that didn't know how to handle binary unreliable, unchecked machines but all that went down after a while. And well, the Univac taught one other thing I suspect and that is that you could put things together that were the products of different persons so that you could build a complex structure out of simple components—

Miya: Systems.

Michael: System, yeah. We didn't have an operating system on that thing. The operating system really was the operator sitting at the console of the machine doing what he was asked to do by the guy who's running a program, so it was customary with the physics people when their time came to be in and sitting next to the operator saying what to do if that was necessary and that was a very, very pleasant style of operation. When the other machines came in, the practice of sitting at the console of a computer just disappeared and I think rightfully but while we had it it was fun. Maybe even one other thought is—

Miya: On the Univac.

Michael: On the Univac. Some creative thought was absolutely necessary for output, how to get output in a time of interest, and that wasn't- we didn't solve[ph?] it on the Univac. We tried several things that have been reported on. First of all, there was no high-speed printer. We had typewriters that had been modified. Secondly, there was no graphic output. We invented that- or well, not invented but we developed it as an emergency set of measures and it didn't work and for me I think the major general lesson in all these attempts to do this, that or another thing was simple is good, complicated is a loser, cleverness gets you nowhere, it's the simple things that are effective and work quickest and most trustworthily, period.

Miya: You say graphics you did as an emergency measure.

Michael: I'm sorry.

Miya: You said graphics was an emergency measure and that you did not solve—

Michael: I didn't mean an emergency in any sense except since there wasn't any whatever we did had to be, so to speak, constructed from the beginning—

Miya: But the reason for asking this question is you said you didn't solve the output problem on the Univac. The question is which system would you regard—

Michael: Your question is what?

Miya: What system did you regard solving the output problem on so when we get to it I can—

Michael: Look. As I've reported in a couple of papers that we put out, we got a machine called the Benson-Lehner flatbed plotter, okay. It was fixed up so that it would adapt or could be attached to IBM equipment. We didn't have any IBM equipment, so our engineers had to build conversion devices that went from Univac output formats to IBM output formats. That means to go from the magnetic tape that we had on the Univac to punched cards for the IBM machines and in the process of doing that, so much time had elapsed that you lost interest in the result. It took too long to get it and it didn't work very well since everything was analog and we couldn't trust any zeroing of voltages or repositioning of cross hairs so that you always got back to the origin of a piece of graph paper or something like that and the plots weren't very good.

Miya: What was the plotter connected to, which IBM—

Michael: The plotter sat by itself, okay, and we would bring Univac output tapes to a unit that had been built by our engineers and it would read the tape and punched cards, okay, and we would take the cards and put them into this IBM machine, I think it was a summary punch but I don't remember, and it would read the cards and move the cross hair- move the plotting device accordingly and put down a mark where it said to put it down. And as a test that it was okay, after you were finished plotting the curves and stuff like that, it'd say return to zero and it never would so that you couldn't trust the curves to tell you anything and we essentially for the first year or so abandoned that approach and did all plotting by hand and it was more effective. It worked fine. On the 701, the operable word as far as I can remember was 'unreliability.' That was our first real introduction to unreliability. There were many other features. Whereas the Univac was a serial machine, the 701 and all of the 7000 series products from IBM were parallel.

Miya: Word parallel.

Michael: Word parallel, yes, and that of course gave a great speedup and so in many respects people put up with the unreliability because it was so fast, okay? You could, in principle, run it several times to cover over the unreliability stuff, though one time I ran a problem 15 times and never got the same answer--

Miya: With the exact same initial starting—

Michael: The exact same. It was just one cycle of a calculation and it never came up with the same answer. There were many clever IBM customer engineers and stuff like that and one of them helped me or he helped IBM. We found some microphonic tubes, meaning that they were seismically sensitive and shouldn't have been. The memory organ on the 701 was the so called Williams tube and IBM built a factory to make those tubes. They were in their own little way a source of unreliability, but one of the principal things that caused them to be unreliable is that they had to be kept in a constant light environment, in the dark would probably be the best thing. Otherwise, whenever someone would come into the room from the outside, come in through an open door, some light would get in and change the memory. It took a while to understand that that was happening but eventually it was understood. The other thing is that we were in a transition period between the absolute decimal notation on the Univac and floating point notation that showed up on the 704. So we had problems with floating point representation with mantissa and characteristics and things like that. There was a board that was wired. It was called the dual—

Miya: Dual, d-u-a-l, right?

Michael: D-u-a-l, yeah, and it had what I don't remember now though it had something to do with representation of numbers and how they were handling the number screening [ph?]. The range that we had to deal with in the nuclear calculations was probably 30 orders of magnitude, maybe more, but as a result we couldn't keep significance at both extremes and it caused problems mostly of the intellectual sort. You couldn't understand how can I represent Planck's constant and Avogadro's numbers in the same scale and have significance when I was using them in calculations. That didn't work out but the 701 principal problem to me was its unreliability.

Miya: You said 30 orders of magnitude. That's a decimal, not binary.

Michael: Yeah. Well, and what is Planck's constant? Ten to the minus 26 or something like that? What's Avogadro's number?

Miya: Ten to the plus—

Michael: Ten to the 26, right? So this is- that's the spread of things that we had to deal with. That's much more than 30 orders of magnitude, but you could slough off some of these things by changing the scale of representation and the idea is that you had to have some significance to the calculation. I go back and say when we started with the Univac that nobody trusted the machine and so, for at least my calculation, every number that the Univac was going to produce in a cycle of calculation was reproduced by hand and that means they were doing it for 12 decimal digits, okay, and where there was a disagreement people thought it was the Univac's fault whereas it was almost always a human error that had been made but it was the exact opposite with the 701. People started to trust the machine and it never gave you good answers. It was an unreliable machine. It was faster than heck and as some of the physicists that were trying to use it would note that it's fast and wrong. Actually, after they got rid of the Williams tube memory systems and began using core memory, the 701's reliability went way, way up, and it was in an acceptable range but by that time we had shall we say iterated to the next IBM machine which was for us the 704. The 704 appeared in 1956, the first one, and there were some new ideas there. The first one was floating point arithmetic and that very quickly got accepted, people liked it, and the other thing was that the idea of index boxes got introduced and after you understood what that was going to do for you, it was an incredibly good idea. Now in looking at the history of those things, I believe they were both contributions that started off in Great Britain, but at the time it was supposed to be a gift from IBM.

Miya: Those are on the 704. Right?

Michael: Yeah.

Miya: Let's stay on the 701 first because there's a couple of logical questions about the 701. We'll get to the 704 in a moment. On this issue of word parallel on the IBM machines and word serial on the Univac machines that you guys had come off of, did you actually program the arithmetic carries serially on the Univac?

Michael: No, it was taken care of via hardware. You could say 'add two registers' and what it would do is it digit-by-digit would add the things together, but we didn't see that. We just saw the result.

Miya: On the Univac but IBM did it--

Michael: On the IBM machine it added them and we didn't see anything there except the result either but it went 30 times faster or something like that.

Miya: How many Univacs did Livermore have? One?

Michael: Say what now?

Miya: How many Univac I's did—

Michael: We had one Univac I.

Miya: You never got—

Michael: I don't think we ever got any more than that—

Miya: You had one 701?

Michael: I think we had two of those.

Miya: Two IBM 701s?

Michael: I think so but that's very fuzzy.

Miya: What software did you use to program in, programming languages or what not—

Michael: It was all machine language, if you will, and the concept of an assembler hadn't really percolated down to us yet. People were writing in absolute octal notation in some cases and it seemed sort of natural. I remember we had a thing called New York Assembly Program 1, NYAP 1, but I can't remember if that was introduced while we had the 701 or the 704 but it was the first assembler kind of vehicle that we had and I must say it wasn't universally accepted, okay? Some people refused to touch it and I can't explain why. I don't know. After some users got their hands on this whole set of procedures, a thing called the Symbolic Assembly Program, SAP, was developed by the users, by IBM users, and that was rapidly accepted and there was a thing called- this is just post 1956, the thing called the FAP or Fortran Assembly Program, and some of the more, quote, advanced programmers used FAP in conjunction with Fortran to produce complicated things that ran very fast, okay, as fast as a machine could do. And FAP was produced by a guy by the name of Dave Ferguson who was programmer somewhere down in the Los Angeles area. I met him once. It was a very nice job and he had introduced the ideas of macros and it was interesting to manage the index boxes with them and FAP was a very good program and had a lot of ardent users at the lab. I don't know how long it persisted. That is, once we got off the 704s and were onto the other- next machine, did anybody use it? I don't remember. By that time Fortran was taking hold and then along the way we had a thing that predated Fortran which was called Compiler and that was under the direction of Kent Ellsworth, a guy who worked at the lab, and he

had a team of people that were perusing it. I won't go into the reasons, but they made a couple of really bad decisions on the input and it was extremely difficult to get something to work, but once it worked it was fine. We had an even earlier attempt which was called LMO after Merritt Elmore, a programmer at the lab who wrote essentially what the format function was on the- in Fortran, okay, so you could design a printout page and so it had more than just the numbers on it and that caused a little bit of trouble because security stepped in and said the numbers alone were okay because they didn't have to relate to anything but now with letters there, words there, you knew what those numbers were supposed to mean and that was classified. So we suddenly had classified printouts. That didn't stop their use but it did constrain it some.

Miya: Does that mean that the output printers had to have the classification keywords on the top and the bottom and stuff like secret, top secret—

Michael: That took many years to develop.

Miya: That was later.

Michael: Later, yeah. At first it was just that they said it was classified and in some cases forced the programs to remove the alphanumerics so that the numbers were just there, okay. So some people would print a blank page with just the alphanumerics on it and leave it sort of transparent and then just drop it on top of a printout and you could tell what each column meant and so forth. There were so many expressions of individualism on the early computers it was just boggling, I suppose. Everybody would have a creative idea, a tricky, little new thing, and it was this- the diffusion of all those things among the user community that led to the system and to development of operating systems at the lab, things like that. It was an interesting-- It was an exciting period of time, just a lot of fun, and I never knew what was going to happen one moment to the next, but we went through it fairly rapidly and whereas I think we kept 701s around for probably three or four years, okay. The 704 was introduced in 19—

Miya: 56.

Michael: --56 and it rapidly became our favorite machine. It didn't have the beautiful internal checking that the Univac had but it was a reliable machine notwithstanding and it can- and of course the difference was its use of ferrite cores for memory. That really made a big difference.

Miya: Wait a minute. I want to finish on the 701 and the Univac I. Can you state anything about either the Univac I or the IBM 701 peripherals or the memory hierarchy, sizes or any special devices like drums or printers? For instance, most IBM people have never heard of a drum.

Michael: Never heard of what?

Miya: A Univac drum. They've only heard of disks as an example so talk about the peripherals for both the Univac I and the IBM 701.

Michael: Okay. The Univac I had a- what we called a thousand word memory and each word had 12 decimal digits. Each decimal digit consisted of seven binary digits. Four would have been adequate. Two were added to provide alphanumeric capability and one was for parity, these bits. There were additionally

pieces of memory called tanks. There was a 60-word memory for input, output so there was a tank that held 60 words coming into the machine and another tank that held 60 words going out of the machine and the beauty of those was that you could use them in computation as regular memory. There was also another set of registers, not 60 but just two, and another one that was just one. So we had one, two, 10 and 60 and then the thousand word memory and programmers learned how to use those cleverly. They learned also since the memory on the Univac was an acoustic delay line you would- you can imagine that you're saving a word that's in the delay line and it moves through the delay line and is brought out, amplified and inserted at the start of the delay line again, and one knew that timing quite well so you could write your program so that just when you needed something it was coming out of a delay line. The delay lines were acoustic. They were essentially a mercury tank with a piezoelectric crystal on each end, which would do whatever it did, squeak in concert with the composition of the number you were putting in and there were, as I said, total of the major memory of a thousand.

In the case of the 701, it had 4,000 words of memory. Each memory was 36 bits and the organ was the Williams tube which was to store discrete charge points on the screen of a cathode ray tube, and then by sampling it, could find where the spot was and therefore whether it was on or off and which word it was in and so forth. It turned out to be that that was sensitive to light and so that'd keep them protected, otherwise they changed their information, and that's all of the general memory that was on the 701. It was the 4KB of 36 bit words and the 36-bit word size I believe came from the attempts on the SAGE system which was being developed somewhat concurrently at Lincoln Lab and had some very creative people on it. We went through the 701 before the 9- the 90 and the 94 all using 36-bit words, okay, so over that course of time, people learned how the floating point representation could preserve their numbers some significance but I believe some of the analyses, which I don't believe ever was published though, showed that after a normal computation if you had one digit of significance left you were lucky.

Miya: The most significant digit, the leftmost digit that can—

Michael: Yeah, one digit of significance—

Miya: You didn't trust all of the digits to the right nearly as much.

Michael: Well, it was just noise as far as the true condition of physics was involved. However, it was I'd say a stylized kind of noise, okay. And a point I wanted to make though was that see, the 36-bit word was divided up so that there were 27 bits of fraction and the other part was for the exponent and sign, it was biased so that you could show pluses and minuses, okay, and that's swell. They got a different kind of representation by requiring that the characteristic was always normalized, that is to say there was always a bit in the highest, most significant position and—

Miya: Anyone who listens to this tape is going to have to know what normalized numbers are as well as the significance of the word 'significance.'

Michael: Well, the major consequence of this lengthy experience with 36 bits was that whenever we got a chance to test anything else, we did. In the case of Stretch and the Star, both of which had 64-bit capability and 32-bit capability, we found that 32 bits was not enough to preserve what do I want to- coherence in the calculation, okay. We found that 64 bits was more than enough and it was always an argument as to where in the middle was the dividing line, okay. Now it's interesting to me because Seymour brought out his machines with 48 bits, okay, and that's interestingly halfway between 36 and 64,

and I don't want to imply that that solved the problem but we would run in- on machines that could do it in a so called noisy mode, not noisy mode but reduced significance, 32 bits. It was so called fleddy[ph?] faster but it didn't give you the right answer and if you could- but you could run a lot of the calculation where it didn't really make much difference and the important part of the calculation would be done in a 64-bit precision and that worked out and in its own little way meant that you would get- you would just ignore the 36-bit machine, okay, for- from the point of view of arithmetic. So have we finished the 701 yet—

Miya: Not quite. We have to take a break here because he needs to change—

Michael: I can't hear.

Miya: We need to take a break because he needs--

END OF TAPE 3.1

START OF TAPE 3.2

Miya: Okay, so we're talking about--

Michael: We're talking about peripherals.

Miya: Peripherals for the Univac I and the IBM 701.

Michael: In the sense of peripherals on the Univac, there was this magnetic tape. It was a metal oxide tape on a metal substrate and they were quite heavy and when they were being used just as intermediate storage, they had a recording density of 200 bits per inch. When it was being used for output devices, like printers, the density was 20 characters or bits per inch. Now we had, I believe, ten tapes on essentially what we would call today two channels and they were a source of the most extensive unreliability that the Univac showed. They were not vacuum column. They were fixed up. The tensioning and stuff like that was fish line and springs and pulleys and all things considered they were unreliable but they worked, you know, when they did. In addition, just because it's possible, there was a console typewriter that in a sense was a peripheral. Now offline there was essentially a thing that held a tape which would be read and its output fed to a typewriter, a wide carriage typewriter with the capability of maybe three or four, maybe six characters a second. And so, whereas some of the calculations on the Univac would run for tens of hours so would the printouts run for tens of hours because people didn't acknowledge the fact that it took so long to print. They just printed what they wanted and since there was no real sophistication or anything like that if you wanted to print a number you got all 12 digits.

Miya: When you say a printout took tens of hours, is this simultaneous to the calculations or is this serial to the calculations after the fact?

Michael: It's independent of the calculation in a sense. Once your calculation has been done or is in the process, it has written a tape with a density of 20 pulses per inch. That tape is put on what was called a Uniprinter. The Uniprinter was a typewriter so the tape would be read and you'd get the typewriter typing at six characters a second. If you have a million characters to print, you can do the arithmetic, it takes a while. Now, as I said before, we attached a Benson-Lehner plotter as a peripheral but it was, I think, not successful and it did not receive any heavy use at all. It took too long to get from the calculation to a plotted piece of paper. You could do it faster by hand. Mylar-based or Mylar-coated on the outside and the beginnings of vacuum columns were in use so the tape was pulled through a vacuum column and I think everybody has seen those since then. Now, additional I/O devices, that was the traditional card reader and printer, which I think was a modified IBM 407 accounting machine. It printed the characters in each column up and down. Do what you want. That was 150 lines a minute at maximum and the card reader read about that same number of cards per minute and that's all there was in the case of the 701.

Miya: I was going to ask if the 704 printer was an IBM--

Michael: The 704 had those things, that is tapes, card reader, line printer and it had also, one of them had this thing called a CRT in two forms. The 704 had a machine called the 740, which was a five-inch CRT that was imaged by a camera and lens and it had also a 780, which was a direct view, modified television set. And that, as I recall, covered all that there was on a 704 and all of the 704. Only one 704 had this graphics device on it. It had a-- it was kind of like an afterthought that that was put on there. I'm sure that some people thought it was important. I do not know who installed the first one. I believe the machine itself was built by Stromberg Carlson and it did not have an accurate camera and accuracy here is that the frame-to-frame distance was more or less constant. So, this was a modified clock motor type motion picture camera and the Univac-- not Univac but the plotter on the 704 could do just a very few things. It could plot a point. It took 151 microseconds to plot the point. It could start at a point, draw either a horizontal, a vertical, or a diagonal line, 45-degree diagonal. And, one other thing, it had a command to say advance the film.

And it was out of that so-called primitive set of instructions that all the complexity that people built into their displays was constructed. Somebody invented a 5x7 series of characters so you could typeset on a piece of film. The film was atrocious. The lens was atrocious. The camera was atrocious and with the help of Dave Dickson [ph?], who was the head of tech photo at the lab, and his crew we put a pin registered camera on there so that the distance between frames was constant. That meant that you could make a motion sequence so it was terribly expensive to tie up the machine to make a movie. People didn't like that. What with that system we put on, you know, that better camera, we also put on a thing where you could put a film holder and a gadget to hold a photomultiplier behind it and that was the first of our say three eyeballs that we had developed. You could plot a point with the CRT and if the photomultiplier saw it, it meant that the film didn't have anything in the way at that point but if the light didn't get through it meant that the film was dark there and you could build up these mosaic responses and infer that this _____ was the letter A or the letter Q or whatever it might be or that we could draw a curve and trace the curve by searching for it so to say. So, as I recall Dale Marshall [ph?] was the guy who wrote some of the original eyeball software that was used on the 704. I think he called it Grope [ph?] and with it, it was a general utility. You could use it to probe the contents of a piece of film.

Now that's all of the peripherals that the 704 had and for that matter excepting the eyeball which we didn't put on any other of the IBM machines, all the 7000 series machines had the same set of peripherals, that is tape, card reader, printer. They had also, you know, which goes with the card reader, a punch so it would punch cards if you wanted. You know it's sort of like I can't think of a good analogy but when the

first automobiles came out they put buggy whip holders on the side, didn't have much to do with a car at all. Well, punching cards is in that same category of thing. Why would you want to punch a card with a fast machine? I supposed we could find good reasons but I can't think of any right now. One of the major operations on the 701, 704, 709, was to not waste time trying to print results on the machine, so what was done is that a tape was written and the tape was taken over to a special installation of IBM stuff which was essentially tape-to-paper as a conversion came. You'd have the tape-to-card or -paper, would read the tape and run a printer. The printer was one of the ones that I've described _____ 150 line a minute thing. It wasn't super speedy but at least it didn't tie up the machine.

Miya: 132 columns?

Michael: Yeah. Well there was always an argument about whether it really meant 132 or 150 or 120. It's just a matter of how you wanted to use it. It was a little bit later that sophistication caught up with the thing and said, look, we don't need to print the full decimal digit significance of 36 bits or 12 decimal digits, whatever you want. So, people were beginning to-- well what would we call it, produce edited output. You only need five decimal digits. It's almost-- that's unheard of, one part in 100,000 you know for this kind of a calculation, so they would truncate the rest of it and then pack the words together and so as a big computational procedure it didn't take much time, was easy to do and it got the most efficient use of storage and tape transporting and things like that. It was along about that time that we started using other computer manufacturer's stuff and one of the big things was I always wanted to be able to move a tape from one machine to another and we couldn't do that. There were errors all the time. So on looking at it, we had engineers from CDC, I think DEC and IBM and our own examining the question and the problem got solved by essentially making the inter-record gap on the tape one inch instead of three-quarters and then everything worked. So you could take a tape from any machine and put it on any other machine and it would work. I don't know if that procedure got past the lab boundaries. Are we the only ones that did that?

Miya: No.

Michael: I don't know.

Miya: I can tell you when I got on 360s that the inter-record gap was an inch.

Michael: Well, anyway when the 709 came I would characterize the 709 as a 32,000 word memory. It was really just a 704 with better componentry. We added as a peripheral our first DD-80. The DD-80 was and still is the fastest display device that could be found. It was custom-built for us by a place called Data Display, Incorporated which subsequently got absorbed by CDC. I'll go back and mention that the 151 microsecond per point plot on the 740 on IBM's machine got to be between six and nine microseconds per point on the DD-80. The accuracy was at least as good.

Miya: Wait a minute. I'm confused here, okay.

Michael: We can't have that.

Miya: I wrote down here you're trying to do the processing time for a point on a display. You're trying to characterize the processing time for a point on a graphics display.

Michael: Yeah.

Miya: So you're comparing a DD-80 to an IBM 740.

Michael: Yeah.

Miya: So, you put for the DD-80 151 microseconds to a point.

Michael: No, no, no. For the IBM machine it's 151 microseconds.

Miya: Okay. And then it's six to nine microseconds on the DD-80.

Michael: Yes. And then in the _____ we had a PDP-1 delivered to us and the plotting time on that machine was 50 microseconds.

Miya: Fifty microseconds.

Michael: So, we have three numbers to compare point-plotting time.

Miya: Right.

Michael: But that's where the similarities go even-- that's where they end. The DD-80 had the ability to draw a line from any point to any other point. It was a maximum time of 36 microseconds. It could be as short as six. It depended on where the line was and so forth. The DD-80 also had characters that it could plot out of a character generator where the time was nine microseconds whereas the characters that were formed in a 5x7 matrix would take an amount of time, the number of points times 151. That's too long. And the DD-80 had one other thing. The camera that we put on it was originally designed in its first version to be a camera for a spark chamber at Berkeley. That was built by film effects. The guy who did it was Bob Walts [ph?] and it was a pin registered camera, what say with claw pull down so you used perforated film. And in a free run mode it ran 30 frames a second as opposed to a half a frame a second on the IBM equipment. So it was a dramatic improvement and I must say that the staff at the lab made very, very heavy use of that thing. Miles and miles and miles of film got pulled through that thing. And the film itself was a success story. With the help of some engineers from Kodak we developed and tested and worked out the procedure with films that were designed to match the phosphor and the light output and the processing techniques and so forth that were on the DD-80. This film was called RAR 2498 as we used it. It was a Mylar-based film, monochrome and it was very, very good film, tough, processed well and it was very fast.

Miya: Okay, I want to wrap up.

Michael: _____<Inaudible>.

Miya: Well, the thing is we're digressing to successors of the 704 and I want to ask one series of history questions. I want to ask you one series of history questions about the nature of the applications that are

being run, raised by a comment you made that it took tens of hours to do calculations and tens of hours [for printing]. Hold on a second. I'll tell you why because I want to confirm with him before I ask you these questions. You say the calculations took tens of hours and the printing took tens of hours. So, I want to give him the option to stop the camera while I confer this to you on this. Okay, so can you suspend?

Okay, I'd like to ask you basically two questions having to do that were raised by your comments here having to do with the length of the running time for a calculation. You said tens of hours as well as tens of hours to print a result.

Michael: Yeah.

Miya: And then one of your favorite topics and mine having to do with security, you had mentioned that output printed by the printers created a security problem and you had to identify it.

Michael: Yeah.

Miya: So, first can you say something about the nature of the calculations in terms of like were they 1-D, 2-D? Were they, you know, how-- if you're dealing with kinds of algorithms or mesh sizes, dimensionality in words or bytes as an example?

Michael: Yeah. I could do it by examples I suppose. Just generally what we're doing is the equations of mathematical physics. To characterize them as anything, they're partial differential equations. Calculus has not provided us with any real solutions for interesting problems. The only thing you can do is approximate. If you use a very large computational area the errors are terrible, so if you divide that area into much smaller things and solve the equations for very local points then put the points together by virtue of smoothing or what you know about a physical process, you're going to approach the actual solution to the problem as close as you're going to be able to. But if you divide the problem up as was suggested, each one takes time. If you have 1,000 points in an area it will take 1,000 times whatever one point takes. If you have a million points it takes 1,000 times again more computational time so that the time for running a problem depends on the degree of precision you're trying to represent in the solution. You assume the solution is valid at a point and at another point, which is nearby and then you assume that you can connect these two by an approximation technique, curve fitting if you will. That being the case, on the Univac when one is representing the dynamical conditions at a point, what does one need? One needs to have the velocity. One needs to have possibly the second derivative acceleration. One needs pressure, temperature. You probably need to know what is this point made of, iron or Jello, stuff like that.

Miya: Chemistry.

Michael: So one needs equations of state to describe or connect these various points I just mentioned and one needs that for every point in the so-called mesh. The mesh is made again because you can't solve the differential equations with the whole thing, so you solve it point by point and connect the points. And that's what makes a problem long-running. The more points the longer it takes to run.

Miya: You started off with 2-D codes first?

Michael: Huh?

Miya: You started off with 2-D, two-dimensional codes first?

Michael: No, the first calculations I think were not two-dimensional so much as they were one-dimensional. It's like this. If you're going to fire a projectile, that's a three-dimensional problem but you can solve it in two dimensions. You can almost solve it in one dimension. If you're going to drop something, it can be one dimension. It can be two or three depending on how you want to do it. The dimensionality that's chosen will be dependent on the guy who's doing the problem. If he wants certain kinds of precision, he'll use extra dimensions and so forth. And these points at which you do the calculation mean that you can reduce everything to say the number of floating point operations that are needed in order to get the answer. When you have say ten dynamical variables at each mass point, velocity, pressure, temperature, water vapor content, equation of state, whatever all that stuff is, all those equations have to be solved at each point. Given you got a million points that's a million times. If you had ten items at each point that's ten million calculations. And how many floating point operations that amounts to may be 100 times that much. It's a very flexible variable that depends mostly on the guy who's designing the calculation. How much precision does he want to produce? Clearly, we solve more complicated problems on the bigger, faster machines. The Univac wasn't the biggest or fastest so that the limitations there were largely brought on by the size of the machine and our inexperience. Clearly, people were more experienced by the time they got to the IBM Stretch or the 7094 about what kind of machine calculations they wanted to do. But fundamentally they're all along the lines I was describing. There's an interesting analogy in Monte Carlo calculations. Consider just some arbitrary function in two dimensions and then consider that you generate a random number and stick it in to find out whether the point it produces is a part of the function or not. Clearly, the more random numbers you use the more accurate your representation of the value of the function will be. So if you only use a few you don't have very much accuracy. If you use a lot, you have more accuracy. Now there's an art to deciding when the accuracy gained is worth the extra computation. Clearly, if you have accuracy of a 12 decimal digit say to do another billion calculations to produce another digit isn't worth it, so people learned to not do that.

Miya: Ten minutes? Okay. Well, I think-- I'm trying to think how to characterize the algorithmic history issue here.

Michael: I can't hear you _____.

Miya: I'm trying to think how to characterize the algorithmic history issue.

Michael: Complexity.

Miya: It's complexity, yes. This is true. Let me put that word here in my notes.

Michael: Well, as people gained experience and machines got more commodious and so forth, there was a natural division I would say into three parts: the part that produces how to run the machine with an operating system, the part that produces the components out of which you build the operating system, schedulers, memory allocators, fault correctors, time allocators, all that sort of stuff. Each one of these is a little separate thing and it can be written in its own little cubicle and then attached. And then the third kind we'll just call applications programming. I assume we got all that stuff that makes up the system.

Miya: Yeah.

Michael: I want to use it.

Miya: Well, we're trying to get a little bit of background on the applications part as it influences, for instances.

Michael: Well I'm--

Miya: Hold it. Let me finish. How we think of supercomputing because back when you were doing this stuff memory was expensive so were you using the whole memory of these machines for instance when you were doing a given application?

Michael: What do you mean by that?

Miya: Like you use 90 percent of the memory of a machine? We're trying to characterize super computing in a few years' time.

Michael: For the problems that ran these tens of hours and so forth they used every memory cell in the place and more than once. In some cases the data store was cut in half so you could put two numbers in a word and then whenever you brought it out you had to go through to make it a one-- split the words apart.

Miya: To pack and unpack words, see most--

Michael: Yeah, packing and unpacking.

Miya: Yeah, so most programmers these days would never resort to anything like that.

Michael: Well, understood. Then it was a necessity.

Miya: Yeah. It was part of supercomputing at the time right?

Michael: Well, no I don't think so.

Miya: Before what you would call supercomputing.

Michael: It was a part of computing that we had to do in order to get something done.

Miya: Yeah.

Michael: And the applications program would spawn lots and lots of ideas to make the system programming better for the applications user. It was a time where you would have to worry about 40 point

fault. Instead of it having represented in the applications program, somebody invented the routine that you branch to whenever you had an applications fault. That meant more memory for application points and a more uniform way of handling error and exceptions and things of this sort. There were simple little things like-- which came much later. A guy wants to keep himself in memory. If he's inactive for a certain amount of time, he will be rolled out of memory so it can be used by somebody else. But if he makes some sort of action, he'll stay in memory. So the applications programmer will develop a little program that essentially rattles his data store--

Miya: To get more time.

Michael: -- to get more time.

Miya: Right. Still done to this day.

Michael: So there was this tension, this dichotomy of pulling and pushing between the application programmer and the system programmer.

Miya: Well, in the beginning there were no such things as system programmers right?

Michael: That's _____<inaudible>.

Miya: So, in time the application programmers which had say 90 percent of the memory got edged and squeezed out by the systems programmers right?

Michael: Yes.

Miya: Operating systems which you guys resented right?

Michael: Exactly right.

Miya: And you guys resented that right?

Michael: I don't know that it was invented by us.

Miya: No, no.

Michael: We certainly used it.

Miya: Resented.

Michael: Oh, yes. Well--

Miya: We'll get to that later. Okay, I have a better example about that. No, see a couple of the things that for instance you talked about connecting parts of a grid as an example of a mesh. Is that called coupling? Can you use that word coupling? Can you acknowledge that?

Michael: I would say no but it's okay.

Miya: Yeah.

Michael: Coupling had other uses.

Miya: Right. We're not going to go into those?

Michael: No.

Miya: Okay. I know that you guys at Livermore, for instance when you were solving your partial differential equations, you guys decided to go with finite element methods rather than finite difference methods like we do over here.

Michael: No.

Miya: No?

Michael: No.

Miya: You guys explored both of them.

Michael: Finite element methods were considered coarse compared to finite difference. Finite difference is closer to the spirit of calculus so that's what was used mostly and still is. There are things like crashing cars that use finite elements.

Miya: Right.

Michael: Crashing airplanes.

Miya: The Dyna-3D series, the Dyna-2D series.

Michael: Yes.

Miya: Right.

Michael: And that was used. That was developed at the lab.

Miya: Yeah.

Michael: But the actual basic physics, chemistry and so forth that handled the behavior of matter at its elemental level was all finite difference.

Miya: So, the impression that Livermore and Los Alamos concentrated on finite element codes is largely because that's been released and the finite difference stuff has stayed behind the fence?

Michael: I don't know if that's a good generalization or not. I just can't guess.

Miya: Okay. No, that's okay. All right, well we have a rough sense of a little bit about the character of some of the codes, okay, roughly in terms of size, orders of magnitude, memory sizes.

Michael: Okay, in the beginning when the Univac was first used there may have been as many as a couple of hundred floating point operations per mass point to get the answers you wanted. There may have been-- there may be as many, and I don't really know this now but there may be as many as a couple of hundred thousand floating point operations per mass point now because the complexity of the problem has grown. The sophistication of the methods used to solve the thing has changed and grown. I think at one point when we were in the 7090 series of machines, we were up to 20,000 operations per point. It's a very crude mechanism for counting these things. You double the averaging for instance. That doubles the number of floating point operations. It doesn't say much more than maybe that's going to give you additional accuracy. I doubt it but it would.

Miya: Yeah. Okay, we have two and a half minutes here. The last little point you mentioned this-- oh, one minute-- security you mentioned that you had to have the headers for classified material on the printers. Did this also have to go with the film output as well too, the graphics?

Michael: Yes but the film is even more sensitive you know because it gives pictures and they required that we record on the film the word, classification level and that was--

END OF TAPE 3.2

START OF TAPE 3.3

<Audio begins abruptly>

Michael: _____ As it was the assembler. There was an operating system that was produced by IBM called Monitor Gate and Monitors-- The mechanism for input was fax-to-cards. And some of those cards that separated one job from the other could be recognized in the 360 version as job control lines.

Miya: JCL [Job Control Language].

Michael: Right. In essence, it was an operating system, but those people who had, you know, lots of computations that they wanted to do; many were unwilling to allocate any time for the operating system to run. So they never used it. Whether it was good or bad is for others to say. I can't. I never used it. I found I could be more efficient with just myself and an assistant who would load the DECs, unload the DECs and _____ the paper and push the buttons and stuff like that.

Miya: So you guys were starting to diverge from what manufacturers offered.

Michael: I can't hear you.

Miya: You guys started to diverge from what manufacturers were offering.

Michael: Yes.

Miya: Okay. That's I think the critical thing there.

Michael: Well, I don't think that you could pick a date and say that's when it started. It just sort of smooged [ph?] into the thing. There was a guy by the name of Pyle [ph?] in England. He wrote the use of the- some of the Fortran routines so he could write Fortran programs. Those things were the genesis for starting of the writing of Fortran/Fortran; that is to write a Fortran program that will produce a Fortran compiler. I think that work was started on this 704. I think that Sam Mendecino [ph?] and George Sutherland and Hans Bruijnes, and there are probably a few others that I'm missing, were the ones who started doing it. I know Sutherland and [Sam] Mendecino and [Dave] Storch and others published a paper *Fortran/Fortran*. On the service of things, it's a brilliant stroke of improvement. You now have a compiler, which is more or less independent of the machine you're on. As long as it understands Fortran or its conventions, you can product object decks that will run on any machine that you're running on. Well, that was very important. We take this program. It's very important and it runs on this machine. Now we have a machine that's faster and it takes two years to get this program to run on that whereas if you go through this Fortran/Fortran rigmarole, it's running a half hour after you install the machine. That's important.

Miya: That Mendecino and you said Sutherland.

Michael: George Sutherland, yes.

Miya: And Hans Bruijnes. Did this ultimately lead to LRLTRAN? Did this lead to LRLTRAN?

Michael: Yes.

Miya: Okay. All right. Let me go on now. I'm assuming you-- Do you have anything else to say about the 704, or can we go to 709?

Michael: There's not much to say about the 709.

Miya: No. No. 704 - last comments? Last comments on the 704?

Michael: Not that I know of.

Miya: Okay. So 709 - how many?

Michael: Well at least four of those too I suppose. I don't remember. I have a--

Miya: I know there was a table.

Michael: --A place where I did that census and I believe there were four of those also. The 709 was just improved components and it also introduced the idea of an I/O computer. Up to then, I/O was done out of the mainframe, by mainframe instructions. But on a 709, there was a thing in between the computer and the peripheral device and you could program that. It was fairly simple. You could say, "Do input/output. Count this. Branch this, etc." It meant that you could do that faster independently of the running program in the main memory. So it was very nice to do buffering in that sort of thing. I believe that this kind of innovation was generated by pressure from NASA. Now they wanted their machines to continue moving the capsule around the world or something like that. So those had to do I/O. So IBM has a special - I forgot the name of the exact thing, but a special purpose device, built them an I/O processor. It was "successful" enough that we all some.

Miya: This was for asynchronous I/O?

Michael: It's for all I/O. The main point about that is that it freed the central computer from waiting so it could do more computing. Remember, these guys are starving for computer cycles; never had enough. This gave them as much as they were going to get. And it worked because now you could isolate the I/O program and if you didn't like something, you could change the I/O program without doing any violence to the main computation that you're trying to do. That was a very useful thing. I think that was the major contribution from that - the 709. We had one other thing, which I think NASA inspired and we made very thorough use of it. I don't remember the name of it exactly, but I'll call a data channel. What it meant was we could be the DD-80 on that thing and then every six microseconds, we could send a word between the display and memory. We had enough control so the DD-80 could run independently of the main memory also and that meant that there was a time when you were advancing film. Even at 30 frames a second, that's slow compared to what a machine can do. So it could do that buffered and it could get a whole bunch of calculations done while you were waiting. And we did.

Miya: Was the software that NASA developed for this called HASP?

Michael: HASP came much later. Well, it came not much later, but it came from that section of the world, which was trying to use the 360. Houston Automatic--

Miya: Spooling Program.

Michael: Spooling Program, yes.

Miya: Yes, you haven't used the word "spooling" yet.

Michael: Yes. I think that was exclusive to the 360.

Miya: Okay. Any last-- Let's see.

Michael: I remember once I was invited to go over and give a talk at IBM and they introduced me and he said, "You know, this guy, he's from that place that has one of everything." Snicker. Snicker. When I got up, I said, "It's true we have one of everything because whenever we got it, it was supposed to the fast, and I also point out to you we don't have any 360s?

Miya: What was the response?

Michael: Very quiet. Snicker.

Miya: Okay. The 709, was it object code compatible with the 704?

Michael: For the most part, yes. I/O was done quite differently because now we have an I/O processor. So it's not compatible in that sense, but the main processing instructions on the 704 and the 709 were essentially the same. The way it handled faults was essentially the same. They way it handled I/O overflow and underflow is essentially the same.

Miya: Did you have an operating system on the 709 as well, or did you ignore that?

Michael: No. It was the same as the 704.

Miya: Okay. I'll just simply put down "Everything else the same."

Michael: Now we evolved and did eventually get an operating system.

Miya: We'll get to that later.

Michael: But it wasn't this thing that IBM supplied.

Miya: Right. Which machine was that first done on?

Michael: Probably the 709. I don't remember. Our operating system was lean and mean compared to the Monitor, which IBM supplied. Monitor made _____ use of memory. It thinks sequentially and so it wasn't very efficient. But that's just bad memory on my part.

Miya: Okay. Well, we're going to do seven DEC machines here before we get to the 7090. But I'll put that in for the 7090.

Michael: Well on the DEC machines and everything, you can all do it at one thing. Back in the 1960s, we heard about the PDP-1. Several of us went and talked to the company, Digital Equipment Corporation (DEC). The PDP-1 looked like it would work very well as a card-to-tape, tape-to-card, tape-to-print and so forth thing. So it would do all these peripheral jobs. That was the original intention. Our PDP-1 was the most heavily loaded thing that DEC had ever built and well, it had the usual printer okay and it had a paper tape reader that wasn't of any use to us. So we originally interfaced an uptime card reader to that - 2,000 cards a minute. But the thing was too rough on the cards and too unreliable. So we replaced it with an IBM card reader - a 1402 or whatever it was then, card reader punch. We had an Analex 600-line a minute printer on the thing.

Miya: A what?

Michael: Analex - A-N-A-L-E-X. It was a noisy thing and it was sort of not able to keep up with us. So that got replaced by an IBM chain printer, you know the kind that were on the 1401 [IBM 1403—Ed.]. That worked pretty nicely. But in all cases, the interfaces for those things were built by our engineers. We had voice input/output on the PDP-1. We had a Rand Tablet and stylus. We had the Eyeball on the thing. We had a high-precision camera to make really super movies on the thing. We had a four-channel stereo output thing so you could make beautiful music and stuff like that. There was a thing that punched Mylar paper tape for the engineers who were running special kinds of machines in their shops. We gave them punched Mylar, which lasted much longer than a paper tape, oiled paper tape.

Miya: Like CNC, numerically controlled milling?

Michael: Yes.

Miya: Okay. That was about when?

Michael: Oh, 1965, 1964, '63. There was so much going on on the PDP-1 that instead of just being an I/O device, which it never really was, everything else got done there, okay? We tested out the quality of the various typewriters that were going to be on the Octopus [network]. We tested out routines that would run on larger machines to give us such things as lines of varying density; so many points per unit, length and things like that, and how to build characters so that they were legible even when the film wasn't very good. The PDP-1 was a place that you could walk up to and in a minute or two, it'll be running on something you wanted to do. We were able to get console time there, in other words. That had an incredible beneficial effect on program development. You could get your program built much quicker, much faster, much more correctly, more commodious. We worked out there on what people now call Mathematica.

Miya: Symbolic algebra.

Michael: What?

Miya: Symbolic algebra.

Michael: Yes, some of that, but it was not quite as elaborate as what they have now. What we had then for instance was a thing that was called Callagraph [ph?] if you will, but just to quickly wrap up some kind

of a display. Here is this function. I'd like to see how it looks in three dimensions. So that was just a couple of type strokes and code the thing in into this thing and you could see a display of a three dimensional thing. You had many ways of seeing it. You could put a view for each eye and put a separator between your eyes. So you see that stuff. You could even orthographic projection. There were just lots of stuff. That was the work of Ken Bertrand [ph?]. We used to have fun with that - "Bertrand [ph?] doesn't write in Fortran." He's a good guy. He was really a very good programmer.

Miya: Okay. How many PDP-1s did you guys have?

Michael: One.

Miya: Just the one.

Michael: Well, only one in comp, yes. There may have been others at the lab - I don't know.

Miya: Okay, and the languages and the operating systems and other software things?

Michael: No operating system. There was a bootstrap paper tape that would launch the operating system, if you will, so you could run an assembler. There were a lot of CAD programs; like say, "I hung a tape on such and such a place. Go get it and print it." That's a CAD program. We found that the Potter tape drvies that came with the PDP-1 weren't fast enough or reliable enough for lots of the work that was being done. So again, our engineers built an interface so we could have IBM type 729 Mod 6s, the best tape drive that IBM had. We had two of them attached to the PDP-1 also. That meant that if you take a tape from either any CDC machine or IBM machine and what we call the physics table. These are the machines that are doing big important work. You bring that tape over, put it on one of these IBM things and do anything you wanted - print, sort, edit, whatever. So the PDP-1 was what I believe we called generally our "Romper Room" where us little kids could play and try out ideas and learn a lot about applied computing. I got to know Ben Gurley, the designer at DEC, pretty well and I remember him telling me that the engineers at DEC liked very much to work on our machine there because it's going to have all these weird things attached to it. It was the most heavily loaded I/O machine that they ever produced as far as I know. It had 4,000 words of memory and these are 18-bit words. As I recall during its lifetime, we added 4,000 words and we were given the opportunity of adding another 4,000 for a total of 12K. But I don't think we ever did. It had some CalComp plotters on the thing too now that I remember. A CalComp plotter is an incremental plotter, right?

Miya: I know. I used one.

Michael: It moves in any one of eight directions in steps of a hundredth of an inch or some constant small amount. You can raise and lower the pen. So with that capability, you could draw all kinds of very interesting and never before seen pictures.

Miya: You told me once when we were driving on Highway 580 about some DEC guy who died or committed suicide or something. That wasn't Ben Gurley was it?

Michael: That was Ben. He didn't commit suicide, no. He was killed, murdered.

Miya: Oh, he was murdered. He was a murder victim.

Michael: Yes.

Miya: Was that local, or was that in Massachusetts?

Michael: Well, I'm not sure about the exact story. I believe it's been recorded by now. But Ben was working at Lincoln Laboratory and he had a guy that worked with him named Al Bloomenthal [ph?]. When Ben came to DEC, he brought Al Bloomenthal [ph?] with him. Then when Ben left DEC to go to work with Ed Fredkin, Al Bloomenthal [ph?] thought that Ben should have seen to Al Bloomenthal's [ph?] promotion, but Ben didn't. That festered inside of Al Bloomenthal [ph?]. So roughly, he bought a 28 guage shotgun and a box of shells, gloves. One evening, he went to the back of Ben's house. Ben and his five kids and wife were sitting down for supper and he shoved the shotgun through the window and blew off Ben's head. The irony is that nobody arrested him and it wasn't until Ed Fredkin and Ed Burley [ph?] retained a private eye that they collected enough evidence to show that that's the guy who did it. So when he was brought up, it was quickly seen that he was mentally unstable enough so he couldn't stand trial. He was put in a home for the criminally insane and every now and then they'd asked, "Have you gotten rid of that list yet?" He had made a list of people he was going to kill. No, he still had the list.

END OF TAPE 3.3

START OF TAPE 3.4

Miya: So what did you say about visionaries a second ago?

Michael: I was saying that even at the Univac time when most of us were children, learning about this funny thing called a computer, there were guys who saw the more important aspects of the machine and how it could be used. They were strange people but they did see these things.

Miya: Can you name them?

Michael: No, I can't. I had the name but it's gone. But I think, you know, that some of them I can like Tom Wiler [ph?] was one of those and Marit Elmore [ph?] and Lester Bomhoff [ph?]. He was crippled but he was an incredible mathematician.

Miya: So these are all mathematicians and physicists?

Michael: Mostly mathematicians yes. There were physicists that do here and there. I think from my own point of view the person that most influenced me and gave me most insight into where we were going was Norman Hardy, in fact, I could say Norman Hardy was one guy who had a tremendous influence on me. Chuck Leith is a guy who had a tremendous influence on me. Dana Warren had influence on me. There were a few others but, you know, these are the most important. In a way, my bride had a great influence on me, just pushing me.

Miya: Heidi had an influence on you?

Michael: Yes. I liked her.

Miya: Did she work at the lab at all?

Michael: No.

Miya: It was just simply raising the kids.

Michael: She was a blood chemist and, yes, when we started having children she stayed home and raised children. And I think she did a tremendous job because, you know, we have some beautiful children.

Miya: But I want to say on the PDP-5, let's finish up the early DEC machine.

Michael: Okay, the most important thing about the '5 is it metamorphosed to the '8. Okay. The PDP-5 was low enough cost so you could put it into a guy's lab and let him learn how to use computers in his laboratory. I have a guy by the name of Don Watson who used one of the generators called the '12, PDP-12, and in a sense he thinks of himself as having one- one of the first persons who had a personal computer. And '12 was unimportant enough so nobody tried to take it away from him. And he builds all kinds of programs for the thing to help him in his lab. He's what I call a real doctor in the sense that he had an MD.

Miya: He was an MD? Was he a health physicist?

Michael: No. He was studying neurology I suppose or– he was one of the first- one of the things he studied was that you take out these micropipe patches and you shove them into a cat brain and then poke the cat to see what kind of a signal you can withdraw. And, you know…

Miya: But he didn't use the link for his…

Michael: He used the [MIT] LINC [Laboratory Instrument Computer] but I don't think it was the first LINC. PDP-12 evolved to the LINC. And the '12 itself evolved from the '5, '8 bind. And it had the benefit of some other DEC people, Wes Clark, Brent Royally, they was interested in building a machine that could be an adjunct to a researcher in the lab, okay? That's important, you know, that it wasn't something that sits in the guy's living room unused or something like that, it's in the lab running experiments, taking data, processing the data. All that was done because the thing could be there and the guys who run these labs are creative individuals. Don Watson's a creative individual. And it isn't to say that everything he did was super perfect or anything like that, it's to say that he did a lot of stuff though.

Miya: So I want to move onto the [PDP-]6, but can you say something about the software of these machines or the peripherals, diverse peripherals. I probably shouldn't cover all of them. Do you know roughly how many PDP-5s? Too many to remember?

Michael: No, I don't remember anything like that. I think that as far as operating system stuff goes in the DEC world I don't really have any memory of anything until you get to the '6 and the '7 and the '9, '11.

Miya: Nothing like MUMPS [Massachusetts General Hospital Utility Multi-Programming System] for instance?

Michael: I don't remember. On the PDP-6, to me its unwed mother was the PDP-3 but it never got built. I remember Ben Gurley saying that one of the smartest things he ever did was to not do the PDP-3, okay? The PDP-3 was supposed to be like two PDP-1s together in the sense that the operand was 36-bits, not 18. And it was, to my way of thinking, a desirable machine because it would mesh nicely with all the other 36-bit computers that we were acquiring in the lab. Okay, so the PDP-6 I think got started this way. Ben Gurley and the president of DEC, Ken Olsen, made a trip around the country talking to people about a new computer, okay? When they got to the lab, we introduced them to the concept of registered files as we knew them on the LARC. Ben liked them. So the PDP-6 became the first of that class of machine that had registered files as a nice context switching thing, takes no time, et cetera. The PDP-6 became a darling of computers because whereas speed wasn't the most important thing, it was fast, but it had an incredible repertoire of instructions, so much so that it got slowed down to handle them all. And again, the world's most creative people got a hold of those machines and developed all kinds of marvelous things, okay? Lisp could blossom on that machine, okay? There were others who wrote fairly exotic stuff for the PDP-6.

Our purpose at the lab was that we were having our birth pains in a thing called the Octopus then, which as we saw it, Norman and, I to begin with was a resource sharing thing. Timesharing was important but Norman, I remember, didn't want to push it beyond the Stretch. The Stretch would be a machine and the only way to officially use it was to share it. But a PDP-6 was to be the tie point for all the big what we call physics computers at the lab, okay? And at the time, we could say that physics computers were 7094s, 6600s, and Stretchs. So we have these things scattered all over the place and it would all be hooked to this great big disk, a disk that had almost 8000 heads on it. And these heads were divided so that each computer had a set of heads where each could write. All the computers could read anywhere on the disk and the PDP-6 was to mange that. And so that meant that Eugene Miya's XYZ computer and Miya Eugene's YZX computer could talk to each other through this big disk. This was supposed to be built for the army by Burroughs, it never got built -- probably we're lucky. But anyway, the PDP-6 is chief _____ with the birth or the conception of what the Octopus became: a place where you passed messages back and forth, where you stored data and had it available whenever anybody wanted it. There were ideas like, god, I don't want to solve those sets of equations anymore, I've done it ten times now, so we write a general program that becomes these equation solvers. And now you just feed it raw data and it'll do your problem. I feed it raw data, it'll do my problem and so forth. That never came about either.

Miya: Well let me ask you this because we're going to be running out of time here in a moment. Roughly how many PDP-6's did you guys have? Just two?

Michael: Just two. Well, there may be some others outside the computation center. These two were to manage the Octopus. Now, it turns out our vision of the Octopus, as I've described it to you so far, can be characterized as the star connection. Right? Okay, that turns out to be not good. And we had nothing but maintenance problems, reliability problems, missed deadline problems of all sorts while we tried to build this star-like connection. Every machine had to have a special kind of a gadget attached to it so it could talk to every other machine and so forth. So, in the just more or less recently, maybe 20 years ago, we went to a-- I guess you would call it party line connection, okay. Everybody's on this wire and you all listen to everything and the only stuff you chase is when it's got your address prefixed to them.

Miya: Like Ethernet?

Michael: Yes. But there wasn't an Ethernet then.

Miya: Right, CSMA.

Michael: Well there was- you know, I think that the biggest difficulty at the lab was that we had everything except we didn't know enough to come out of the– out from behind a fence to talk to other people about what they were doing in a substantive way.

Miya: And that's why you hired Dick Watson?

Michael: No, I don't know why they hired Dick Watson.

Miya: I think that's why you guys hired Dick Watson.

Michael: Well, it could be. I had nothing to do with that. When I hear about all these marvelous things that people want to do and so forth, I can look back and remember that we tried this, we did that, we thought about all this kind of crap, but we never tied the knot.

Miya: Right, you guys never published papers on these failures.

Michael: That's one of the very big serious problems -- we never published papers. And I don't really care about that except it would have been nice to have record of what actually we did think about okay.

Miya: And that's why you had this museum.

Michael: Well, yeah, in a way I suppose that's true. But, in any case, some of us did come out after a while and got to recognize, hey, that's a better idea than the one we had, you know, and we go on from there.

Miya: Okay, I want to wrap up the PDP-6 so I can go onto the PDP-7. So, the PDP-6 was not really used as a general-purpose computation machine, it was merely a synchronization point for communications.

Michael: Every machine at the lab was used by a person, whoever it was, for whatever he damn well pleased, but the nominal function of the '6s was to service the Octopus.

Miya: And we'll get to Octopus later.

Michael: All right. And what we found is that by the time we got to the place where things were settled and reliable and so forth, the '6s were obsolete and metamorphosed to '10s. We did everything the wrong way in a way. We built the memory to add to the '6s. We could have bought it, it was more

expensive but it worked. The stuff we built had bugs in it, it was unreliable, and would, you know, put everybody down. I mean, you would want to run and all of a sudden you can't do it.

Miya: So some people would say you guys were early indicators of what would become to be _____.

Michael: If you want, I don't know. You know, it's okay with me. I think that PDP-6 had one other major contribution and that was it managed the photostore [IBM 1360]. The photostore had a ten to the 12th-bit memory [one trillion bits] and when we put it online in 1969, it gave the lab more online memory than the rest of the world put together. That maybe is an editorial in the sense that let's say there weren't- you know, a 7094 had a million bits of memory, it would have taken a million 7094s to give us 10-minute 12 bits of memory. A photostore had 10-minute 12 bits. And it, in my opinion, was one of the three most important things that all the effort at the lab produced. It gave us incredible memory; it never cost anybody a lost bit. You put your stuff away and you could get it back. You could share it. So, it was a progenitor of an enormous number of innovative ideas in computing.

Miya: Okay, well we'll have to save the photostore for some future time. How close are we to when you want to leave? Five minutes okay. Do you want to say anything else about the PDP-6 before we move onto the PDP-7?

Michael: The PDP-7 is entirely different from the '6.

Miya: Anything else about the '6 though?

Michael: Not immediately.

Miya: Okay, we'll get to the photostore later.

Michael: And as far as the '7 goes, it was just a workhorse computer.

Miya: You guys had lots of them?

Michael: I don't think we had very many of them at all.

Miya: But it was done for what purpose or…?

Michael: I don't think we had very many PDP-9s. I don't think we had many PDP-15s. What we got a lot of was PDP-11s and VAXs. And that's largely because the micro versions of those machines were so cheap it was better to get it and just use what you wanted, than [the idea that] the designs that were special. So the- all of these- the PDP-11 and the VAX, MicroVAX, and to some extent the '8s before them, were all used in the Octopus as servers. One of them serviced each Teletype network. We had several Teletype networks. Not because we had different things on them but just because we had so many Teletypes. There were some that serviced the television monitors places. Some would service the entry into the photostore and the data cell thing that CDC sold us. So these machines, the '5, the '8, the '7, '9, the '15, PDP-11s, all were, you know, not used by users but were part of the structure called the

Octopus to supply services. We don't see whatever else it might be. We had, say a PDP-5, then a PDP-8, that serviced 128-Teletypes, okay? It didn't service other than one kind of Teletype, but it served 128 of them. We had another one that served 128 Teletypes. There were- I think running out to everywhere 256 Teletype channels, that were vastly more than that Teletypes and the question of what was hooked to what at any given was in the hands of these computers. So it could tell you, I have to have three channels here because I want all the bits to feed to my television display to make a color picture or a gray-scale picture. Fine. Those machines would do that for you. In ordinary things, if something gets munged up, it's the PDP-8, '9, '11 whatever it might be that gets it straightened out. Okay. So I don't know what you call them, is it server computers? Good to know.

END OF SESSION 3

START OF SESSION 4, July 12, 2006

Miya: It's July 12, 2006. We're again here with George Michael. This is session number four. We left off after finishing the DEC PDP-1 through '15, and now we're going to go to the IBM 7090. But first, George, do you have any good jokes?

Michael: No.

Miya: George Michael is famous for telling jokes.

Michael: They're not jokes; they were true stories. I don't remember any of them now.

Miya: Anyways, I'll plug you again in the future. Okay, so tell me something about the 7090 series first.

Michael: Well the 7090 was a transistorized version of the 709. And the 709 had the benefit of an improved I/O command structure, thanks to NASA, and improved floating point hardware and commands, thanks to the AEC. It was not fulfilled, but it stabilized the 36-bit word.. And from the point of floating point, 8 of those bits were assigned an exponent and the 27 bits were the fraction. So there's 27 and 8, 35, and then assigned bit 36. Until we had more experience with Seymour's machines that had 48 bits and 60 bits and the Stretch which had 64 bits, 36 bits was considered okay for doing the kind of calculations that had to go on within the lab. Though the mathematicians were busily pointing out all the time that by the time you finished the calculation, if you had one significant digit left, you were lucky. However, the 7090, in a sense took away a lot of the thunder that the LARC was going to provide by being a transistorized machine, though the LARC was the first designed transistorized machine that we knew about. Its speed improvement over a tube based machine was so much, but the 7090 achieved most of that by having transistors and that meant it just was less of a splash for the LARC.

I think the people who, shall we say, caused the existence of the 7090 were great visionaries and very practical designers at IBM. They had the example of a transistorized machine that Seymour produced, the CDC 1604. Rather than going into a special design effort at IBM, they just transistorized the 709. The 709 had sort of I/O channels that you could program to run under the control of the CPU and it gave us the rightful control over I/O processes. The 709, I think it was NASA that got it. I can't remember the name of it, but it produced a 36-bit word every 6 microseconds that you could move from one place to another over that channel. That was enough for making the display devices on the 709 very useful. We

ran the DD-80 on that and it went over to the 7090 with great ease and the DD-80 was able to produce a character every 9 microseconds, I guess it was-- 6 to 9. Vectors drawn from 6 to 30 microseconds and plotting a point every 11 microseconds at the most. So it was the world's fastest display at the time. We equipped it with a special camera that had been originally designed for the spark chambers up at Berkeley and they just fixed it a little bit differently for us so that we had pin registration and we could move 30 frames a second when the thing was free-running. At that, we could actually have some programs running, producing graphical output at the rate of 30 frames a second. That was very impressive and in its own way, maybe made people a little more sloppy than they ought to have been. But it was the thing.

Miya: Was that color?

Michael: Huh?

Miya: Was that color?

Michael: It was black and white.

Miya: Black and white only at the DD-80?

Michael: Yeah, we made color by logical color separation.

Miya: Right, RGB.

Michael: Yeah, well we switched to the '94. The 7094 is a 7090 with improved logic on the one hand and making use of the Stretch memories. The Stretch memory had a 64-bit word with an 8-bit ECC, single bit error detection and correction. So it was 72 bits. That meant from a 7094 point of view, that was two words. You could fetch and, in lots of cases, have almost a zero execution time or fetch time, I should say, for the second of the instructions. If you learn how to program that, you could arrange to get that to work much to your advantage. Other than that, the 7094 was a little more reliable than anything else that we had. I think a great number of people liked it and it led people to use the machine beyond its capacity in the sense that you were trying to run programs that were essentially too big for the machine. There was a lot of calculating in the interstices for I/O commands. You could be computing on a band, outputting a band, and inputting a band. It did that little 3-way role and everything could be hidden, buffered, so the thing looked liked it was running very, very fast with no penalty for the I/O. The 7094, I don't remember how many we had now, maybe 4 of them, but it was a nice machine. It was, for us, the culmination of the IBM 7000 series. I'm told that it had its infancy in the thing that was done at Lincoln Lab for the Air Defense SAGE System. IBM had the contract there to build the memories, but they attached the instruction set and the architecture and stuff like that and made the 704, which was very compatible with the SAGE system, from what I'm remembering.

We had the fastest of the 7000 series in a sense, which for us was the Stretch, the 7030. Its memory we had on our machine, 96k. That's each word with 72 bits, of which 64 was the data word and 8 were the ECC portion. On the 7094, that word was divided equally in two parts, two 36-bit words. People were able to exploit this little bit of symmetry and get what they wanted. The Stretch, I remember Norman [Hardy] saying he can't imagine a bigger machine ever being built. But then, we didn't plan on the

engineers. They built a Star, which was quite bigger than the Stretch. Not that it was better or anything like that. The Stretch, like the LARC, was late. It had troubles with the transistors that they had chosen, and their engineering unfamiliarity with their behavior. The Stretch probably had for us, the least amount of software delivered to us from the manufacturer. Most of the stuff that we had on the Stretch really was done inside the lab and it was very minimal, I guess is the way to put it. The Stretch was a machine that was, again, had its thunder stolen a little bit. It was supposed to be so much faster than the 7094, but again, the transistors, the improvement that you expected to get from the transistors, was there but it was ho-hum. Maybe it was a factor of 2 or 3 speed-up with the Stretch over the 7094. The overall effect at the lab was that we had gone through essentially about a solid decade of improvements in the machines that we were using and there was really improvement all the way. There was hardly any thought about how to design machines so that they would perform better. It was just expected that the manufacturers would continue improving whatever they were doing and delivering a machine that's 2, 3, or 4 times faster. There was a point at which I think they actually had a limit. If the machine only improved over its predecessors by a factor 2, we said we weren't interested. The basic reason for that is that the software that had to be moved from one machine to another had such a cost for movement and so forth, that if you couldn't get a factor of 4 speed-up between the two machines, the extra cost of doing the software made it not worth the effort. I think that persisted all the way up to the 6600.

Miya: What I'd like to do actually is, you made a number of statements of--

Michael: I know.

Miya: --three or four different machines, which I let you do. I want to go back and I want to fill in for my notes--

Michael: Do it!

Miya: --some of the things. We'll get to the 6600 here in a second. You also talked about the issue of scaling things here.

Michael: The what?

Miya: Issue of scaling.

Michael: Oh, yeah.

Miya: Factors of just two and four, which I gratefully have learned from you, the great master. Getting back briefly to the 7090 and you're crediting the 1604 for influencing the 7090. You said that the 7090 was designed by great visionaries and practical designers. Who were these great visionaries and practical designers of the 7090 that you're referring to?

Michael: I don't know their names specifically, but I think Bo Evans was one of them. I don't remember. This is strictly an internal IBM thing. What the 7090 represented was a short-circuit attempt to convert from tubes to transistors without the complete redesign of a machine. As I heard the story, it was-- they had this meeting at IBM in Poughkeepsie, saying what we've got to do about this. They decided then that

they wanted to work along the path of getting a 7090 from a 709. The engineers stepped up and said, "Here it is." They didn't have to wait; it was there.

Miya: Was this the book that you had me read, called "Memories That Shaped an Industry" by Emerson Pugh?

Michael: I don't remember that being in there, but he certainly talked about IBM's role in memory construction and memory building and stuff like that. I remember the numbers that it was over $1 a bit when IBM took over the problem of building memories, and they reduced the cost to three-hundredths of a cent per bit. But they didn't really pass most of that saving off to the users. It just meant nicer profits for them. I think Emerson Pugh pointed that out in his book.

Miya: What about 7090 software?

Michael: The 7090 software was upward compatible from the 709, with the exception that-- no, I guess it was just upwards compatible with the 709 practically all the way. It was faster and it had an additional set of instructions, none of which I remember at this point, but other than that, it was a transistorized 709. It was really a very, very impressive machine, because of the compatibility on the one hand, because of it's improvement in speed on the other.

Miya: Do you want to say anything about the algorithms that were used in that period of time?

Michael: I don't remember any particular algorithms that were used in any general way. We had algorithms that would draw a circle without taking any square roots or making any divisions. We had algorithms that would allow you to take a square root by using a table and a step of interpolation that were accurate enough for the purposes of their calculation and very, very fast compared to anything else. We had algorithms that did the same sort of thing for-- if we had a series of sines or cosines to compute, the way we did it was very fast, faster than anything else. But the fundamental thing-- let's put it this way, this is truly an estimate, but it took about say 500 instructions to compute something at each match point on the Univac. To do the same match point on the 7090, it took about 20,000 instructions. While the machine was faster, it had a heck of a lot more to do per mesh point in the calculations. I think that upward trend persisted in the other machines, I heard as high as 200,000 instructions per mesh point to compute, say on the 6600 or the 7600, I'm not sure which.

Miya: Were these problems zero dimension, like just a point, like a calculator might do? Or were they one-dimensional, two-dimensional, three-dimensional, four-dimensional tie bearing?

Michael: All of them. All of the above. Whatever the person had to do for his own calculations for his own project could be one or two or three or four-dimensions or more, but he could make use of these algorithms, which just passed from one person to another without very much formalism being involved.

Miya: It would seem because you don't have a lot of memory, that you can't really do very many big 4-D problems, as an example. You couldn't do that many match points.

Michael: You do one at a time. Secondly, because of the buffering of the I/O, you could get that essentially free. You didn't have to pay any delay time to move stuff in or out. It could be done buffered. As a consequence, you just were compute bound. I think it was pretty good.

Miya: The 7094 and then the Stretch, were they instruction set compatible with the 7090? It would seem like in the case of Stretch--

Michael: No, the instruction set under the 7094 was compatible with the 7090. But the 7030 was totally incompatible. If you look at the genealogy line of these things, the Stretch really represented the first glimmerings of the System 360 that IBM introduced in 1964. The idea of what these instructions might be was largely worked out on the Stretch, the 7030. One of the big things from our point of view was that you could run what's called a 32-bit calculation or you could run a 64-bit calculation. We could find out that 64 bits was quite adequate for the accuracy and stability, but the 32 bits wasn't. So somewhere in between, there was a place where it was okay and you didn't need to carry 64 bits. Actually, the way that worked out, I think, is that most of the extra bookkeeping that was required and so forth used up the memory cycles and the space to a large extent that we didn't really get much advantage. It was just learning that you could run the Stretch in a 64 or 32-bit mode and the people who hand programmed their stuff used both, depending on what they were doing. For instance, you could move interfaces with 32-bit precision, no sweat. On the other hand, you can't compute thermal diffusion or shock migration with 32 bits. It takes more than 32 and it certainly is satisfied at 64, as far as the Stretch was concerned. The Stretch, as I said before, was late as was the LARC. The Stretch wasn't quite as late as the LARC was, but--

Miya: The LARC is next, so hang on.

Michael: Huh?

Miya: The LARC is next. Hang on before we go to the LARC.

Michael: Well, I'm not going to talk about the LARC right now. I'm just saying they were late.

Miya: Okay.

Michael: The other machines that were clustered around made the presence of the Stretch less exciting to a large number of people. But from the point of view of weapons design, the Stretch paid for itself in the 1962 test series that was conducted at Christmas Island. It was a joint exercise with the British, as I recall. They were able to do things with the Stretch in support of that test series that made it worthwhile. It, so to say, paid for itself. Now you can take that any way you want. The Stretch, we heard very interesting warnings, you might say, from the IBM customer engineers that would come out and fiddle around with the Stretch. The Stretch had 250,000 transistors in it, something like that. The statistics on that said every 20 minutes, a transistor would pop. But our experience was that the Stretch was quite reliable and it didn't go belly up every 20 minutes. It was maybe once or twice a day it would be. That was adequate for the kind of stuff that we were doing. The other thing that the Stretch had was a fairly extensive reporting system that reported all errors. That reporting could, itself, be run through some statistical analyses to give us ideas about trends and things like that. The engineers used that pretty well. They kept the Stretch running very nicely. The Stretch, I think, was on a boundary of we're going out

beyond IBM cards on the one hand and plug boards on the printers and stuff like that. All that stuff disappeared after the Stretch, but the Stretch had all of them--card readers, card punches.

Miya: It was kitchen sink machine. It threw everything in.

Michael: Yeah.

Miya: Right. Kitchen sink machine.

Michael: Stretch had that name--Stretch--because they were stretching the technology everywhere they could.

Miya: What about the software on the 7094 on the Stretch? Was the 7094 software the same as the 7090?

Michael: The software was only compatible at the Fortran level. We had our own in-house Fortran translator and it had been made to run on a '94, as well as the 7030 and every other machine that they could get their hands on. The users didn't see any difference in their code preparation. There might be a few things like if there was a job control language, you had to put different cards in front of your deck to say what you wanted to do or something like that. But there wasn't very much of that on the Stretch. It was--

Miya: So both of these machines had operating systems, then?

Michael: Huh?

Miya: Both of these machines had operating systems, then?

Michael: What do you mean?

Miya: You had an operating system. You had a scheduler, process migration--

Michael: Operating systems then, where each machine was an island unto itself. It didn't share a network or anything like that. On the Stretch, the operating system was "very primitive." It served the purposes of the laboratory well, but the Stretch was, to the people around, not in the mainstream of computer directions anymore. So they're not going to put much effort into it. It came, it was used, and it left in a reasonably short amount of time. There were people who learned about interesting things that the Stretch could do. The Stretch could do binary-coded decimal arithmetic, which ended up appearing in the 1401. The Stretch did 32-bit stuff. That ended up in a whole System 360 and then the Stretch had its own 64-bit operands and stuff like that. I'd say that they got all they could out of the Stretch. The Stretch came to us at about a 42% discount over its announced price. That was because it had "failed to meet the predictions that the engineers had made for its performance."

IBM sent out a crew of people under Sully Campbell and I've forgotten who else, but they came and they got special clearances. They were looking into the codes that we were going to run on the Stretch with the idea that they were going to show the laboratory people how to do large scientific calculations. But I remember Sully saying at the end, we had a last meeting before they took off, that they had failed to find any significant stuff. That the lab was doing stuff much better than you would expect, and we couldn't expect more speed out of the Stretch, because of its design. Stretch was being crippled in a sense by the so-called lookahead unit, which as it was fetching, would look ahead to see what was likely the next instruction and to keep it close by. Then, of course, it didn't happen that way, so it would slow down. All things considered, the overall construction cycle was being lengthened in the design and construction phase by each person who had to deal with it. It was an overly long instruction cycle and the consequence is it was a slow machine. They adjusted the price of the Stretch to us, I believe, by taking essentially a 42% write-off. But the way that did it, was they wrote off stuff that was peculiar to the Stretch only. That is, what was called the low-speed exchange of the CPU. But they didn't write off the cost of the memory. That didn't get a 42% discount. So even though they were able to produce the stuff pretty well, we didn't get any benefit from that. I know that when Sid tried to order another machine, since it had been knocked down by 42%, IBM wouldn't sell it. So in overall, I think there were 11, yeah, maybe there were 11 Stretches built overall.

Miya: And you only got one of them?

Michael: And we had one. The first one went to Los Alamos. I don't remember us getting serial two. Places that went to them were probably a security agency had one and the CIA had one and we had one. There was probably one or two in the data centers, things like that. But it didn't ever approach the economies of scale that the 7090 did. There were well over 300 of those that had been build and sold and branded. The same thing for the '94. The Stretch had 11 machines.

Miya: Do you know roughly how many 7094s were made?

Michael: I think about 300.

Miya: 300 for both?

Michael: Yeah. There used to be a little census published. I think it was done by Brookhaven National Laboratory.

Miya: I recall that.

Michael: None of us really paid much attention to it. It was there.

Miya: You mentioned the labs own Fortran. Was this one LRLTRAN was developed?

Michael: Yeah, it was the original version of LRLTRAN.

Miya: On the 7090 was the first one?

Michael: To be honest, Gene, I don't remember which machine it was first put on, probably a '94, but it could have been the '90. The idea was to write a program that would read Fortran and produce a Fortran deck. That simplified the interface for every machine that it was going to run on. The very small amount going from the Fortran statement to the assembly language statement. Except for some incompatibilities with the Fortran standard that crept in as the thing aged, the LRLTRAN was a good thing. It justified the efforts and the visions that people had about it. It could move from one machine to another, moving these tough things like operating systems, the file managers and stuff like that. It could move there with a minimum of hassle.

Miya: So you guys didn't rely on the vendor, the IBM Fortran compiler at the time?

Michael: No. Well, we had it, but I don't think we relied on it.

Miya: And that varied with the 7094 and then later the Stretch?

Michael: On the Stretch, I should correct myself. It's entirely possible that it was the IBM version of Fortran that we did use, but the I/O was produced for that compilation was home-grown stuff, to a large extent. The problem is, and you weren't really deeply involved in this thing yourself, there weren't any reports to read that were circulated among the staff or anything like that. It's largely hearsay that fills in the gaps now about what was happening and what wasn't happening. Suffice to say that the LRLTRAN, as it eventually got to be called, was pretty effective. It got us to be able to move from machine to machine in the shortest amount of time.

Miya: You guys didn't call it "portability" back then?

Michael: I suppose you could call it that. By the same token, what else would you call it?

Miya: Moving from machine to machine. It's terminology. Do you want to say last things about those three IBM architectures before I go to the LARC?

Michael: No.

Miya: Okay. Let's go to the LARC. We have five minutes before the end of this tape and then we'll take a break. Tell me a little bit about the LARC. What did LARC stand for?

Michael: Originally, it stood for Livermore Automatic Research Calculator, something like that. It was built to the specifications which the laboratory listed. The laboratory's experience, by the time they were writing specs for the LARC, included the 701, if you want a little bit, and the Univac. The Univac was a decimal machine. That's what the LARC was. The only other decimal machine that the lab had-- well, that's not quite true. The IBM 650 was a bi-quinary decimal machine, but there wasn't a mainstream computer. The LARC was a remarkable machine in its own way. It had one weakness and that was the Remington Rand tape drives. They were not very good. Our engineers and technicians at the lab redesigned that interface so it would adapt to IBM 729 Mod 5s or Mod 6s. Then the tapes worked very well. We've subsequently found that by adjusting the inner record gap to make it bigger than IBM needed and definitely bigger than what Remington Rand needed, it would do away with most of the errors. The LARC pioneered so many other things though, that were pretty important. The idea of register files for

very rapid context switching, the idea of separating the I/O from the CPU and both being able to proceed independently on their own thing, those were very good. The other thing that was nice was that the LARC met or exceeded every specification that it had to meet. The LARC was a very good machine and its thunder was largely stolen from it by the other machines, the 1604 and the 7090 and the 7094 and the Stretch, all had the use of transistors also. The LARC was the first one that it designed and they ran a file of the people who were going to supply the transistor for the LARC. They eventually gave up and couldn't do it. So Remington Rand-- well, Eckert-Mauchly had to backfill and get their own transistors.

Miya: So it was Univac then, that made this machine for you guys?

Michael: Made you what?

Miya: Univac made this machine for you guys? Who made this machine? Who built the LARC?

Michael: Remington Rand, it was then called Sperry. It was built in Philadelphia at their plant, which is where Univac was built. There were some of us who thought that it would have been better served if they'd built it in St. Paul with better designers there. There's this book From Dits to Bits [A Personal History of the Electronic Computer]. I can't remember, Herman... [Lukoff] I can't remember his last name now.

Miya: That's okay-- Dits to Bits.

Michael: Dits to Bits. He talks about his own contribution to making the LARC a reliable, deliverable machine, and how they did a lot of the stuff at St. Paul. But the LARC was a very good machine. Its word was 12-decimal digits. The only anecdote I have about it is that when the specification was presented and the order was made, we opted for 30,000 words of memory. It was possible to have up to 100,000 words of memory on the thing, but it may not have been possible to run it very well that way. But in any event,

Miya: Okay, we've got to stop; we've run out of tape. Sorry.

Michael: Oh.

END OF TAPE 4.1

START OF TAPE 4.2

Miya: You were talking about 10,000 words of memory.

Michael: I was going to tell you about an anecdote that I picked up. As I said, we ordered 30,000 words of memory. It could hold 100,000 in theory. When [John] von Neumann, who was on the advisory committee in Washington, heard that he was telling his friend Edward Teller, "That's too much memory, you don't need that much." It's well known that he thought that a 1,000 words was quite adequate. Edward said that he was too smart to argue with von Neumann in Washington, but he came home here and he invited John von Neumann to come out here, and discuss the question. And basically the

message that came through in the meetings that they had was, “You’re right, if you’re a von Neumann you could deal with 4,000 words and make it work. But most of the people that’re going to use the LARC aren’t von Neumann-like, and they need more memory,” and von Neumann agreed. So we got 30,000 words of memory. And Edward considered that as a reasonable triumph of what you might call political massaging.

But the LARC, it had some very strange stuff in it. They would just power it with one voltage, and then knock it down to get to the different voltages, rather than have transformers all over the place. And the LARC had another nice feature that was tried for the first time, and that was to record electronically on a film for print-outs, okay? They got a machine built by Stromberg-Carlson, which was called the Electronic Page Recorder (EPR), and it wasn’t very good. It lacked robustness, is perhaps the best thing you could say. In the displays you could see an AC component on the lines it had drawn, and it really didn’t expose the film correctly, adequately so you could print, enlarge onto a piece of paper and stuff like that. So we re-designed the LARC EPRs, and put them into operation. The original EPRs had been designed to do their own automatic film processing, and that just didn’t work. The long and the short of it is it was considered to be not worth the effort to try and get an automatic film processing on that thing. It didn’t save you much time, and it was an enormous maintenance problem. So we just got better film and good lenses and good CRTs and good power supplies and put them all together, and it was a fairly effective. The EPRs were a very good way to get lots of numbers printed out real quickly, and then you could decide by looking at them on a microfilm reader what ones you want to bake up and produce the paper. Another thing that I think was nice about the LARC was the drums, of which there were 12, were so designed that one rotation of the drum gave you 2,500 words. But the point is you could start anywhere in the rotation and get the-- as long as you took the whole thing, you got all the 2,500 words in the right order and stuff like that. And that had an enormous effect on, again, buffering, and hiding the I/O cost behind computation.

Miya: So the buffering was very important.

Michael: Yeah.

Miya: What about the software?

Michael: Well, again, the software, such as it was, was mostly homegrown. We had an industry designed or maybe even produced Fortran compiler, but it was modified to be something from the inside very compatible with the LARC and stuff like that. And I don’t recall there being any monitors or time-sharing executives or anything like that on that machine. When a person ran on a machine, he was the only person on that machine.

Miya: How many LARCs did Livermore have? One or two?

Michael: One.

Miya: One only.

Michael: I think that there were only two LARCs built. One that ended up at David Taylor Model Basin, and the one at Livermore. But as I said, the LARC met or exceeded every specification that had been

placed against it. It was a very good machine, very reliable, and it had these new features which people learned how to use. Separate I/O, so you could control whole I/O processors, without bothering with the CPU cycles at all. And the idea of rapid context switching with these various register blocks was very, very helpful, it was a new idea when it first surfaced in the LARC.

Miya: As I seem to recall, you gave the LARC another more humorous name, because you called it a single user machine. Now that single user, wasn't his name Leith or something like that?

Michael: Oh.

Miya: This is Leith's personal computer, or...?

Michael: Yeah. Well, lots of times the LARC got to the lab the only program that was really big and ready to run was Chuck Leith's atmospheric...

Miya: Zonal weather have here.

Michael: Yeah, it was zonal flow and stuff like that.

Miya: Drum-mapped isomorphically are in my notes.

Michael: Well, all of that, true. And people were calling it the Leith Atmospheric Research Calculator (LARC). And Chuck is a very, very gifted person, and one of the very rare kinds who's good at mathematics, and can write excellent programs. And he was able to show on the LARC zonal flow, and his weather maps were as if you were standing on top of the North Pole looking down, so you were seeing an equally _____ polar projection of the world, and you could see pressure fields on the thing, moisture fields, temperature fields, wind fields, and put any set of these together, and that was where we started experimenting with color. In other words, he'd write the field that he wanted to be in red as the only thing on that piece of film. And then we'd take that piece of film and put it through a projector with a red filter onto some color film, and then on the same piece of color film project the green information, and the blue information. And it was awesome, I guess is the best way to say it. There as no guarantee that the weather he was calculating had any real life cognate, but it looked very much like what people see every day in the weather. So he started with essentially what's called the primitive equations that'd been written down by Louis Richardson during the first world war or shortly before that, and then he added to it in some of his models. It would be taking in all of the reporting stations that would report, and learning what was a mistake and what wasn't a mistake. And using that to reset the model to a current condition. That turned out to be, as I recall, a causation for enormous noise spikes in the various fields. And it turned out, if you just turned your calculation on and run it, the spike will come out and then dampen out, and then pretty soon say after four or five days of simulation of the weather, the weather is normal, not spiky, and it's stable in the sense that you can run forever without errors defeating you. Again, there's no guarantee that you actually, for instance, if it predicted rain in Livermore at 100 inches a minute, or a 100 inches a year or something, that would be wrong, but it would still fit the general idea that the weather looks this way when you view it far enough away.

Miya: So these were the first weather calculations ever done.

Michael: No, I don't know if they were the first. There were other people who did them at the Institute for Advanced Study, as I recall. Joseph Smagorinsky was the person that Chuck communicated with, and somebody else at the US Weather Bureau. I think Chuck's calculation was probably the first that ran without corrections and without blowing up. It ran, it produced believable weather patterns, and then the only thing that would've made it better was if you could add, in the three dimensional sense, an enormous number of zones.

Miya: So it was a three-dimensional calculation, or a two-dimensional?

Michael: Well, I think it's got to be three in the sense that you have the surface of the earth, and then above each point there're these, in this case, six levels of atmosphere, and you're computing in each one of them. That's three-dimensional.

Miya: Yeah, it is. So knowing that there are six levels is important.

Michael: Oh, I don't know that.

Miya: Oh, I think it is.

Michael: All right. I'm not going to argue about that. The basic thing is the data that's resident in all these cells above the earth is finally mapped onto a pressure field, or a temperature field, or a wind velocity field, or a moisture field, showing you where it's raining, or where it's going to rain. Showing you where it's such and such a temperature. Now all that stuff is related mostly to the surface, but you could sort of ask for it to be related to any altitude you wanted in the last zone in the vertical sense went out to the sun, I suppose is a way to put it.

Miya: So was von Neumann still alive to see this?

Michael: No.

Miya: He passed away when Chuck was getting his first results.

Michael: I think he died in 1957.

Miya: Yeah, '57 or '58.

Michael: And that was before we had the LARC running.

Miya: Now did other people use the LARC besides Chuck?

Michael: Yes, there were lots of people using the LARC doing nuclear design calculations. I don't know specifically about any of them. The LARC arrived at such and such a time in the history of the laboratory that there were alternatives to running on the LARC. You could run on a 7090, you could run on a 1604, you could run on 3600. Eventually you could run on 7094. And all those gave you a lot of what you

would get off the LARC, in terms of accuracy, speed, capacity. After all, 30,000 words on the LARC is the same as 32,000 words on a 7094.

Miya: So it was basically an arbitrary amount of memory.

Michael: Yeah.

Miya: Now before I move to the 6600 here. One was Paul Woodward. I didn't know if Paul Woodward was one of the people who used the LARC. No?

Michael: You what? I don't understand.

Miya: I have two notes underneath the LARC. One was Paul Woodward. Was he a user of the LARC?

Michael: No. Not that I know of.

Miya: And then you had a thing you told me about on Friday afternoon. It was called the Livermore Multi-Megaton Group. You want to say anything about that?

Michael: Now Livermore Megaton Group, that didn't have anything to do with the LARC either.

Miya: Okay.

Michael: It preceded the LARC. All it was then the staff, which was then the theoretical division, and the people who were doing the actual nuclear design calculations would get together, and people would talk about what they're doing, and accept remarks from other people in the media about whether it was good/bad, or needed to be extended or whatever it might be. It was at one of these meetings that Edward Teller proposed that now they've actually been through a test series, that's 1954, it's time to get on with some physics. And he was proposing that each person have a project that he wants to push on, depending on the calculation that these machines will provide, and it just should be scientific, but other than that there's no constraint. So out of that Bernie Alder and Tom Wainwright produced, I don't want to call it an empire, on molecular dynamics. They did the first calculations with the various models of the nuclei banging into each other in a confined space. Out of that it was Chuck Leith and I think it was Gail Marshall produced a thing for the studying of very specialized populations. You know you could take those to mean neurons. What happens to a group of neurons as they go through a life cycle given a certain set of rules? How often will they be allowed to reproduce themselves, and how many in any reproduction will there be, and things of this sort. And in a sense, some of the pictures that would have resulted from that particular study look a lot like Conway's game of Life. It had that certain characteristic. Out of that actually I was able to expand our study of graphics, and graphics hardware. And there were others that were just minor little things, but they were very important at the time in the sense that they hadn't ever been done before, at least that we knew about.

One of the persons, Bob Price, who got to be president of CDC most recently, was fooling around with a random number generator and studying what limitations there are in the various forms of random number generation. This was you take the middle-- two M-- how I put it, two M-length words, and multiply them

together, you get a word that's got-- a number that's got 2M in it. Now if you take the middle half of that out, which is a whole word, that's supposed to be a random number, okay? And you re-square that and continue doing that. And that doesn't-- turns out to be not the best kind of random number of generator there is, but it was one of the first. There was a point which we had these random number calculations that were going to be done. They got a New York telephone book and threw it on the floor, and Harold Brown put his pencil somewhere in the book, okay? It was pointing at a number. That was the first random number. I don't know how true all these things are. These are the stories that were told. <laughs> It was fun.

Miya: I'll move off of the Multi-Megaton Group from the early '50s. We'll move into the '60s now, and we'll get to some things which are more current. So the next machine is the 6600. Tell us all about the 6600.

Michael: Well, Sid and Richard Von Holt, who was the assistant department head, visited Seymour [Cray] before the 6600 existed, and he explained to them what he was planning to do to make a 6600. And there was enough grist in that explanation that when they got back and Von Holt was debriefing himself with the staff, we're talking about you know that this multiplier has been turned on, and how many cycles must it run before it's finished, you know, that this adder has been turned on, how often will you have to restart it with new numbers? So you could get an assembler that would take your code and lay it out showing when you could next make use of the multiplier, next make use of an adder, next make use of the incrementer, all of the various functional units that were in the 6600. And then by clever programming, which is roughly like minimum latency program, you could again hide a great deal of the latency that these arithmetic units imply and run at machine cycle speed or some reasonably small multiple of that. Now after that, people were sent on a regular basis to visit Seymour and see the progress that was being made on the 6600 and then they would come home and talk about how to write programs for it, and how to verify things and so forth. The 6600 had-- its console was a pair of CRTs. And just to make it happen, the programmer would write the program so it would show a sailing ship sailing on the water on one tube, and then it would disappear, and reappear on the other tube.

Miya: At the appropriate speed.

Michael: Yeah.

Miya: Inter CRT speed has to be consistent.

Michael: That was its idle mode thing. I don't know how much it was used, but again, the 6600 had these separate I/O processors and they would report when they were ready to be used, and when they were busy and stuff like that, and you could by clever timing and program and command variation, you could hide a lot of that latency so you could make the I/O process that you're trying to deal with as transparent as possible in the sense that it-- you don't have to wait for it.

Miya: Do you recall how many 6600s were made?

Michael: Well, I think there were over 55 or so. Now I don't believe I ever knew that number, but I may have heard that.

Miya: And how many did the lab have?

Michael: It had four at one point. 6600, it's interesting, with almost all of Seymour's machines he was pushing the memory cycle, okay? In fact, I've heard him identified as the Evel Knievel of memory design. Just pushing out beyond the limit. And as a consequence the memories were flaky. So among other things, Chuck noticed that this was the case. He also noted that he had his program set up so it would repeat a calculation cycle by cycle until the current results agreed with something in the past. Then you said, "Well, that's the stable part of the machine, and we go on from here." He didn't want to tell everybody about how that worked, but the CDC engineers considered it a great, great step in the right direction, because it helped them identify weak things in the machine, and correct them that much easier. So every 6600 that went out the door supposedly got this set of programs that just ran and re-ran until there were two consecutive cycles, calculations, fairly extensive calculations that agreed with each other. So that was-- whether Chuck wanted it or not, that was a benefit that he gave the engineers. I remember when we were trying to get stuff done on the 6600, there was always this guy in the machine room who was in our way, and then I learned it was Seymour. <laughs> When the first machine came to Livermore, Seymour came with it. And he was developing what we call the Chippewa operating system, and we made use of that. It was a very simple, very fast, didn't take much memory. We used the compiler that one of his employees had built. His name was Garner McCrossen [ph?]. He's dead now, unfortunately, but he produced this compiler, a Fortran compiler that was very fast, very clean, it generated good code, and let the user get the maximum benefit of the machine.

Miya: Was this a Fortran?

Michael: Yeah, it was a Fortran.

Miya: And the Chippewa operating system, was this the notorious operating system that Seymour supposedly just did it all in his head and ran first time without any bugs and all that stuff.

Michael: Essentially, yeah.

Miya: It's that operating system?

Michael: Yeah. Anyway, we got to know Seymour pretty well, and he certainly was a remarkable person. But there were various and sundry things that some of the people were trying to develop, sub-routines. Some of the people were trying to develop the very first timesharing operation that was going to run on the 6600. And there were some people trying to develop a thing called the monitor to run on the 6600. The monitor was something that IBM had produced for the 704 and went beyond that. It was a card-oriented thing, or you might say sequential inputs, one process after another. Whereas, the timesharing thing had the question: who can run now, and get it ready while we start the previous process and things like that. I think many of the better features of timesharing systems in general were worked out on that machine. We had a guy by the name of Bob Abbot who was in charge of the group that was producing the timesharing system for the 6600. He had a thing called Gob, which is the Russian word spelled backwards for God, Bog. And it was a very simple machine, simple-- I shouldn't say machine, it was a very simple machine code, okay. And it gave actual timeshared experience to the people who were wanting to run that way. And I suppose that there were other similar examples in other places, but this was the one that was at Livermore.

Miya: Was this going to ultimately become, LTSS, the Livermore Time Sharing System?

Michael: LTSS evolved from this machine.

Miya: From Gob.

Michael: Yeah. Actually there was a thing called CTSS in between. And you could argue [it] was the Compatible Time Sharing System, or Cray Time Sharing System. I don't ever remember anyone coming down authoritatively on either of those interpretations.

Miya: Well, Cray timesharing system was the mid- to later '70s, right?

Michael: Well, with Sid's effort, the timesharing system itself was exported to Los Alamos. They had been insisting and running in monitor mode, one after another, and so forth. And they thought that there were some people who wanted to try the timesharing system. So it was set up for a certain number of hours per day to run. And you know the result. That number of hours expanded and expanded and expanded till the entire day was under timesharing, including the monitors that would run. It's interesting that the Los Alamos' people produced user manuals, and instruction manuals, and programming manuals and things like that, that were far superior to the stuff that Livermore had done. But the Livermore timesharing system was the nucleus of all that stuff that Los Alamos had produced, well, it was first, that's about all you can say.

Miya: I'll get to CTSS and LTSS a little bit later for the software part. But for the 6600, though, so the first operating system was called Gob, Russian for God.

Michael: The first operating system was CHIP.

Miya: Okay, CHIP, then Gob, G-O-B.

Michael: Well, you see timesharing systems didn't really exist anywhere except in those places in the academic community where the machines that they ran on were very small relatively speaking. The timesharing system at Livermore was on this mammoth machine called the 6600. It had a 128K of words. You know, that's unheard of. The only thing that even came close to it was the PDP-6, and PDP-10 from DEC.

Miya: So it was about a factor of four over the other IBM machines there.

Michael: No. Not that I know of.

Miya: How much was it then?

Michael: I can't understand.

Miya: You were saying the IBM machines had 30,000 words of memory.

Michael: 32K, yeah.

Miya: And these machines, the first 6600, had 128K of memory.

Michael: Yeah.

Miya: Words.

Michael: Yeah.

Miya: So that's a factor of four.

Miya: Michael: Yeah.

Miya: That's what I was asking. Hey, I know some of these things. And its cycle time was, as I recall, 100 nanoseconds for a clock cycle.

Michael: Oh. I think it was more microsecond, but I may be wrong.

Miya: Yeah, I think it-- I didn't quite get to that period of time. I only ran on 6400. But I seem to recall 6600 was about...

Michael: One microsecond was a fundamental clock cycle. And the PPUs, the peripheral processing units that did the I/O and moving stuff in and out of memory and so forth, each-- there were ten of them. Each of them took one-tenth of that cycle. So that was 100 nanoseconds just for each computer in the I/O chain, okay? And I'm pretty sure that's right. You know, the way that I had it laid out.

Miya: So Horst Simon and others are waiting on baited breath, and I seem to recall you telling me it's the 6600 in 1964 that's where you guys decided to come up with the million floating point operation as the metric, the megaflop.

Michael: I think it was done before that.

Miya: Do you recall who did it?

Michael: Well, yeah, it was a way of telling the manufacturers what kind of a machine we were going to buy, okay? If it had an instruction execution rate of such-and-such, okay, that you could say this is a flop, a floating point operation, then you'd be talking about megaflops, that's millions of oper-- floating point operations per second, okay? And in a given range, that's a class six machine or a class five. We initially began with, say, class four machines.

Miya: 6600s were class four as I recall.

Michael: That could be, yeah. And when we went out for a bid for the next machine, would say, "Well, we want a class five machine," and that would be the definition that was supplied similar to what I just--now...

Miya: Who came up with this? Was it Sid?

Michael: I think it was Sid, you know, the director of computation. And there was a guy by the name of Bud Wirshing, a guy by the name of Stu Stone, and there may have been others. But these were part of Sid's administrative staff, and they're the ones that pushed this idea. It was the easiest way to get around the wooly crap that the government was putting on procurement. You want to procure a computer, and you don't want to have to deal with those things that're just repeats of what you've got, or even slower. So you just define this class, which is beyond where you are now. Class four is what you have with the 6600. Class five, that's going to be a 7600. Were there other class five machines? Not that we got, okay. There was some likelihood that IBM would attach something to the high end of its system, 360 or 370 machines. They had a model 85, a model 195. These were very, in their own way, fast machines, but they were not at all in the same league as the Seymour machines were. Seymour machines were "easy to use." I'd say that that's not the case for the IBM machines. They weren't bad, they were just hard to use. And you had to go into an entirely new religion called OS. We used to hear frightening stories about OS. Takes 103 hours to compile a system, blah-blah-blah, you know? I know it took an awful long time when I was at Haverford to get the stripped down operating system to run at all. But it was maybe two shifts of a computer time. So, the classes, to get back to that, gave the manufacturers a clean target to look at. And gave us the ability to not look at machines that didn't meet that class. Because we were allowed to say what we want. And I think that it was a good idea that in the sense that practically all other computers, all people, computer procurement groups, adopted the same idea.

Miya: We're at the end of this tape, so I'll take a break.

END OF TAPE 4.2

START OF TAPE 4.3

<Audio begins abruptly>

Miya: --And classes of machines. So we ended with what constituted a class five machine and you said it gave a clean take and you were able to ignore other machines as a result. But let me ask this question, when did you guys start using the term "supercomputer?"

Michael: That's a question that hasn't really any answer. I've always thought that it was Warwickman [ph?] who coined the expression. He's denied it. In fact, he tried to blame it on me. I don't deserve it in this case.

Miya: When did this start to happen?

Michael: After the 6600. Recognize the 6600 was the so-called first fast computer, supercomputer. It wasn't maybe the biggest computer ever built or the earliest or something like that. But there was no question; it had the largest memory. Every design feature in the thing was aimed at speed. I think I told

you; when we first got the 6600 and were running our calculations, they were very, very slow. In looking into the question, it's because the kind of calculation we were doing depended, in a way, it depended on the number zero. But when Seymour produced an operation that yielded a zero in the floating-point field, he didn't zero the entire word. He just let the zero be the fraction and the exponent could be anything you wanted. So testing for zero failed. That meant that our calculations we were trying to do, their slow calculations, further and further into cold material, cold dark material because they don't know how to terminate on zero, this funny kind of zero. It turns out that Seymour doesn't clear this when he got a result zero because it saves a cycle. That's a tenth of a microsecond in those days. That's what you were saving. It was interesting, but the laboratory prevailed on Seymour to fix it so that he cleared the register all the way to zero. The speed up was dramatic. All of a sudden, our calculations were running faster than they had ever run before because now we were terminating on zero, the way they should have. But it just sort of emphasizes how very thorough Seymour had been in having only one goal - that is go as fast as you can. In this case, he was not cheating, but taking a short cut. I think that Professor Kahan in Berkeley has gone after Seymour on several occasions because he says the crazed don't do good floating-point arithmetic. That, again, is this exception handling stuff. Seymour, for instance, rounds before he divides. That's not good. That's a no-no in mathematics. But what can you say? You have to know about those things. Then you can avoid them if it's possible. Mostly, you get the benefit of the speed improvements that he built in. So it seems like a reasonable trade off.

Miya: When did you guys start benchmarking?

Michael: We benchmarked--

Miya: From the beginning?

Michael: From the beginning, yes.

Miya: So even going back to Univac I, you had benchmark codes.

Michael: No, we didn't benchmark Univac I because there was nothing to compare it with.

Miya: But all subsequent machines had unclassified benchmarks.

Michael: No. We didn't have programs that served as benchmarks until we got to essentially the 7000 Series, the 704 and beyond. Up to that point, a machine was just considered okay if it ran the problem you wanted to run. That is, could you fit it into their memory? Remember, the Univac had 1,000-word memory and the 701 had, I think, a 2,000-word memory. Neither of which was quite adequate for the tasks that we had in mind. But as the competition for building computers grew, the way to prove that your computer was going to perform the way we wanted it or needed it was to have it try to calculate your program as a benchmark.

Miya: And these were unclassified benchmarks, right? They weren't classified benchmarks.

Michael: Yes, they were unclassified.

Miya: And these are precursors to Frank McMahon's Livermore Fortran Kernels [LFK].

Michael: No. I think Frank McMahon and Lance Sloan codified the thing. Like they said, "This little fragment of code is something you'll find in the I/O chain of a big calculation. This little fragment of code makes heavy use of the multiplier. This little fragment of code goes after memory in a fairly serious way;" that kind of thing. So they put that together to run one after another and they would compare results. I think Jack Dongarra does the same thing with his linear algebra stuff that runs on every machine in the world. That's a doable thing. Benchmarking is very useful from the architect point of view. It's easy to build a machine that multiplies and adds and divides and steps and things of this sort. But it's not easy to build a machine that does that efficiently and very accurately. That's what the benchmarks have helped to point out. I think without any exception, benchmarking has improved the result of what people have done for their computers. It's giving people, like those at the lab, a leg up on where to start trying to go faster.

Miya: So I seem to recall that the 6600, at least for a megaflop rating was about - what - a three megaflop machine, 3.3 megaflops or something like that?

Michael: Well, it was a megaflop machine. That is, every microsecond, it would issue an instruction. Now these instructions, in many cases, would take longer than one or two or three microseconds to get committed and done. But you didn't see that time because you were buffered. It gets a little fuzzier when you go to the 7600 where the cycle time of the small core memory was - what - 27.5--

Miya: 27.5 nanoseconds.

Michael: --Nanoseconds. So that's awful fast relative to a microsecond, but it's not easy to decide what the megaflop rate is for that unless you say in the citation, "We stayed away from large core memory," or "We stayed away from exceeding small core memory and branching out of bounds" and things in the citation. There are a variety of caveats you have to worry about.

Miya: So the 6600 and the 7600 were core machines. They were not transistor machines for their memories.

Michael: The 6600 had core.

Miya: Right, and the arithmetic units and logical units and the peripheral processors, they were all transistors.

Michael: Yes.

Miya: I'll get to the 7600 later. It was a 60-bit word.

Michael: In the 6600.

Miya: Right.

Michael: It was a 60-bit word.

Miya: Yes, and you guys had a 1604 before that, right?

Michael: Right, and we had a 3600. We had two 3600s two, for that matter.

Miya: Did you only have one 1604?

Michael: Right.

Miya: Okay. Now, here's one of your favorite topics that you yourself wanted me to ask you about on the topic of the 6600 – one's complement arithmetic.

Michael: I can't remember all the details about that. I liked one's complement arithmetic, but I thought that it needed to be augmented with "As a result of any arithmetic operation, if the result is all bits, clear it to zeros." Then there's only one zero and you get the symmetry for everything else, plus and minus the number. From the point of view of fancy display stuff, that's sort of helpful. But it's a minor point in the sense of, "Am I'm going to increase by 10% or 50% the number of instructions I can process because of this?" No. The problem is that zero is an important number in physics. It's also a good basis for measuring other symmetries that are likely to show up on a display. So we had lots of little pretty displays that were on the PDP-1 because it had one's complement arithmetic there. I don't think there's anything numerically significant about that, though.

Miya: So you think maybe students these days are missing the subtleties because they don't see one's complement arithmetic?

Michael: I'm not sure I'm qualified to talk about students today because their educational horizon and their educational basket of things is so much different from what it was when we started. Most of them don't know what machine language is. They don't know what a low-level assembler is. They don't understand the fundamental boolean and logical operations. They're dealing with a level of C or C++ or above. After all, the world's greatest mathematician now is Mathematica, right? It can solve all that stuff that was just blocking everybody in the 19th century. Now, it's just a couple of keystrokes and you're all set to go. Well, the students aren't being cheated on any of that sort of stuff. They're pretty good. But they are certainly being taught different things. In a way, so were we when we started. We didn't have a heritage of computation. We had a heritage of numerical approximation. One of the things that we did then was to combine numerical approximation with whatever it was I was saying.

Miya: Oh, you guys also used Symbolics' Macsyma.

Michael: Macsyma was very heavily used, but that was in the '70s. Well, Mathematica has, from the superficial point of view, the same capabilities that Macsyma had and then some.

Miya: Okay. So you want to say anything more, though, about ones complement arithmetic?

Michael: No, I don't think so. Programmatically , it's easy to use. The one's complement arithmetic is one thing. Two's complement arithmetic is another thing and signed binary notation is a third thing. They all have usefulness in their own way and they all give you the same answer if you're careful. I think that ones just gets us easier display routines to write and there's a certain symmetry where you don't have to worry about two different representations for zero or in the twos complement case, there are more numbers that are positive by one than there is negative. There's one number less in the negative sense, right? Something like that.

Miya: Well, it's typically one more negative number.

Michael: It's one more negative number.

Miya: Typically.

Michael: There I was backwards.

Miya: It's okay. Okay. I seem to recall Seymour made some comment about his use of one's complement, transitioning to two's complement. Something to the effect of he didn't learn it until--

Michael: Oh, these first machines were one's complement and he said that he didn't put the two's complement in because he didn't understand it. Finally when he understood it, he put it in. That's classical Seymour, I suppose.

Miya: I think that's a famous quote too actually, that he did that way. But that's a decade ahead. Okay. One question we need to elaborate on is the social history at the time, which was the fact that in the world that was going on when the 6600 was delivered, the national laboratories had to stop above ground testing. Livermore-Los Alamos had to stop above ground testing.

Michael: Livermore and Los Alamos stopped above ground testing initially in 1958. Then, because of the presumed deterioration in the international scene, they had a further test series above ground out at Christmas Island in 1962. Then more recently, there was a below ground test called the Canican [ph?] shot up in Alaska. Now, when you can't test above ground, you have to learn how to interpret that data you're getting from the underground tests. It's not clean. It's not clear how to do that, but they've been fairly successful. I think that the technicians who run that stuff are true entrepreneurs _____. They're really very good. I'm thinking particularly of Bob Breedy [ph?]. But, you know, they have come up with schemes for getting a maximum amount of data out of the hole before it gets destroyed. We have incomplete models of what it is to have an explosion in an enclosed space. They get the same good data out of an above ground test or below ground test. There's a whole class of data that you can get whether it's above or below. It shouldn't make any difference. But the larger effects, obviously, have some difference - if you're going to go off in the atmosphere versus you're going to off in a rocky hole. That's the way it goes.

Miya: But the importance to computation to this in design?

Michael: Obviously.

Miya: I'm trying to get this out of you, George. You're talking about stuff in the past, that you're able to do Carson's [ph?] caution, that you're able to calculate before testing and the like. You test everything.

Michael: I think that the experiences that we've all had lead you to feel that in addition to experimental physics and theoretical physics, there is this physics which is proved by computation. We call it computational physics. It's different. There is no experiment you can conduct that'll show you the effects of an uncertainty constant. If it's change by a factor of ten to the 20^{th}, what's it going to do to your view of nature? But you can show that in a very good way in a model that's run in a computer. So the value, from the point of view of the experimenter or the researcher, is that he gets to let his imagination run a little freer and he gets certain answers that don't do violence with what he knows about the laws of physics or mathematics or chemistry. Yet, he learns more about the actual way things behave in nature by seeing these extreme examples. So you can learn that way. I don't agree that it's bad.

Miya: No, no. You wanted me to tease this out of you as to the arrival of the 6600 at the time when all above ground testing had to be put down into the ground. So I'm expecting you to say what its influence was, the 6600, on this change of vertical--

Michael: Where you test.

Miya: Yes.

Michael: I don't think I had ever thought about that before.

Miya: I think you did. We talked about this before. You thought it was a good thing, that you basically were able to refine thermonuclear design because of the 6600.

Michael: As I said a few minutes ago, the data you can take about the dynamics of an explosion are probably the same whether it's above or below ground. The thing that's different is that below ground, you're in an enclosure and that changes the dynamics, the macro-dynamics of the explosion different from above. There are two things that I think we see. One is establishing scaling law. So you know roughly how to evaluate something that's otherwise not available. The other thing is you're not polluting the air and all that other stuff.

Miya: But given their druthers, most physicists will want above ground testing, wouldn't they?

Michael: No, I don't think so. Not at all.

Miya: Even back then because really, Teller wanted above ground testing.

Michael: I don't think so. I don't think he cared.

Miya: I think he cared because he had a big argument with Linus Pauling and the public about it. That's how Linus got his--

Michael: Well, that's not the basis of the argument, though. The basis of the argument is whether he had test or not. Linus Pauling was not, he was against testing.

Miya: He was against surface testing. His big thing was he was against fallout I remember from biology.

Michael: He was against testing and the lesser of two evils is to--

Miya: --Test underground.

George Michael: Go below ground. But a test is polluting no matter whether it's above or below ground. Teller wanted to test. Pauling didn't. It's okay. I think that both had valid positions to defend.

Miya: Well, you held his gold medallion for peace. That was given to him for testing underground, forcing testing underground.

Michael: No. That was given because he was trying to establish-- He was doing something that leads to peace, not because he wanted to test underground. I don't believe that anyway.

Miya: So have we beat the horse to death with regard to computing and the shift in testing?

Michael: I don't know how to answer you, you know. I think I'll just pass it. Yes.

Miya: Okay. The next machine you wanted me to get to, which I don't know anything about, was the 6800, which I'm guessing was a dual processor 6600.

Michael: No.

Miya: 6800 then.

Michael: It wasn't a dual processor anything. It was just an extension of the 6600 using new components. It wasn't necessarily compatible with the 6600 at all just as the 7600 wasn't compatible with the 6600. The 6800 was a new kind of packaging, used new components and you'd get the advantages of these various things when you're running it as a processor. Seymour never liked the results that he was getting. So the 6800 never saw the light of day. There was no delivered machine until the 7600 and then between it and the Cray I, there was a 7800. The same thing is true for that one too. It was a complete repackaging and redesign with the new technology, with higher performing memory chips and logic chips and so forth.

Miya: Okay. So going back briefly to the 6600 from the 6800, at this point in time, LTSS wasn't in existence and LRLTRAN was in existence.

Michael: LRLTRAN and LTSS are roughly the same thing. They were changing names to emphasize the different services that are available. LTSS was more aimed at networking than LTSS.

Miya: No, I asked LRLTRAN, language and compiler. Excuse me - language and operating system. So the LTSS operating system was, by now, in existence. LRLTRAN compiler was in existence. That's a question.

Michael: Yes. Well, for the most part, the stuff was homegrown again. In a sense, the logical mother of that stuff was Bob, but then people went on and improved things by virtue of having learned a bunch of stuff and by virtue of maturing and getting to test it in different ways. The fact that it was going to be written in LRLTRAN is only incidental, I'd say. Yes, it's being written in a higher level language. That means you can move that operating system to another machine, providing you have the right translator.

Miya: Okay. CHIP disappeared. So CHIP was no more. Did you guys talk to the other people who had the 6600s for their operating systems?

Michael: Well, I think there were two or three places that people wanted to talk about that. One was at NYU. They built a thing called PUMA, which was a 6600 kind of thing. There was one at UCLA and Los Alamos. But I think everybody was independent and went on their way irrespective of what was decided.

Miya: Yes. So you guys didn't look at SCOPE or Mace or any of these other operating systems.

Michael: No, not that I know of.

Miya: Okay. You guys didn't consider any of the less expensive versions of the 6600, like the 6500 or the 6400.

Michael: As far as I know, there was none of that done.

Miya: Okay. Let's go on to the 7600 then.

Michael: Well, what can you say about the 7600?

Miya: They're faster.

Michael: It had, again, a re-designed memory architecture. There were two memories. One was small core array 65K or 64K or whatever it is and one was large core array. That's 512K.

Miya: I recall SCM and LCM.

Michael: Who?

Miya: SCM for small core memory and LCM for large core memory.

Michael: Well, I agree.

Miya: What was the LCM size? 512K, was it?

Michael: It was 512 interlace? Oh, I don't know about that.

Miya: No, size - 512 words.

Michael: Oh, 512,000. I've always heard it said it's 500,000 words _____.

Miya: So 65K and 512K.

Michael: Yes. You can certainly call it that.

Miya: Yes, I never ran on one, but I've talked to people at NCAR who had one. Okay. So did it have a cache? People want to know, does this have a cache? No cache.

Michael: Not that I know of.

Miya: Seymour didn't believe in caches.

Michael: Then. Maybe later on he did. I don't know. There are really two things. We're getting into a later part of the whole adventure and we're getting older. So memory problems are coming up here for one thing. For another thing is that people were drifting apart. It's just impossible to maintain this intense communication over a long period of time. So whether Seymour had caches or not in his first machines of the Cray line, I'd say no. But did he contemplate them? I don't know.

Miya: How many 7600s did the lab have? Four max?

Michael: I think we had five.

Miya: Do you know how many they made roughly?

Michael: No. The 7600 in its own way was the longest-lived computer at that point that we had. We had it longer than anything else and people objected to when it was taken away because since it had no development going on it, it was fairly stable and they could run their little physics problems just time after time after time to test the results and stuff like that. The 7600, from the purest point of view, makes it a bit difficult to understand what is the best way to perceive to levels of memory. When you cycle the large core array, you don't get just one word. You get eight words - 480 bits. When you cycle the small core array, you get one word - 64. I don't know where this comes down. I was going to say something, but I can't remember what I was going to say.

Miya: That's okay. I was listening to your poignant pause there.

Michael: Oh, the two-level memory. It was a large problem for some people because they didn't know how to manage the mismatch in times and how to cover the large core because it gave you eight words at

a time. There were some people who decided to run their programs out of the large core array and just leave the data in the small core array. That worked pretty well. But overall, I think the judgment is that having two levels of memory is not an easy thing for a programmer to overcome.

Miya: Did you guys talk about virtual memory at all?

Michael: I don't remember any real conversations on that, Gene.

Miya: Because IBM was certainly going that way.

Michael: Well, yes, of course.

Miya: You guys didn't pooh-pooh virtual memory like Seymour did?

Michael: IBM has several people working on any imaginable facet of every problem in the world. They're a big research institute.

Miya: Right.

Michael: They have some very good people.

Miya: Did you have any other 7600 software besides LTSS and the Fortran, the LRLTRAN compiler that you want to say anything about for the 7600?

Michael: I was thinking.

Miya: Okay. Other software on the 7600 of note besides LRLTRAN and LTSS.

Michael: Well, there was a compiler called CHIP.

Miya: A CHIP compiler separate from CHIP _____.

Michael: Yes, there was a great deal of competition. The timesharing system maintains tables that tells whose doing what to whom. So it was always interesting to notice that though the official compiler was LRLTRAN or something like that, the most heavily used compiler was CHIP. That had been optimized, as I recall, by Steve Jasse [ph?] who was then working for Control Data, I think. It was a clean thing. It had good documentation and people who used it, liked it. The same thing is true for any of the official word processing software and stuff like that. The overwhelming majority of people at the lab were using TRIX, TRIX AC, Red PP [ph?]. Those are all the same basic thing. It's a pattern matching kind of software that was very, very effective. It was maybe even more effective because your results could be flashed up on a TV screen very quickly. You didn't have to wait for a typewriter to bang it out or anything like that. Well, there's just no question. I think TRIX and TRIX AC are superior to any of the stuff that's been shown in other places. Since it was local stuff, it's being allowed to die.

Miya: In the period of the 7600, do you want to say anything about any algorithms that were in use at the time?

Michael: No. I don't know of any hot things that developed. We just didn't have any of that.

Miya: We have two more minutes.

Michael: I think that in general, we are at the end of the Golden Age.

Miya: The 7600.

Michael: With the 7600, yes. The thing that came in next was the Star and that was a tough experience, I guess, is the way to put it.

Miya: I have five things before the STAR-100 I want to ask you about. Let's go ahead and change the tape. Are we ready?

Michael: Yes.

END OF TAPE 4.3

START OF TAPE 4.4

Miya: 7600 was the end of the golden era of super computing.

Michael: That's what I think, yeah.

Miya: So what about, you said the 6800, which is the repackaging of the 7600?

Michael: No, I'm saying that after the 6600, Seymour designed his next machine, and it was called the 6800, and he didn't like the way it was coming out, so he did not produce it. He went on and produced the 7600 instead.

Miya: Okay. And then after the 7600 he was going to repackage that as well to the 7800?

Michael: He was producing his next machine which was called the 7800, and he didn't like it. So he didn't produce it. But he did produce the next machine which was called the Cray-1

Miya: Now, what about the 8600, which apparently Gordon Bell has a photo of the 8600? It was supposed to be four processor...?

Michael: I think I've said something wrong previously.

Miya: Okay.

Michael: The remarks I was making were about the 8600. The 7800 I don't remember.

Miya: Okay. So we will scratch the 7800 and we'll note the 8600, which he didn't like, right?

Michael: Yeah. He didn't like that. And we saw the circuit boards that he wanted to use for that 8600, and the transistors looked like little drops of solder. I don't know where those boards went.

Miya: Gordon is the only person we know that has photos of the 8600, which looked very Cray-2 like.

Michael: Fine.

Miya: You don't remember anything about the software or anything like that?

Michael: Of the 8600? No. Again, if you notice that no two machines that Seymour produced were really compatible, instruction compatible. And that's true for the 8600 as well. He was pushing an established technology to be better then they thought they could be, I guess is the way to put it.

Miya: And CDC was okay with this up to this point, I take?

Michael: Yeah, I suppose. I think he voted with his feet so to say.

Miya: And the lab had no interest in the machines like the Cyber-73, or the Cyber-173 and those mainframe-ish follow-ons?

Michael: Well they were all with the Star as the parents. And see, which the only thing you could say about the Star was that at the point at which it appeared, the people at the laboratory did not understand vector processing to the extent that they have to in order to exploit that machine. So, no one knew how to write codes for these machines, and you could write codes at the higher level language level, but that didn't get you the efficiency that people wanted. So, that's about all you can say for the Cyber line, okay?

Miya: The lesser Cyber line. Well we'll get to the Star in a moment. Okay, so three remaining quick things. When did the Octopus system come about? Was Octopus an LTSS? I saw this horrible diagram by Dick Watson in 1978 where everything was connected to everything before hyper-channel came in.

Michael: Yeah. Well, the Octopus is a proposal that Norman Hardy and I produced, which was-- we didn't call it timesharing then, we call it resource sharing. And it was to say that when you weren't able to use something somebody else would be able to and we've got to fix it up so that that person could. And that's certainly in the era of 1960 to '62. And by the time the 6600 got there in '64, our ideas were fairly solidified and Sid had agreed that people could spend time developing a timesharing system for the 6600. The original idea, I know this sounds weird now, but the original idea was that we would have a disk which had a lot of heads on it, and each computer had a set of read heads that would read anywhere, but it could only write in a certain spot. And the idea is that if a computer wants to tell another compute about

something, it writes it interval place and then tells a machine like the PDP-1 "I have sent this message to Joe over here." So then the PDP-1 will alert Joe to go and look in this location and see the data, or the message, or whatever it might be. This bid disk was inspired by a rumor that we had picked up that-- I think it was Burroughs, but I'm not real sure. Burroughs had an army proposal for a disc with 7700 plus heads on the thing that would have done nicely with what we had in mind. Remember at that point there were no FTPs or any of that kind of stuff here for you to send data from machine to another. The way you did it was to write it on a tape and carry the tape to the other machine. So here we are trying to invent this way and this was called the Octopus, and this disk that sat in the middle with all the machines clustered around it that could read anywhere, but only write in restricted areas so that you couldn't write all over you, or I couldn't write all over you.

There was another aspect of it that was called the archival store, and we were after an invention that had been proposed by Art Halt [ph?] at the Rabinow Engineering Company in Washington. At that point Rabinow was just Jacob Rabinow, and he was probably the most successful engineering patent guy that ever existed. What Art Halt proposed was that you have these little tapes just like we use in our tape recorders, and they could read or write at any speed. And if they're sprocketed there would be no problem, we could do that. And so again, you have a machine like the PDP-1 controlling this array of tapes, which he called the basement tape system, and it was a thousand tapes. We estimated that would be more memory then they would need, but it would be easy to handle, would hold 10 to the 11^{th} bits, an enormous amount of bits. And, well I don't know, the PDP-1 would get a request for some data, and it knew where the data was and it knew where it was, where the tape was, so it knew how far it had to spin up or down to the data to get to it. That speeding up or slowing down would have nothing to do with reading and writing, you could read and write at any of the speeds. The only thing is, you know, it was not very fast to begin with, and then it would catch up to speed. And then the data would be transferred the way it ordinarily would be transferred. Well we went to talk to talk to Art Halt in Washington, and it turns he was talking about 'vapor' [i.e. non-existent] inventions. What he was proposing to build for us was a quarter of what he had said initially, and it would be at a quarter of the speed that was proposed initially, and it would cost four times as much money. So that was abandoned, and I say the first incarnation of the Octopus had only IBM compatible tapes that could be moved from one place to another, and as I said we found that if you had the inter-record gap enlarged then that tape can be read either on a control data tape handler, or an IBM tape handler. I think that the IBM tapes were better, just as I think that the Control Data _____. I've forgotten what I was going to say, Gene.

Miya: You were going to say something about the inter-record gap with the IBM control data tapes.

Michael: Well, that allowed you to read the tapes on either of the tape handlers. The IBM tape handlers were essentially better then the ones that CDC was building, though the opposite was true for disks. CDC disks seemed to be better. Anyway, we worked out a proposal, which we called the Octopus, and it was at least in two parts. That is, the software which was ultimately called Gob would handle the resource that was on the 6600. It didn't have anything unusual or mavericky or anything like that, it would just handle that stuff. And we went out to bid, and low and behold we got a thing called the photostore in 1969. Actually we got it '67, and it took two years to fully integrate it into a real data handling system, not just saving stuff. So that was one of the things, and there were other things like a high-speed printer that was needed, there were things like terminals that were needed, and we initially picked the IBM data typewriter, but they turned out to be very expensive and they weren't very reliable. So we ended up with ASR-33s [a model of Teletype], which was a new kind of thing then, Teletype, you know? And this mishmash of stuff then is going to somehow be integrated into an operating system. The operating

system will decide who's ready to run and start the one who's running next, and get the next person in line ready, and just handle the outstanding I/Os, and the printing loads and stuff like that, and then turn the machine over to the next user.

Miya: Let me read some of that back in my notes, okay? So you said that Octopus consisted of two parts, right? The first part was the software, Gob, the operating system which handled the resources on the 6600 bit for the photostore.

Michael: Yeah.

Miya: And then the other part was everything else, which are the printers, the terminals and the like. Is that correct?

Michael: Well I don't think you're getting that the right way, but that certainly is the division. Gob was the operating system. Under the operating system were all the things you find on a computer, compliers, assemblers, file managers, debuggers. Okay?

Miya: Uh-huh. And the other stuff was hardware, the resources.

Michael: Well the hardware resource...

Miya: Printers, terminals...

Michael: You know, from the terminal line in, the terminal, the printers, the remote card readers, remote card line printers, all that stuff grew like wild mushrooms, and ended up under the Gob operating system. But by the time all of it was really done, Gob was no longer really there, you know, it was something else. Call it what you want to. There was a group of people under Hans Bruijnes who insisted that the operating system on this Octopus network had to be LRLTRAN. So they had a bit project under way where they were rewriting the Gob system in Fortran, in LRLTRAN if you will, so it could be run on any machine and so forth. It was okay. I don't think it was mortally deadly or anything like that, you know.

Miya: Now for the hardware resources, which were CPUs, like 7600s and 6600s, you guys didn't name those, you guys just simply designated them with letters. Is that correct? A, B, C, D, is that right?

Michael: Well, I don't think that was very official, but yeah I suppose that true.

Miya: They didn't have names or anything like that.

Michael: The A machine, the B machine. Yeah. That had something to do with the sequence in which these machines arrived at the lab.

Miya: Right. But there was no concept of a host name like we have now.

Michael: No I don't think so.

Miya: Okay. Two more things before the STAR-100. You mentioned both of them. Do you want to say anything specifically about the Radiation printer or the terabit store?

Michael: Well, I don't know. Radiation printer was our reaction to the fact that we had to have a lot of printouts, and on demand you might say. Up to that point we were struggling with line printers on each of the machines. These were 150 lines a minute, terribly inadequate. We had some film-to-paper capability so things would be written on film and then put through a blow-back system onto Xerographic paper and the paper was a big Z-fold from a reel, and that went a page a second when it worked. And when it didn't work, it was an automatic soaker for fire. The page would go into the oven to be fixed, and the paper would break. So the oven would catch fire, and you're putting more paper in all the time. As I recall the guys name is Roger Door [ph?], he was with Litton at the time, he was a former lab guy and we were over the one day and he was showing us that using Xerography to tone the paper, but using electrostatics to mark the paper initially he was able to print it at the rate of, well I guess the simplest way is to say almost nine pages a second. That's more then one page a second. And it turns out that that's the magical area which we had to get into in order to have those systems built. So we put out a bid.

The Radiation Incorporated from Melbourne, Florida, proposed this big behemoth printer. It was really a production printer from a newspaper or something like that, but they were modifying it by putting in instead of a print head, there was a set of styli that were arranged so they would arc across the paper when they were energized, and that way you could put little dots in a shape that would recognize the letter A, or you recognized C, or Z or whatever it might be. So the Radiation printer printed that way, it would burn a hole in this paper, and the paper was a piece of carbon web, black, with this smeared out white paint on the surface that really didn't come out white, it looked gray, and the process was called revelatory, or a revelation printer. This was because, you know, you've got all this gunk there and now this styli burnt a hole where they're needed, and now you have graphics. Fine. Radiation printer performed at seven pages a second. It really wasn't seven pages actually because when it's not printing, it's moving across paper that's blank. When it's moving across paper that's blank two things may happen, it may punch binding holes in the paper, and it may also score the paper so you can separate it at that point. And that's what was done.

The Radiation paper was this carbon web with this paint on it that made it look white or green. The rolls of the paper weighed 300 pounds apiece. They had a machine that would go out and pick up a roll of paper and the operator would bring it. The paper would go through the thing and it would print essentially 60,500 characters a second. 30,000 lines a minute, there's a variety of ways of saying it, and it would fanfold the paper and punch binding holes in the paper, and do a scoring of the paper so that the Z-folds were laid flat and so forth. As the paper came out of the printing head, it would be fan-folded and sit on a moving belt that pulled it along until the operator got around to separating things. When the printer worked this scoring, you could tell it where you wanted to separate this piece of paper from this piece of paper, that's a job, that's a job, and you could tell it that and it did it. If you do the calculation that says how much of the paper area you're moving over where there's nothing happening, and subtract it from the total height, which is eight or seven pages. I think it's 11 inches high, just like an ordinary piece of paper. So it comes out to be just over six pages a second, but we found in usage that the Radiation printer could be down and then we'd turn it back on and let it print until it had taken up all the back log, it kept up with the printing load, and it's heyday the Radiation printer was accounting for something like three million pages a month. And the paper had a funny odor to it because the styli were burning through the paint that's on the surface. Now to keep the styli clean, they would blow ground up walnut shells against the styli and that would keep them clean of the paint residue and stuff like that, but it left people with headaches and so forth, so what the lab did was to supply those who complained with a little fan that

would just sit on your desk and blow air at your face when you were trying to read the stuff. The paper was exceptionally heavy even though it was thin.

Miya: So here's a question about the printer and the paper. Were there other Radiation printers around, or was the lab's the only one?

Michael: The Radiation printer was unique, just one.

Miya: So the reason why I ask this is if you read James Bamford's *Codebreakers*, when it talks about the NSA they talk about having to destroy gobs and gobs, millions of pages of classified documents. Did you guys have to do that a little too?

Michael: There were regular bins for classified paper that would be put in there. I must admit I don't know how they destroyed this Radiation paper. It wasn't ordinary paper like the kind that's sitting in front of you there, it was a black belt with a thin coat of white paint that turned out to be grayish. And we had along the side of the paper a marking which was put on when the paper was printed which said "Protect this restricted data." And that was one classification marking, and if you didn't think your stuff was classified, there was a nearby knife which would cut off that red border. If the paper had more sensitive data on it, it would have printed its warning when it was going through the printing press, but that paper couldn't be thrown in a garbage can, it had to be going into a separate container that did nothing but accumulate paper to be destroyed.

Miya: So you don't know the ultimate disposition of that paper then? Somebody took care of it.

Michael: Yeah, as far as I know.

Miya: Lots and lots.

Michael: Well to give you some data, when the Radiation printer started working, it quickly built up to about three million pages a month. So ordinary line printers accounted for maybe one or two hundred thousand lines per month. They couldn't even come close to it. The film which we introduced as a substitute for the Radiation printer for those who wanted it was producing 300 million lines per month. Now those are only ballpark estimates. There are documents that have a more accurate rendering on them. But there was a big disparity. The Radiation printer came up to a level of say three million, and it stayed there. When the microfiche capability was added, people found that they could put everything that they wanted on this-- we had 48X microfiche. And then they would only have to print certain things, and if they couldn't avoid it any other way, they'd get it out on this Radiation printer paper. Well Radiation there doesn't have anything to do with nuclear activities or anything like that.

Miya: Right. It's just the name of the company.

Michael: Name of a company in Melbourne, Florida.

Miya: So once story you told me when we had lunch at the lab once, in the south cafeteria, was about some guy, some physicist who self-taught himself Fortran, and he did statement numbers and he numbered all of his statement numbers starting with a one. Remember that?

Michael: Yeah.

Miya: And what happened?

Michael: Well that one was inside Fortran format, and it was the belt that controlled the printers recognized that one as a page restore. So he would get one line per page, and he would have literally thousands of pages. The operators eventually threw him off the machine because he was just using paper up, and he never bothered to correct it. You know, it didn't bother him that there was only one line per page instead of 50.

Miya: So that's being self-taught, right?

Michael: Well I don't think it's so much self-taught as he's interested in other things. He was interested in relativity, I know I've talked a little about it.

Miya: Can you name the scientist who did this?

Michael: Well, I was trying to remember his name. I think it was Jim Faulkner [ph?]. I don't know. He was very early in the labs history in the 50s, and that was when Harold Davis, a visiting mathematician, was at the lab. And Harold once told me that at the time, this is in the 1950s, he'd been publishing for over 40 years, and his average was one page of scientific text per day over the 40 years. Now he was instrumental in a series of lectures which theoretical division studied some of the interesting cosmological stuff, and the question was "What is angular momentum in a universe?" Does the universe have angular momentum? And some of the ways to get a hold of that turned out from them to be dealing with general relativity. And Jim Faulkner, if that's his name, was a collaborator of Harold Davis and they were talking about some very esoteric aspects of general relativity. And these numbers that he was producing related to that. But as I say, he was more interested in the numbers then the fact that there was only one per page, it didn't bother him at all. I think instead of throwing him off the machine, it would have been better to just take a minute and explain to him what the one does when he puts it where it is, then there's no problem.

Miya: But he was self-taught as I recall what you said.

Michael: Everyone was self-taught. All there was was a manual. Incidentally, the first Fortran manuals produced by IBM were superb. You could learn from them quite a bit. But I don't think he was self-taught by studying those manuals. He may have heard a lecture or something like that. How do you write a format statement?

Miya: I thought that it was basically a lunch conversation kind of talk.

Michael: I don't think so.

Miya: All right. Let's do one more peripheral unless you want to add anything else about Fortran and format statements and Faulkner.

Michael: Photostore?

Miya: We'll go to the photostore next. But anything else about printing and format statements and Faulkner?

Michael: Well, I said the Radiation printer had a ten-year lifespan, which is quite good, and it would print maybe three million pages a month. It was only a third or less then a third of what the photographic stuff was on microfiche. We had a bunch of FR-80s that were dedicated to producing microfiche, 48X microfiche. And we used to produce on the average a million chips of microfiche, as I recall, a million per year, and each one would have on the average 120 pages or so. So those numbers are not going to be consistent because I haven't looked at those numbers in 50 years I suppose, and I can't remember what they are. But in any event, when we introduced microfiche that's where the bulk of the new printing, and the radiation printer just stayed more or less constant at what it was, it didn't grow anymore. Up that point it had been.

Miya: Okay. Do you want to say a few words about the photostore and we'll get to the STAR-100?

Michael: Which is first?

Miya: The terabit photostore.

Michael: Okay. I'll go back and refer to the basement tape system again, and when we learned that that couldn't be delivered to us the way they had promised, we went out and talked to people in the industry about what was available. Itech [ph?] had a process where it would develop without having-- it was photographic, and it would develop without having to go through photographic processing of hypo and bleach and all this stuff that they do. So a spec was written and Itech bid it, and so did IBM. IBM proposed the photostore, and that's what we eventually got and I think that was one of the smarter moves that Sid insisted on. That is to say we went with a guy who had a big shovel in case he had trouble, and also had very good mechanical skills available when he needed them, and also we're dealing with a very stable technology. So the photostore had as its parents the Walnut file and the Cypress file, which is IBM's analog stuff I think, in one case. That it went to all the NASA drawings for the Jupiter rocket, or whatever it was, would be on these strips of film, and if you've ever looked at the IBM data cell, that was the basis for the photostore. The only difference is that the photostore chip was essentially 1.2 inches by 2.3 inches or so, it had 32 fields on it, each field held a certain number of bits, and the recording technology was something that had been developed for a translation project in IBM Kingston, it was called Manchester, which says that we're going to have a zero represented by a black and a clear area. We're going to have a one represented by a clear area and then a black area. So that the average is always going to be half of that. And this stuff was written on these chips which were made for us especially by Kodak because they had to survive in a vacuum where they were being printed on by an electron gun, and they had to go through an eight step automatic processor which processed the film, and the film was put in little boxes that could be blown around like the big vacuum systems that were in old department stores. So you can imagine that a little chip box would be selected from a chip store and it would be blown into a pipe, and blown up to a read station where some fingers would open it up, pull out the piece of film that you wanted, and get you positioned to where you wanted it, and read it. The chips

were superb in the sense of clarity. They were really sharp and they maintained an 8.1 by 8.0 size of a tooth. And if this tooth was attached to a black stripe and it would go back and forth like a snake, and this was called boustrophedonic recording.

Miya: How do you spell boustrophedonic?

Michael: B, A, U, S, T, R, O, P, H, E, D, R, O, N, I, C, or something like that.[correct spelling is, as above, boustrophedonic. –Ed.] That's enough to get you in a dictionary. Well the technology is that the recordings look like the cornice moldings on these old classical buildings, with the teeth sticking down. The boustrophedonic made that the teeth were on both sides of that black stripe, and that the reading beam would read-- it was a flying spot camera, could read these teeth and it would swing and come back reading on the bottom of the reading teeth, and just continue that way.

Miya: You have one minute, wrap it up.

Michael: Well I don't think I can wrap up the photostore.

Miya: That's fine.

END OF TAPE 4.4

START OF TAPE 4.5

Miya: STAR-100.

Michael: No, let's talk about-- finish the photostore.

Miya: Oh, okay.

Michael: The photostore, the way we got it, had a 30 percent burden on it in the sense that where we said it was 10 to the 12 bits, it was actually 1.3 times 10 to the 12 bits. The extra 30 percent was for redundancy and error control and addressing controls and stuff like that. The photostore was a very successful device at the lab, you know. It gave everybody a reliable place to store stuff and the only thing that I'm amused [by] was when we put the photostore online, it gave the lab more online memory than the rest of the world put together.

Miya: Even more than the NSA?

Michael: I think that's classified. <laughs> And now, today, we have people like Brewster Kahle who's banging against the petabytes, bytes, that's eight times 10 to the 15 byte--bits. And-- where we're talking here about one times 10 to the 12 bits. But the photostore looked awfully big at the time. Over the course of its 10-year lifetime, we filled it essentially 11 times. That's just [an] approximation, okay? Certainly wasn't more than 11. And the only quote anecdote I have about the photostore is that along towards its-- the end of its life, it became quite erratic and was, I mean, lots of trouble. And the trouble

was traced to a worn out spot in the pneumatic tube where the little chip boxes would go in order to go from the reader to back to the chip store. So IBM ordered a new part and it was put in but it didn't correct anything. So some very careful measurements were made by the engineers and they learned that this new part is made of a different plastic than what the original parts were made with. And they changed the coefficient of friction enough so that the time for the chip to move from the, say, chip store to the write or read station varied. And the chip store itself was designed for dead reckoning in a sense that it had to be in a certain interval otherwise it would say it's in trouble. So if the chip box was there a little bit later than it should have been, even though it got there, it's an error and it would shut down. IBM had a, I think it was a 1401 but a very much modified 1401, that worked as a supporter for the photostore and kept statistics on errors and the number of chips that were made, so forth and so on. We used to call it the 'photosnore' but even so when it was up working, it was a very, very good piece of equipment.

One other thing, the blank film chips were moved one in a box, which ordinarily holds 32, but they'd just have one in a box, moved into a vacuum chamber where fingers pull the lid off the box and pull out the piece of film and hold it in front of a flying spot electronic beam writer. The electronic beam writer will write with this two micron-thick snake and little teeth of 8 microns by 8.1 microns wide sticking out of that snake, all the way around. And to avoid having to go to err too often to replace cathodes and stuff like that, the IBM engineers built the electronic beam recorder to hold 16 guns. And after a certain number of exposures, they would rotate the turret to a new gun and so that it would have--

Miya: Even the wear out.

Michael: Yeah. Only problem is one day the corrosive chemicals that were used for developing ate through the metal separator between the developer and the writer and short circuited the power supply. So we had a time when it was very smoky and foul smelling. But, again--

<crew talk>

Miya: Okay, the STAR-100.

Michael: Yeah.

Miya: I never ran into a STAR-100. Tell me about the STAR-100.

Michael: Well, it's-

Miya: You had two.

Michael: We had two.

Miya: How many were made?

Michael: Maybe three or four.

Miya: Three or four. I think one was at [NASA's] Langley Research Center.

Michael: One was at David Taylor, one was at Cappel [ph] Nose [ph] Atomic Power Laboratory.

Miya: 64--

Michael: And I don't know where else _____.

Miya: A 64-bit post Seymour Cray machine

Michael: 64-bits, yeah. Well, the STAR is an interesting thing. Remember a guy by the name of Dan Slotnik?

Miya: Right, an award is named after him.

Michael: Okay. He came up with what we now call SIMD [Single Instruction Multiple Data] architecture and it showed up in the SOLOMON [vector supercomputer] which was being put together by Westinghouse Aerospace in Baltimore. So we thought that would be interesting and I remember Norman [Hardy] specifically went back there to the Westinghouse place to show them what floating-point arithmetic was like and to see if that could be added to the-- these little simple processors. And they thought they could do it. So but since we're now in this stage where you have to everything done by bid, a bid was required for a thing. And among the people who responded beyond Westinghouse was Control Data and IBM. We chose the Control Data thing and in the long and the short, it was the thing that was described in this book by Ken Iverson in the first two chapters of-- a program language is what it was called.

Miya: APL, yeah.

Michael: But anyway, instead of having an SIMD structure, the STAR turned out to be a vector machine. And in no case did the lab or anybody else then have experience with that.

Miya: The Cray-1 did not exist.

Michael: Cray-1 did not exist, no.

Miya: It was a long-vector machine, 65,000 elements.

Michael: Well, it could be that long but the better way to see it is that it was memory-to-memory and, yes, you did control the length of the vector. But it was a memory-to-memory movement. In the case of Seymour, it was memory-to-register movement and you were limited to 64 there. All right, so the STAR was delivered, not delivered, but it was spec'd and there was an awful lot of inner-processor meetings, you know, that, "How do you write programs for the STAR? And what will Control Data provide? What will the lab provide?" and so forth. Well, the STAR attempted some things that had never been attempted before.

Miya: What was the cycle time?

Michael: I don't remember but it was a hundred nanoseconds? I can't remember.

Miya: I can't remember either. It wasn't-- so it wasn't faster than 27-and-a-half nanoseconds?

Michael: No.

Miya: Slower.

Michael: It doesn't get its speed from a given memory cycle. It gets its speed because it's able to pre-fetch stuff before you need it. That's what the vector idea was. And as a consequence, you know, you-- if you-- once you paid the start-up cost which was several hundred machine cycles then from then on, things go through at an enormous rate. And the problems in the STAR were perhaps best characterized by the disparity in speed between this vector processor and the scalar processor. You can imagine that if you're calculating a, I don't know, a radiator or whatever it might be, and you get to the point where you've been through all the zones and the mesh, now you want to calculate a new dovity [ph], stability can _____. But you don't need a thousand of those. You only need one, right. So this is a scalar calculation and it's very slow. Very slow. Given that, that was the thing that determined the overall performance of the STAR, not the vector but the slowness. What is the slowest component? That tells you how fast the machine's going to be.

Miya: How big were the memories?

Michael: On the first STAR, a million words. And the second STAR was two million words. But remember I said, nobody had any real experience with that. Now one of the great flamboyant guys of the era, Neil Lincoln, who was project engineer on this thing, I guess, he characterized the machine by saying, "It is 90 percent of the first two chapters in *A Programming Language* by Ken Iverson." It would have been better if they'd done it on a 99.99 percent but in any event, it wasn't complete and it was, well, very foreign to people who had been accustomed to writing programs that are so declarative such as you find in a 7094 or 6600.

Miya: Actually, I think most people would have called it not declarative but imperative. All right, you know, do this, do that. Yeah.

Michael: Good point.

Miya: But it ran--

Michael: The STAR--

Miya: The STAR--

Michael: -- would run fast--

Miya: No, the languages and the operating system?

Michael: The operating system was a kind of a derivative of Gob. And it was being-- it was not a-- STAR was not at all a good timesharing machine. But it did this.

Miya: It ran LTSS?

Michael: Yeah.

Miya: Because the [Cyber-]205 ultimately ran something called VSOS, I think.

Michael: Well, we never got a 205. So I'm not sure about what to say about it but--

Miya: So it-- basically it ran LTSS, though.

Michael: Or something equivalent to it, you know, to run on the STAR is a different kind of exposure than it is on an ordinary computer. This vector thing is very serious and there was one unclassified code, I think it's called Hemp, which is a form of continual mechanics. It's a hundred percent vectorized meaning that every instruction in that program is a vector instruction. Okay, it runs fairly fast. We have a limited number of applications for such a program but those will certainly be shared very quickly. However, you have a guy like, the guy who's doing the astrophysics on magneto-hydrodynamics. He says, "We spent seven years trying to learn how to vectorize for that machine and in that time, we never did any physics anymore. We're never going to do that again." So the STAR came in to a place that had no experience with vector processing. It came in when we were really better off with anything but a STAR for computers and it had very little support, both from the user community and high-level support from the lab community. And we ended up with two of them. I'm not sure how accurate this is but the story I hear is that Los Alamos was getting one also. And when they learned how difficult it was to make it perform, they canceled their order and switched to 7600s with the idea that they would get whatever they could, you know, when it was available after the STAR. In other words they were aiming at Cray-1's at that point.

Miya: Do you recall, regarding Los Alamos, how many 6600s and 7600s they had besides not getting the STAR-100?

Michael: No, I don't know.

Miya: But they at least got at least one of each, right?

Michael: Oh, yeah. They had maybe even, you know, more computers than we do. I don't know. We did. But in this case, these machines were acquired because the controller of the DOE wanted to have a bulk purchase for the entire spread of 10-DOE laboratories, you know, on one contract and therefore saving economies of scale. So like the Savannah River plant got an IBM [System/370 Model]195 and somebody else got this, somebody else got that. The two laboratories, the weapons laboratories, were getting each a STAR. But when Los Alamos discovered that the STAR wasn't completed and it would be difficult to program it, they threw up their hands and said, "No, no. That's not us." They said, "We're staying with this thing until we get some other machines." At that point, you could argue that if that

contract cancellation had been allowed to stand, Control Data would have been out of business. You know, you could argue it. I don't know if it's really true or not but the fact is that Sid chose to take the second STAR. Now that was good news for Control Data and it may have been--

Miya: Good news for Control Data. May have been--

Michael: Say again.

Miya: You said the second STAR was good news for Control Data. Were you going to say that Sid was burned because of that?

Michael: Well, I think they-- the project could stay fluid and financially safe. Whereas if that half of it had been canceled when the STAR was having trouble being constructed, that might have been enough to kill it. So it may have been good for the nation to keep a thing like Control Data in business but it was a bad thing to have two STARs at the lab. Number one, people are exhausted at the lab now and they don't understand how to do the STAR to begin with. But beyond that, they're now being asked under pressure to make them perform really wild. Well, the fall out in Washington was bad. The lab got not blacklisted by they said, "Look, you have these world-fastest computers there. You can't have a Cray-1. You can't have this. You can't have that. You've got this world-fastest stuff." And, of course, that wasn't true but it didn't make any difference. Well, if you look to see what the STAR would be good for, I think it's pretty safe to say you can imagine, say, a line-scanning satellite reading transfers to the direction of travel crossing a 200-mile swatch-- swings that just keeps moving and doing that. And if it's radiating that data down, the STAR could eat that up right away and do the image processing, whatever else they want to do on the thing, on the fly, you might say, and that would fully occupy the STAR would be a very smart application. What we found was that generally the equations of mathematical physics are not compatible so much with the STAR architecture. And to learn how to get the compilers to find parallelism or vectorism, whatever you want to call it, in, you know, the mathematical statement of a physics problem, is a big deal. The lab had the beginnings of a good compiler that would do just what I've said. That compiler was exported to Japan and to some place in Georgia where Steve--

Miya: Wallach.

Michael: -- Wallach is. Now they both started with a lab project of this smart compiler for a vector computer and went on to develop a good compiler for vectorization. So now you could be less cautious about how you write stuff and the compiler will figure it out at least better than it was. Again, the equation of mathematical physics that the people in the lab are interested in doesn't do you any good. That is, that's not the stuff they're going to do at the lab.

Miya: I seem to recall, since we're talking about APL, George, that one of the big problems was the order of evaluation from left to right versus right to left and APL reverses that. Is that right? Was that one of the problems?

Michael: No, no, no. It's certainly possible. There's so many things that get in the way. I can't tell. The STAR wasn't a model of reliability. I remember Neil came to one of the Salishan meetings, one of the early ones, and he heard Ernie Pleketti [ph], our guy, talking about Monte Carlo neutronics. And he said, if he had known that before, the STAR would have come out differently. So it really was a good machine

in its way maybe. But it didn't have a problem set or a section of problems that-- where people were interested in the problem who were willing to try and put them _____.

Miya: You don't recall what Ernie said that influenced Neil?

Michael: Oh, well, I recall in principle. You start with a random number and you say, "That's a neutron now." It goes some place. It's going to interact. Will it produce another neutron or more than one neutron? What are the directions at which it's traveling? Everything about that thing is a random number. The direction cosines have to be random numbers. The probability coefficient is a random number. The number of particles that are produced in the _____ a random number. It-- just go on. The idea is that I think the simplest way to put it is that Neil would have put more juice into the scalar [mode] or made it possible for the scalar [mode] to be much more simple than it was. And to have several of them so you can sort of daisy-chain them. You know, I'm guessing. I don't really know what's the best way for-- to rebuild the STAR, to take advantage of the actual architecture of a big physics pro_____.

Miya: So the Star didn't even use the 7600 or the 8600 as its scalar processor?

Michael: I can't answer it. I don't know.

Miya: Yeah. So we will-- it's going to be one of the obscure portions of history that we're not going to ever know about.

Michael: Oh, I don't know if that's the case.

Miya: Or we should go interview Neil and get his--

Michael: There are guys both-- that were at CDC and guys at the lab who spent a great deal of time trying to understand how to program the STAR. The problem is that you don't-- if there's a fundamental design flaw, the scalar system wasn't right for this particular vector process. You don't want to learn how to fix that. You want to learn how to write programs for that. You want to learn how to fix it maybe but not writing program for it. It's not going to do any good.

Miya: Yeah. Well, certainly the [Cyber-]205 and the 203 had their advocates and they hated to see long vector machines go.

Michael: Those machines, 205 et cetera, were, you know, corrections of the STAR-100 in all cases. But it doesn't change the-- well, I think as an observation it doesn't change the-- what kind of problems that can run. Well, if it does, are the kind of problems that it can run going to be interesting enough to people so they'll pay to have it done or do it?

Miya: And you guys decided against ever buying 203's, 205's or ETA-10's.

Michael: That's quite true. It's-- I think you could say that by the time we would have been in a position to buy a 203 or a 205 or a 202, there were other things that would be less expensive and have just as much bang for the buck. You know, it's sort of like saying, "Well, we know now how to build the 709." But

I don't need that built that anymore. I got this other thing, you know, called a Macintosh. Or I got this other thing called a Sun _____. I don't need to go back and _____ other thing. The same is true for the STAR. The STAR may have had a computational window, I think it maybe missed, but I'm trying to tell you that at least from my point of view, the thing that made the STAR difficult to use was it had sort of flaws in the way the vectorization went, where the-- how did the scalarization go. Those things were incompatible. Now-- and I also gave you an example of this line processing, you know, the--

Miya: Right, the scanning.

Michael: That's a thing that the STAR would ____terate, okay. The problem is it's not the only machine that will do that. Now the question is it the cheapest machine and the answer is no, it's not. I mean, the STAR is a multi-million dollar machine. I think it missed its window. It's the same as Burton's multi-threaded architecture?

Miya: Oh, MTA.

Michael: Yeah, I think he's missed the boat on that, too.

Miya: So do you want to call it quits? Are you done with the STAR or do you want to talk about Q8 calls?

Michael: Well, go on. Tell me what's next.

Miya: Well, Q8 calls. The next thing we'll do is talk about the Cray-1 which we might as well start next time. So it's a matter of wrapping up the STAR-100. I only used it very briefly. I remember the Q8 system calls and similar. But maybe you guys didn't have these Q8 calls on STAR-100 for operating with vectors.

Michael: By searching through the literature and algorithms and stuff like that, we identified a thing called basher sort as something that would naturally fit well on a STAR. We identified the thing, and I was trying to describe before as a thing that would run well on a STAR, the satellite thing. You know, in more mundane applications it could be probably very good for processing chest x-rays and stuff like that. All that stuff is useful but the question is when a thing is useful, it also has to be paid for. And apparently that's not, I mean, there are other ways to do it. You can't find today a person that wants to build a high-speed machine because the machine that sits on your desk or your laptop is--

Miya: More powerful.

Michael: Blazing fast, yeah. It's-- when it tries to do I/O, it goes out to lunch. True. So do all these other machines. That's what I think is the next barrier to machine usage. We have to learn how to use the machines that they supply with special kinds of, oh, what am I trying to think of that--

Miya: Special calls, special instructions.

Michael: No, I mean, you know, with the note and the algorithms that are being talked about now, there are simpler machines and it's very rare to find a good application that's almost exclusive to the STAR. There's nothing about the STAR that's cheap. Given that, it's not likely you're going to see any applications being made to that. Our guy, Mark Wilkins, developed his thing, the Hemp code which is, well, he calls it continual mechanics. And it's a fully vectorized thing so if he's confronted with a STAR, that's how he can use it.

Miya: But Hemp was not a production code. Hemp was a research code.

Michael: Well, it's production _____. Production in that sense. Everything was researched. <clears throat> That's what made the lab a lot of fun, you know, research. Free to try all kinds of weird things.

Miya: Yeah. Well, do you want to wrap it up? You have ten minutes of tape left. They're getting ready-- they want to get out of here.

Michael: It's up to you.

Miya: Well, I'm-- yeah, I'm okay ending it right now.

Michael: All right, whatever you say.

Miya: And then we can let them go and then when we resume, we have to figure out when next.

Michael: Next.

Miya: We can start with the Cray-1. I'll ask you about DMOS.

END OF SESSION 4

START OF SESSION 5, July 25, 2006

Miya: Okay where we last left of George, you had finished the Radiation printer, the terabit store.

Michael: <Inaudible>.

Miya: The Radiation printer, the terabit store, next time remember to bring your things and you were talking about.

Michael: We talked about those things.

Miya: Yeah we were talking about--- you had mentioned just a moment ago the Golden Age of computing, that grand old time right so.

Michael: I think you can actually quantify what this.

Miya: Oh what's the quantification?

Michael: Well we have some problems that have never been approached before, when you have some very talented people who could work on them, when there's sufficient funds to get the work done, and when management stays out of the way. Largely they weren't really organized yet so they could not impose a lot of structure on you. You also found that the people using the machines were being helped in lots of ways by the people who were building the machines. But as with everything else that people do, after a while, they impose structure okay, a manager's got to have committees and they have the taxes and all this other stuff and the people who were really talented and worked on the problems drift off into management or go off to form another company, which also imposes structure and a number of problems that you could do quickly diminishes. So the Golden Age ends.

One of the most important things I think to begin with was for the first time with the help of a computer, you could explore alternate realities okay and know that they were probably calculated and things of this sort. The only explanation for that is that when you read it like the book *Mr. Tompkins in Wonderland* by George Gamow, here he poses a question what happens if the speed of light is say 10 centimeters a second instead of 10 to the 10^{th} centimeters per second? What happens if Planck's uncertainty number is one instead of 10 to the -26^{th}? Well with the computer we can explore those things, that's assuming that the laws of physics don't otherwise change and they didn't. Well a machine made it much easier to do that, it was a very exciting time, we were getting results that fit the reality that we have plus getting these delicious insights into that stuff with the help of finding alternate measures for the various constants.

Miya: You mean a different unit?

Michael: No, different constants. There was a theory that said, you know, all the constants that describe the physical universe were otherwise tied together, they just didn't happen and all of them lead to the one thing that you could have many formed. Yeah I'm sure you read that thing. Well that's a pretty interesting idea but what we were doing is varying these constants and testing the assertions that various laws made how good were they and learning about the limits about computational accuracy and stability. All those things made for an exciting time. And it's sorry that sort of changed now, it may still be exciting but it's got to be exciting for somebody else, not me and the things I think that are turning people on now are not really new ideas and all that sort of stuff, it's just seemed to be more interested in engineering details then in great steps in the theory.

Miya: Well that's because as you and I have discussed in the past when you've mentioned that users were helped by the builders of the machines, you basically mean end physics users as opposed to the consumer. The builders of machines went from dealing with discrete components to integrated circuits and that's when you guys lost your power, because you guys failed to appreciate the complexities and the costs of the infrastructure which went up non-linearly.

Michael: Well I don't know. You're touching on a thing that's, I think is still an open question.

Miya: I'm trying to yank your chain.

Michael: Have we achieved, have we arrived at a level of complexity that we can't cope with?

Miya: Nope, we're not saying that.

Michael: I think that's a question you--- what I thought you were arguing.

Miya: No, I'm arguing that when we stop being able to use soldering irons and dealing with discrete components easily and then we had to have machines help build our machines, the end users got further distanced from the problem of what it took to actually put machines together and then we became contract monitors in large part and the people who put the infrastructure, the fab lines for instance together to design and build machines had to go to lighter, larger, consumer markets rather than specialized niche markets in order to appeal to their machines, which we'll get to in a moment. So here let's wrap up the STAR-100, you said a bunch of things about the STAR-100 in terms of its hardware, it had long vector pipes, memory-to-memory.

Michael: Memory-to-memory.

Miya: Right, 65k words, roughly actually powers of 2.

Michael: No the STAR-100, the serial number 1 had a million word memory, serial number 2 had 2 million words of memory and both machines though, the basic design didn't fear or didn't balance the computer well enough so that the vectors were much, much faster then scalars. None of us, none of the people who were trying to learn how to do this, were expert at anything but it turns out that all the computations that were interesting to do, had portions of them where there were scalar computations and that meant that the machine slowed down the scalar speed. Therefore if you can't have a very fast scalar, the machine is slower then you'd expect and that's sort of what happened. When we started, the idea of what is a vector processor was pretty hazy okay and some of the programmers demonstrated saying a do loop is the equivalent of a vector and we can learn how to unfold do loops, unwind do loops so we can make these vectors. But when you get out of the vector or the do loop, you have a piece of scalar computation and if it's too slow, you're slow. You could assume that the speed of the vector processor was essentially infinite, like it didn't take any time at all and it wouldn't change the overall perception of how fast a machine was, very much.

Miya: What was the speed of the scalar processor?

Michael: A 1 microsecond machine.

Miya: So it was actually slower than a 6600?

Michael: As I remember it, people were saying that the scalar processor on the STAR was probably effectively at half the speed of the 6600. I think that number varies a little bit but it gives you an idea.

Miya: So CDC made a step backwards?

Michael: I can't hear.

Miya: CDC made a step backwards?

Michael: I can't speak for them, I don't know.

Miya: Okay, can you say anything about the software that was running on it, was it just _____ and _____?

Michael: There wasn't any real software but there was the attempt in fact, they put the timesharing system on our LTSS and of course it wasn't perfectly efficient because a great number of the things that happened in LTSS are scalar computations and you can't vectorize questions like who's ready to run next, how is this I/O working, things to that sort.

Miya: I thought you guys were super programmers?

Michael: That's not the answer, super programmer or not, if the basic question doesn't have an answer, that's the way it is.

Miya: We ended up with the--- these are very hazy numbers now but on the order of 300 design programs were in use at Livermore in that period of time when the STAR was there and about somewhere between 6 and 12 of those programs turned out to be reasonable applications for a STAR, okay, and that's not a very good ratio I guess you'd say. So the STAR's bigger importance in a sense was that it ushered in the idea of vectorization and it stimulated the development of really effective compilers that could do the vectorizing and it didn't cost you in the sense of accuracy or stability. So that's the way it was. It turns out that a lot of the mathematical physics that people wanted to do is vectorize the raw material, okay? Not everything and we're going to do Monte Carlo style calculations, not necessarily the best thing to is to vectorize, okay? It raised the questions about pollution of the _____ and number seed and things of this sort and those occupied people who were working on the STAR.

Miya: So you told me about some of those Monte Carlo calculations, were they the main calculations of the 6 to 12 that you refer to?

Michael: No, Monte Carlo calculations were used to do neutronics mostly, there may have been other things, I couldn't speak for everybody there, you could do almost all the physics Monte Carlo style but there's no advantage to that okay. The big time burners that made use of the vectorization and so forth <inaudible> equations, learning how to express those as vectors, was an interesting experiment. We had a couple of the experiments where it'll be translated to modern times or to current times, you don't write programs so much as you just state the equations that you're trying to solve and, as given the system like Macsyma or Mathematica the translator, will put those equations into forms that can be solved on a computer, okay? So the initial work for that kind of stuff was done in connection with the STAR. The guy said "I must have written this particular set of equations a hundred times by now, I'm tired of it." So the stimulation for that was enough, somebody would sit down and realize if there's a set of equations can be solved mathematically to give you a Fortran program with boundary conditions and all that sort of thing and that's what was done. Mike Wirth got his--- he wrote a thesis on that basis.

Miya: I saw him yesterday; he said to say hi to you.

Michael: Sorry?

Miya: I saw him yesterday; he said to say hi to you.

Michael: Oh you saw him?

Miya: Yeah he was at this IBM thing.

Michael: He's at Adobe isn't he?

Miya: Right he's at Adobe right now, that's what his badge said.

Michael: But he's a great person.

Miya: I know that, I told him he could come to dinner tonight with us. But last time I think when my power ran out from my taking notes I think you forgot to mention or you briefly mentioned the role of APL and that and that maybe the order of evaluation of right to left rather than left to right may have done APL in.

Michael: I don't think I can add anything to the other arguments that were there, that just was one of those things, APL was a stimulation for the STAR okay, it provided the most convenient initial notation to write programs for the STAR. But it wasn't comfortable enough for the people who had real problems to solve and most of the people wanted to stay with Fortran, which is what they did. So while APL was a definite candidate for writing problem programs, most of them were done in Fortran.

Miya: So that was an installed base thing or that was entrenched mentality?

Michael: The last word?

Miya: Entrenched mentality.

Michael: No I think that remember the principal thing is that a job has to be done and there are two things about that, you have to get it down and to be sure that it's more or less accurate and you have to be able to show that it gives reasonable results and both those things meant that you have to stay in a familiar territory and that was Fortran for most people. I'm not saying it was better or anything like that, I'm just saying most people chose it.

Miya: Well were people at Livermore crucified for APL?

Michael: I don't think it's a Livermore question, I think it's a much more general question, if APL is so good, where is it now relative to a Fortran or C++ or any of the other things, it's not a main line or mainstream language.

Miya: Runs on IBM mainframes.

Michael: It's very succinct in notation to the point where after you've written something and it came back to look at it a month later or so, you couldn't understand what you'd done.

Miya: Write once, read never. Well the interesting thing I think from our prior discussions before this sort of interrogation or interview was that you were explaining to me in the Monte Carlo and Fortran aspect that in order to get rid of IF statements or minimize IF statement processing, you guys came up with mask vectors which determined where in an array of the elements that would get operated on versus not get operated on and that was a way of removing IF statements from loops.

Michael: I don't know enough about that to comment I think.

Miya: There was that thing that Frank McMahon did.

Michael: He and Lance Sloane and before him it was a guy by the name of Fred Andrews, the person who came up with the idea of stack loops to begin with was Chuck Leith and working with Dick McLane and Fred Andrews, they produced the macros that turned the 6600 into this 4 operator kind of thing okay and Lance Sloane and Frank McMahon took that stuff over and extended it to other machines. They wrote reports too which was unusual and I think I've given one of those reports to the Museum. The hazards of living in the [San Francisco] Bay Area, you get these coughs.

Miya: I think that's an excuse personally but.

Michael: Well go on.

Miya: Okay well I'm trying to tease out of you the ideas of how codes got vectorized and Dave Cook is the guy who normally deserves credit for most of this sort of stuff.

Michael: There was, Dave Cook was one of the several people who was involved in this kind of effort okay, there was a group, I think they were at Convex that were also involved. There was a Japanese at Futjitsu who were doing this stuff.

Miya: Yeah but that's the 80's, we're still in the 60's and 70's.

Michael: And there was the Fortran group at the lab that was working on this sort of stuff and all of those efforts looked rather similar and you could argue are you going to do mathematics as a right way, a most efficient way, whatever. I think that Seymour gave us the extra tools to, which the STAR didn't have that I think the Cray for instance had a set of performance registers on it okay and I remember there was a guy by the name of Eugene Miya who was working on that.

Miya: That's the [Cray] X-MP, that's actually Steve Chen but he's not the one.

Michael: <Inaudible>?

Miya: Yeah that's the X-MP and that's also Steve Chen who was one of [John] Cocke's students.

Michael: Well, properly used those things could teach a researcher what to do and how to do it, that's very useful stuff. It wasn't carried on unfortunately.

Miya: So what was the ultimate fate of the STAR-100s and the people who worked on them?

Michael: The STAR-100 ended up being able to run about somewhere, maybe a half a dozen to a dozen programs that occupied it all the time, these are the ones that vectorized very well. As I said of the 300 design programs that were in use, this is just an estimate, the 6 to 12 programs worked fine. Of those 6 to 12, maybe a half of them were really important in the design loop okay and the rest of the time the other programs didn't really contribute that much so they ran well. One of the things that STAR could have done very well is to process satellite data but there were other ways to do that better so that didn't get done either.

Miya: So you guys are processing satellite data at the labs?

Michael: Well, you know, the satellite camera's looking at it one line at a time as you move forward, looking sideways and all this stuff. Those lines can be fed as a continuous vector to the STAR processor and the process result goes back into memory with nothing in the way and the STAR would eat that stuff alive, okay?

Miya: Well Livermore's mostly known for neutronics, it's normally not thought of as doing space stuff right?

Michael: I don't know, I can't say.

Miya: Well can you say anything about the 6 to 12 codes besides they solved PDEs [Partial Differential Equations] or maybe they did a little bit of <inaudible>?

Michael: Well I can describe one of them roughly, continual mechanics, okay?

Miya: You can't give a name for it is an example?

Michael: No but it's not classified, continual mechanics, this was the work of Mark Wilkins and _____ and they had for instance a hyper velocity kind of thing which NASA was interested in, they were interested in how to build a space suit so that it would be able to survive micro meteorite hits and things like that. So Mark's calculation would allow a particle say to come from somewhere and enter this zone region and you could track out where the shocks were, where the materials were mixed and things of this sort and it ran very fast on the STAR. He had a 100% vectorized problem, they made movies of it, showing a bullet coming into the material. That particular kind of calculation was not heavily in the mainstream of nuclear design, okay? It just was a very elegant application of classical physics.

Miya: So is this where the programs DYNA2D and DYNA3D got developed?

Michael: No I don't think there's any real connection, some of the mathematics may be roughly the same. I think DYNA3D at least was a contribution down by the engineering groups at the lab and it used a thing called Nastran or it was like Nastran and it's value took, you know, from the Mark Wilkins kind of work, you could treat a material zoned and the thing could break apart leaving voids and things of this _____ and the computations could still proceed. It's a mechanical problem in a sense that suppose you have a plate and you shock the plate, now when the shock moves back and forth through the plate; when it verifies at it's air surface it may throw off material, that's called spalation and you want to keep track of it, it's zoned material but it's not physically connected with the main plug anymore so being able to do that was one of the great contributions of Mark and Toke and that's what they did.

Miya: Mark and who?

Michael: Mark Wilkins.

Miya: Okay, Mark Wilkins

Michael: And Toke _____.

Miya: Okay I don't know, are there any other programs I should ask you about from that era?

Michael: Why don't we go on from, you know, let's leave the STAR.

Miya: That's what I want to do.

Michael: Okay.

Miya: But there's no other programs I should ask you about from that era though?

Michael: Well, in what respect?

Miya: I don't know, DYNA2D and DYNA3D are the only things that come to the top of my head, minus one or two codes that we're not supposed to use code words for.

Michael: They may use the finite element methods more than finite difference methods.

Miya: Right, that's what they're famous for.

Michael: And learning how to compute a finite element was an important thing and it had great value.

Miya: See I told you, you should have brought your oxygen.

Michael: It had nothing to do with this. DYNA2D and DYNA3D had great value in the crash program studies that the automotive industry was doing and in the movement of space debris or spaceships and stuff like that or it would crinkly up metal and have some hope that your computation was close to physical reality and that's. It was very heavily oriented toward Fortran, it still is as far as I know. I never paid too much attention to it.

Miya: When did the last STAR-100 leave the lab roughly, 1982 or so?

Michael: That sounds reasonable.

Miya: But you don't know where they went after that?

Michael: No.

Miya: Go home and get an artifact.

Michael: I made the suggestion then that one be saved for a museum thing but I don't know what happened to them.

Miya: That's unfortunate.

Michael: See by that time we were passed the Golden Age, son.

Miya: Okay.

Michael: And given that you couldn't expect anything, new managers just, you know, said junk, we'll throw it away.

Miya: Now was the Cray-1 still part of the Golden Age?

Michael: Yes.

Miya: Okay so lets go to the Cray-1 then. You'll like the Cray-1, Dag wants to hear about the Cray-1. We're doing a Cray-1 thingy in September for Dag.

Michael: Yeah they're here.

Miya: Yeah.

Michael: Well, I think I'm maybe a little too close.

Miya: Why didn't you guys _____ the first Cray-1?

Michael: Why didn't we?

Miya: Were you guys not good enough for the first Cray-1?

Michael: That's about right in my opinion.

Miya: I think that by the time that the machine was running around, the people in Washington were more involved with the actual distribution of these kinds of things okay and they said 'well, look you have two things, you have this thing called the S-1 which is the world's fastest computer, which was a surprise to all the people in computation and they also had--- we have this 2 STARs, you don't need a Cray and we have to give it to Los Alamos because Los Alamos didn't even get a STAR while they were waiting for the Cray to be built. So that's where the first model went, it was called the Thermal Model; you could actually Cray a half instead of Cray-1. It moved around, it was in Brackno, England, with ECMWF and I think it went to Germany and I think it went to Monterey, all these people were testing out the idea that Cray-1 represented and well while he was arranging to build real Cray-1's people were able to use this Cray a half. The machine itself again as was typical with Seymour, it was not really compatible with any of his previous work okay but nobody cared because it went nicely, it went fast. Seymour was beginning to complain though that the people who were manufacturing components weren't manufacturing everything with improvements across the board. Some things were more improved then others. Logic would go very fast and it was being improved but memory cycling wasn't. So he had this problem of fetching and storing operands from a slow memory and then they're going to look into very fast logic and arithmetic and that very much annoyed him. In the case of the Cray, Seymour kind of joked saying that he had transferred from wherever he was to becoming an expert plumber in order to build the Cray-1, now is it the Cray-1 or the Cray-2, I don't remember.

Miya: I thought Freon was first used in the 6600.

Michael: Yes, but the Fluorinert.

Miya: Fluorinert was in the Cray-2.

Michael: Cray-2. One of the other things he said was that it helped to define the kind of technicians that he had to hire, the Cray-1 because they had to fit in a little hole in the middle of the Cray and if they didn't they couldn't work on the wiring and stuff like that.

Miya: Yes, Dennis Ritchie just told me about that and it made the difference between two women working on it in the middle or one, that's when Bell Labs was buying their Cray.

Michael: You can't do it.

Miya: You can't do it, that's right.

Michael: But I think some other remarks that he made was Cray-1 is where he first introduced two's complement arithmetic and he said the reason he did that [was] because he understood it. He held off

not using two's complement arithmetic through all of his other machines because he didn't understand it, he said.

Miya: Do you remember roughly how many Cray-1's are delivered or built?

Michael: To Los Alamos?

Miya: No world wide before they went to X-MP's and 2's?

Michael: Well how many, I don't know.

Miya: Do you remember ever visiting Cray Laboratories in Boulder?

Michael: Yes.

Miya: Who was there besides Dick Farley?

Michael: The Cray Labs in Boulder had essentially nothing to do with the building of the Cray-1, they were building a Cray-2 that was radically, not radically, but significantly different from the Cray-2 that Seymour built and Seymour didn't like that.

Miya: This was the Gallium Arsenide Cray-2?

Michael: It was beyond that, I mean they had air cooling instead of using Fluorinert and they were using electron beams to ask some of the circuits and Seymour didn't like that either. But that's where I met Howard Davidson and I hired him to come to the Lab and the other guy was George Stuart Patterson, who was the president of Cray Labs in Boulder and he had collected an incredible list of talented people who were working at Cray Laboratories. It was kind of a re-expression of the Golden Age so to say <inaudible> that happened during the Golden Age but eventually Seymour ordered it disassembled and he thought that most of the people would come to his new place, which I guess was still Chippewa [Falls, Wisconsin] there and only one person went. Many of the other people clustered out in the Bay Area here but I can't say that I know where any of them are now. The Cray Labs is sort of an untold story, I tried to get Patterson to write about it and he won't do it. I've mentioned it to Howard Davidson one time and he's not interested. So I'm not quite sure how we'll find out but it's an important story, part of Cray, an important historical artifact of Cray.

Miya: But that's still the Cray-2 though, that's not the Cray-1, we should still stick to the Cray-1 before moving to the Cray-2 and that way I can.

Michael: I wasn't moving to the Cray-2, I was just talking about they started off wherever and they built an extension of the Cray-1 and Seymour built an extension of the Cray-1, he called it a Cray-2.

Miya: So there's two different Cray-2's?

Michael: There were two versions of the Cray-2.

Miya: Yeah, yeah, so lets stick on the Cray-1 first?

Michael: Alright.

Miya: So you guys moved or was it Los Alamos who moved LRLTRAN and the Cray operating system or I should say CTSS, the Cray Time Sharing System, independent of the Cray Operating System?

Michael: CTSS was our thing, LTSS.

Miya: And it came from LTSS right.

Michael: And it was set up down at Los Alamos because Sid had the influence that he had and he arranged to have it run at Los Alamos okay and initially they didn't--- the administration down there didn't like it and so it was constrained to just a few hours a day. But the more the people used it, the better they liked and so pretty soon it took over the entire machine.

Miya: So the machine basically was delivered bare metal, no operating system at all?

Michael: That's essentially true yeah.

Miya: Didn't even have the Cray operating system COS?

Michael: Well the Cray operating system was rooted in CTSS okay.

Miya: It was?

Michael: Not CTSS, LTSS okay, it was rooted in that thing and the Cray Corporation software people contributed to that program, you know, it was the operating system for the Cray and it was excellent because it was more or less stable and it didn't crash too often and it gave everybody a good.

END OF TAPE 5.1

START OF TAPE 5.2

Michael: I wanted to just make a note that the documentation that was almost produced, superb, was superb, especially compared to the stuff we had produced at the lab. Our documentation was not nearly as good or as clean or as cohesive as theirs.

Miya: Does the Museum have copies? Should we be pursuing this documentation?

Michael: Say again.

Miya: Should the <useum pursue this documentation, or do we have copies already?

Michael: You have whatever they have given you. I do not know if Los Alamos provided a copy of all their documentation. But it is more than likely that they could, because they are an orderly kind of people down there.

Miya: Well, I do not know. The question ultimately is.

Michael: Well, they were undergoing some fairly traumatic experiences there. They had some very good computer scientists there, and they went ahead and produced their operating system, the DEIMOS.

Miya: Yes. I wanted to ask you about that.

Michael: DEIMOS.

Miya: Yeah.

Michael: And DEIMOS was relation to LTSS like IBM's operating system, o.k. It is very slow. It took a long time to understand it and things of this sort.

Miya: I heard it was UNIX-like. Is it a message passing [system]?

Michael: Well, all that stuff at UNIX. Look. UNIX is just a big joke as far as that is concerned. What it is. What is important about UNIX? I will answer my own question. It is the user interface that is important. The rest of the UNIX is childish? LTSS is vastly superior in structure to UNIX. But the people who produced LTSS are grim in relation to the UNIX people, and there is no such thing as Awk on a LTSS. There was no such thing as a vi editor on LTSS. Those things I put on, but they were not documented very well comparatively. The user interface is the thing that made UNIX work well. And the reason it got the chance to, was that initially they gave UNIX away to everybody who wanted it. See? LTSS and NLTSS are both vastly superior. This is a trivial thing. You can be running I think maybe five separate processes on a LTSS or NLTSS, whereas you can run just one, while you are running one, on the UNIX.

Miya: No.

Michael: Yeah.

Miya: So you side with Paul Woodward then?

Michael: No. I do not. I have not discussed this with Paul.

Miya: Yes, you did, three years ago at Salishan.

Michael: I did not discuss it with him.

Miya: Paul used the “F” word many times.

Michael: Well, that could be, you know. And I remember Paul cut his teeth at the lab.

Miya: Yeah.

Michael: Well, that is inexperience. And if you are a clever person like Paul is, that is going to leave a mark on you. It will set your style.

Miya: So the problem unfortunately for the Museum is it is largely going to be getting documentation from other CRAY sites, so it is largely going to get the Cray operating system, which seems more like the UNIVAC operating system, EXEC8. And I suspect its lineage probably carried over to Scope and NOS, is my guess.

Michael: I think that is pre-judging stuff, but I do not think you ought to bother with it.

Miya: No. I think the problem is the fact the Museum does not have very much CTSS documentation, or CIVIC, the vectorizing version of the LRLTRAN compiler, right?

Michael: The people who used LRLTRAN, not LRLTRAN, but LTSS, the operating system, there were a large number of them all over the country, and they liked it. But I will tell you again, the documentation for LTSS was inferior to the documentation that Los Alamos produced under CTSS.

Miya: No, no. I do not disagree. I think the big problem is the fact that the audience, those people you are talking about, the audience, is the physics community. This is why the San Diego Supercomputer Center and the University of Illinois attempted to foist CTSS on the rest of the world, and we ended up having a UNIX war. But that is the CRAY-2.

Michael: You know, if you do not have the right kind of talent to do that kind of work, the stuff you get will not be very good, whether it is in theory the right way or not.

Miya: So why didn’t you guys move LTSS, LRLTRAN to the VAX if your environment was so superior?

Michael: I told you. There were two things that were.

Miya: What two things?

Michael: Well, actually at that time, one of them is the Golden Age of computing was over. That is enormous amount more meddling by managers and people who did not know much about computing. And they kept getting in the way. Second thing is, and exactly, the money to do this kind of stuff was drying up. And the second thing is the people who could do the work effectively and in a reasonably quick amount of time were tired, old, retired, whatever you want to say. The Golden Age is over.

Miya: By the Cray-1.

Michael: Yeah. The Cray-1 was the end of the Golden Age.

Miya: The Cray-1 was the end of the Golden Age.

Michael: I do not know what else you could say, you know.

Miya: Well, I am trying to tease this stuff out of you. So were there other memories about using short vectors versus the longer vectors, the STAR-100?

Michael: No. A point was made by people, physicists, engineers, and chemists and stuff like that, that the vectorization techniques used on the Cray-1 and subsequent Crays, were easier to handle than the long vectors that were on the STAR. If the STAR vector was not logged, the setup time and the startup time, and all that sort of stuff, became a larger fraction of the overall cost, and the machine was not very fast. On a CRAY, there were all kinds of little charts that show what happened. There would be a spike in the timing between the 64th, and if you made the mistake and had 65 elements in your vector, it cost you. So by learning to fit your zoning capabilities to what was on that machine, you could run faster. So if you had to have more than 64, it was worth [it] to have 128 element vectors. 65 was no good.

Miya: You guys were willing to sacrifice the memory then?

Michael: That is what you say. I do not. It is fine with me.

Miya: Okay. And that is where you guys also learned about memory stride and stuff like that, right? Or did you guys know about memory stride back in the '60, '76?

Michael: Well, it was a thing that was being learned, okay. And one of the things that was terribly important to begin with was the 'gather-scatter' capability. That was available on the STAR.

Miya: Was that vector gather-scatter, or was that just general gather-scatter?

Michael: Vector.

Miya: Okay.

Michael: And it turned out that people had to sort of coax Seymour into doing that on his machines. And it was a very important thing to have that capability.

Miya: But that was not added until the X-MP, right? The Cray-1 did not have gather-scatter.

Michael: Right.

Miya: Right. Okay. You mentioned something that I forgot to put in here of that same era of time, that I did a little bit of research on, and we actually have friends in common with, and that is the Livermore S-1. Say some words about the S-1.

Michael: It is difficult. S-1. Let's start it this way. Lowell Wood was a protégé of Edward Teller's, and a bit of a gadfly. That is, you know, he was in his own. He had his own special projects group, and he was fairly well vocal about being dissatisfied with the capabilities that were inside the computation department. So when one of his friends went to Washington, between the two of them they conspired to develop a machine called the S-1, which is going to be faster, cheaper, more powerful, blah, blah, blah, blah, blah. That S, I think, stood for Stanford because that is where initially a lot of the work was done. I am not going to name names, because I am not too certain, but eventually some of the people that were in that project came to the lab. And they were building a machine which was a superb piece of bad engineering, if you will. It was like, you know, the famous Mulligan stew. Whatever you can find in the refrigerator or out in the woods, on the trees, on the floor, throw it into the stew. So the people who did the initial designing of the S-1 were familiar with PDP-10's, and PDP-1's, and VAXen [plural form of VAX], and things of that sort. So all that capability was put into the thing called S-1. And they made some, I think, ultimately not very good decisions about the actual construction of the machine. Whereas at the end there, they had to go through every wire, if you will, and trim them so that the timing came out right. The machine was huge. It did not have a tiny footprint. Originally it was supposed to be part of anti-submarine warfare, or something like that. And as I said, they had some excellent people on the job. And they did a good job, but they were not, you might say, commercially effective, computer designers. So the S-1.

Miya: They never finished.

Michael: Well, in a sense they did. But it really was not a very successful project.

Miya: I seem to recall it was supposed to have 16 processors, and they never had 16.

Michael: That is true.

Miya: It was supposed to have a big 16-by-16 crossbar.

Michael: They had so much, the big one they had was, it would barely fit in this room. And you cannot imagine 16 of those things. And they had trouble keeping things working.

Miya: Why not? It is a supercomputer. It is supposed to be big. It is like the State of Texas.

Michael: No.

Miya: No?

Michael: Well, my protégé, Howard Davidson, was on loan to them, and what he did was to redesign the packaging of that machine, and he shrank it from something that would barely fit in this room to something that would fit on this table.

Miya: Was that the Mark-II?

Michael: Why, I do not remember what the number was. It may have been a Mark-III. But he learned how to take heat out of these components. Remember, all this stuff was individual components. And he learned how to take the heat out of that so he could get things much closer together. And he was able, I remember one number, 1,500 watts per square centimeter. It was a very impressive number from what they said to me. And that machine was not completed in the sense that they had many more built like it. But it was a separate machine and it proved the point that they wanted. This was supposed to be something you could carry on an airplane or in the hold of a small ship. The S-1, it was unfortunate that its enthusiasm outweighed its conservatism or its knowledge of good engineering practice for building stuff, things of this sort. They had some nice software, but it was not stuff that would be- it was nice and tight and robust. That is the word I was looking for. It should be robust. The machine should be robust. And I cannot help it. They got every instruction in the book in the thing. The DEC engineers discovered that, too, you know. When they went to build from the PDP-3 to the PDP-6 to the 10, they had all these instructions in there. Programmers loved the thing because it had an instruction for everything you wanted to do. But when you notice that it is slowing the machine down by factors of three or four, you think, hey, let's get rid of that stuff. And it made perhaps the most cogent argument for RISC [Reduced Instruction Set Computing] type programming. Well, the same thing was true with the S-1. They had [an] enormous list of instructions that they could execute.

Miya: Yeah. The 72-bit word, 64-bit, 64 registers.

Michael: But it inherently was not going to be fast.

Miya: I thought they were saying it was going to be fast. They are contradicting themselves, hey?

Michael: I am not contradicting me.

Miya: No, no. I am saying they were contradicting themselves.

Michael: Oh. I said I think that it is an example of some bright, young people, students, graduate students, whatever you want, you know, bright, young people doing their first thing, which may show more of a lack of experience than anything else. It was not a model that you could build on and be proud of and all that sort of thing.

Miya: Well, actually they went off to form Key Computer, didn't they?

Michael: I do not know.

Miya: I recall they formed Key, and one of the things they did was they did the SCALD design methodology.

Michael: SCALD was a logic design program that you could use to lay out boards and processor elements and things of this sort. It was done by, I do not know.

Miya: Jeff Broughton.

Michael: By whom?

Miya: Jeff Broughton.

Michael: It could be. I do not remember.

Miya: I do.

Michael: Okay. Tom McWilliams was one of the guys who had a program that would look at a module and do the timing correctly for it. That is not the best way to say that, but that is what was happening. And Curt Widdoes, I guess is the other one. And there is Mike Farmwald. Right. Mike Farmwald, I think he sort of survived pretty well as a RAM designer, and his designs are used today. But as a designer of computers, all these guys were still too green to know the difference between elegance and effectiveness, things like that, or to savor the difference. I think that in their zeal to show up the computation department, Lowell Wood did some things that were not quite effective. In a sense it was his job to impose discipline on that group so that they could take their creativity, focus it correctly, and get a really effective machine.

Miya: And this is all before the Strategic Defense Initiative.

Michael: Before the what?

Miya: SDI, Strategic Defense Initiative.

Michael: Yeah, essentially, yeah. I am not quite sure where the origin for SDI was, but we talked to Bob Kahn and Bob Cooper from Arpa, and I, at least, failed to understand where they were coming from the right way. So there was no connection between us. My argument was that we have some good experience about being effective in computing. And, yes, if you want to have a showplace on the aircraft carrier Carl Vinson, or you want a neat robot that will drive on a road and stay on the road and will not get overthrown by the falling leaves and stuff like that, those are important projects. I think it was kind of a waste of money to install those projects in the academic circle, without any obvious relationship to experience.

Miya: Well, what I am aware of with regard to Lowell Wood and Strategic Defense at that time was because of a college roommate who was on the Canoga Park High School track team, and he knew a guy named Peter Haggelstein.

Michael: Haggelstein? Yes.

Miya: Peter apparently was the one who came up with the idea of doing hard x-ray lasers, and that is what Lowell glommed onto.

Michael: That was not the S-1, though.

Miya: That is what I am asking is what was Lowell's relationship with the S-1 to Strategic Defense. People are going to ask that.

Michael: No, no. They were totally separate projects.

Miya: So S-1 was not going to be involved with Strategic Defense at all.

Michael: Right.

Miya: And Peter Haggelstein was not one of the people.

Michael: Peter Haggelstein, and stuff like that, but this was a set of laser projects in outer space, so to say. The S-1 had its genesis in water. They wanted to process sonar buoy data, and stuff like that. So they were separate.

Miya: Right.

Michael: That does not say they could not have used the ideas interchangeably, but I do not recall that ever happening.

Miya: Well, you know, I mean there is this book out there by what's his name, called "Cold Warriors", which I remember came out, and you guys at the lab really digging the "Cold Warriors."

Michael: Oh, that was that New York Times.

Miya: Well, no, it was "Star Warriors." That is what it was. "Star Warriors."

Michael: Brock, or whatever his name is.

Miya: Yeah, something like that.

Michael: Yeah. Well, he was profiling a lot of the S-1 group. The S-1 group, though, should be seen in its context. It was part of O Group. O Group had the project that Haggelstein started, independent of S-1. O Group was looking at the technology of laser warrioring, laser fighting, or whatever you want, up in outer space.

Miya: And in the end it did not work.

Michael: I cannot comment.

Miya: Is that classified?

Michael: I am sure it is classified, but I do not know enough about it to comment.

Miya: Well, there were supposed to have been some tests in the middle of Nevada that did not lease [?] properly, or they did not get enough yield, or something like that. To this day we do not know.

Michael: Well, that is possible. But, you know, that is just hearsay.

Miya: Well, you know, historians are going to try to track this down to ultimately where Lowell was pushing things like Brilliant Pebbles [an SDI anti-satellite system], or smart rocks, and stuff like that.

Michael: All of that comes out of Lowell's mind. He is a very, very gifted person. In fact, it may be he would have been more effective if he had not been a protégé of Teller's, where if he had to more accurately defend his ideas in a real court of physicists. Many of the physicists at the lab, for instance, were sort of against him because he misled some strategic planners, I guess you would call them, in Washington, about something. I do not know what it was. He was overly enthusiastic, shall we say.

Miya: He was an optimist.

Michael: No. I could be an optimist and still be accurate. But if you are overly enthusiastic, that sort of attacks your accuracy, the proper description of the models and so forth. That is the way I see it.

Miya: So anyways, the S-1 was never completed.

Michael: I think that is true.

Miya: Earlier you said it had been though. Earlier you said that it had been though.

Michael: I do not know.

Miya: Okay. Anyways, enough of that in terms of history. In addition to the S-1, and all these other machines, the Cray-1, the 7600, the STAR-100, you want to say a few things about your guys' interaction with Ames {NASA Ames Research Center] and the ILLIAC IV? You guys must have come over and used the ILLIAC IV a bit?

Michael: Well, the ILLIAC IV was born we say in the mind of Slotnick, Dan Slotnick. And the father of that in a sense was that we asked for a bid, or a proposal, to run some what is called SOLOMON-like architectures on our kind of problems. The SOLOMON, I think we called it today technically a single instruction multiple data path, SIMD. We called a bidders' conference. People came out to the lab and CDC effectively proposed the STAR. IBM proposed a 32-component machine. I do not know what name it had.

Miya: VAMP maybe? Was it called the VAMP maybe?

Michael: I do not remember.

Miya: V-A-M -P?

Michael: And Westinghouse proposed what eventually came to be the ILLIAC IV.

Miya: Not Burroughs.

Michael: Four. We chose the STAR architecture. The ILLIAC IV as proposed initially, was four times the size of what was delivered here, and four times the speed. And, again, it was built by very clever people who did not know diddly squat about large scale computation. So they could build this fast processor, but it did not fit into any architectural scheme where you could use it, exploit it. They started making some headway here when they hired a bunch of computer scientists like Mel Pirtle, Wayne Lichtenberg, Butler Lampson, and so forth. And I forgot the engineer's name. But anyway, his big contribution was to surround the Illiac, such as it was, with a bunch of DEC hardware, which people could use, because they knew how to work in the timesharing system on it, TOPS-10, things like that. It did not do the ILLIAC IV any good, but it started to show how to integrate this so-called fast machine in a structure where people have to be involved, along with programs and real problems. There were lots of people who actually got to use the Cray-1 out of the ILLIAC IV. I think that the only way you can characterize it is they were able to do some calculations that had to do with electromagnetism in the atmosphere. I think they were trying to do DYNA3D. When the machine was delivered to NASA Ames, it only had 64 processors, instead of 256. And it was one-fourth the speed of what they had expected. And they did not have any real software on it at all. And then they finally noticed there really was not very much in the bin where you will find problems you want to compute. There were not very many that would fit on the ILLIAC.

Miya: Not enough memory?

Michael: No. There were a variety of reasons. Let's put it this way. If you cannot organize your calculations so that in this case 64 operations can go on at the same time, and they do not have to be the same operation, multiply, add, you cannot lay out your program so that it works that way, then you are running one at a time, or whatever else it might be.

Miya: The von Neumann bottleneck.

Michael: Well, in the case of the vector processor, it is fetching out of memory and putting back into memory at the same time it is computing. So there is no wasted motion in there, and the business of going memory-to-memory is very fast. In the case of the ILLIAC, single instruction of the 64 processors that are in the memories, which one can execute that? Which one needs to execute that particular instruction? And you cannot use that one. You just say pass, and now there are only 63 possible things, and so forth. Given that, to say it another way, it was hard to find interesting, valuable physics problems that would fit on the ILLIAC IV, things for which people burned to solve. The lab was lucky in the sense that it had nuclear design. People wanted to solve those problems, so they put a lot of money in and they exhibited a lot of patience when it took longer than you might think to get those problems solved. There was not anything comparable to the ILLIAC IV. There are some people that are apologists for the ILLIAC, who want to convince everyone that it was a success.

Miya: Was my old boss, Marcy, one of those people?

Michael: Marcy. I never did discuss it with her. Was it Peterson?

Miya: Marcy Smith.

Michael: Her boss.

Miya: Oh, Vic Peterson?

Michael: Vic.

Miya: I do not think of Vic as an ILLIAC apologist.

George Michael: No. Look. I have a book at home about the ILLIAC IV, and its success.

Miya: Michael Hord's book.

Michael: Could be, I do not remember. But the measure of success is not impressive if you control the definition. If you can accept some other definitions, and you can still show a success, that is good. I think the ILLIAC IV was not a success except in a negative sense. It showed how not to do stuff. It showed that very little in physics of importance can be done within a SIMD structure.

Miya: Very little that is important in physics can be done by SIMD.

Michael: Say that again louder.

Miya: I am trying to paraphrase what you are saying here. You said, "Very little in physics that is important can be done SIMD."

Michael: Right.

Miya: Including vectors.

Michael: It is not SIMD.

Miya: Some people think it is.

Michael: Oh, I do not think so. There is only one data path, not many data paths. The other thing is that I will try to name some things. Heat flow and fluid flow, electromagnetism, these things you can do them in an SIMD way, but they do not give you accurate results, or as accurate as we do them in the implicit way. SIMD is explicit in the sense that everything happens. In effect it cannot propagate more than one cell in a given cycle. Of all the physics problems that could be done, the government will pay for some of them to be solved. Those are interesting physics problems by definition. Of those interesting physics problems, there is a smaller subset that can be done in a SIMD structure, but they are small compared to the set of problems that require a different kind of architecture, not SIMD. MIMD, in some cases, will not do this stuff as well. Now, it is one of the research frontiers right now. People are struggling to learn how to do MIMD calculations for these old problems that were being worked on many years ago. And, who

knows, they may find a way. They can develop algorithms that work, that are stable, that converge, etc., etc. If those things are not satisfied, calculating SIMD is not going to do you any good.

Miya: I think that you have just convinced me that historians will never understand the difference between implicit algorithms and explicit algorithms because they are going to interpret it in a non-mathematical sense. I think that part is going to be hopeless for better or worse.

Michael: I had two images given to me to tell me about that. One of them is an Eulerian view of calculation, in which case you are standing on the edge of the river and the fluid is flowing past you. In the other vision, Lagrangian, you are in a canoe in the river, and you experience these various changes as you move. So that is the first thing. In an implicit calculation, you are in a zone and you make use of the numbers that represent the edges of the zone the right way; whereas in the explicit calculation, you are sort of more isolated. You do not take the advantage of the updates, whereas in the implicit thing you do. No?

Miya: No. The problem is you and I can appreciate that. Historians are not going to understand this. What everybody learns about, what I realize in terms of how we teach mathematics to kids, is that we teach them essentially the Pythagorean theorem. They are looking for direct solution. You do direct solution to get an answer as opposed to iterative methods which, like Gauss, gets there along the way. And along the way, you gain some advantages and you gain some disadvantages, one of the disadvantages being the propagation of errors in floating point. And this is something that a lot of, even physicists, do not even see, the purview of numerical analysts.

Michael: It is a matter that experience shows you do not get the convergence, or the stability, when you are doing explicit calculations. You lose accuracy. You know, the circle algorithm? It is called the tripos.

Miya: Oh, oh, okay. I was thinking you meant just drawing a circle. Go ahead.

Michael: Yeah. Draw a circle, okay. And if you do not use the updated coordinate, but use the old one instead, the thing blows up. That is an example of an explicit calculation, whereas if you use the updated thing, you compute it and then you use it. That is implicit. And it does not blow up. It develops into a circle.

Miya: You have to explain how that gets a circle though.

Michael: No. I do not want to go into that.

Miya: See? That is why actually historians are never going to understand this stuff.

Michael: Well, but I do not have a blackboard to draw on and show the equation itself.

Miya: You have a white wall behind you.

Michael: Yeah.

Miya: Okay. I want to go back to a couple of other things, too, while we are still on the Cray-1.

END OF TAPE 5.2

START OF TAPE 5.3

Miya: The Cray-1, and you said a little bit about DEIMOS, the operating system for the Cray-1. A first operating system for the Cray-1? Who led that project?

Michael: The operating system for the Cray 1 at Livermore--

Miya: No, no. At Los Alamos. DEIMOS. Nobody has ever heard of DEIMOS except people who attend SOSK [ph?].

Michael: Well, all I know is the name and that it was very slow.

Miya: Who was the leader of-- ?

Michael: His name was Roger Lazarus, and Forrest Basket was one of the persons involved.

Miya: Right.

Michael: I think also Jack Worlton may have been involved. And it was a very good operating system, you know, from the theoretical point of view, but it wasn't very fast.

Miya: And what's the point of having a slow operating system? Right? So.

Michael: People can be otherwise unsophisticated about all kinds of things, but they notice when things are fast or when they're not.

Miya: Not only that--

Michael: You even have limits about that. You know that if you can sense a hesitation, it's too slow.

Miya: Well, it's not just that too, but DEIMOS has a very positive reputation in the operating system world -- and yet, very few people know about that.

Michael: No, I don't know that.

Miya: Well, you know, Forrest Basket is a dirty word--

Michael: Why?

Miya: -- for everybody. Because a lot of people-- I heard a lot of people at Los Alamos didn't like him. And yet, we're in a building that's a monument to Forrest.

George Michael: Well, you know, I think that dislike can be characterized as-- by saying that a practicing scientist does not like a gadfly that knows just the computers. I think in a sense, without trying to say anything about Forrest, is that the people who are running the machine have an obligation in general to learn as much as they can about the behavior of their users, the behavior of the user programs, and the needs of these things.

Miya: Now, was Seymour good at that?

Michael: I don't think-- let me add one other thing to what I was saying. There are a group of people-- there is a group of people that are so structured that they don't need to ask such questions. They have a sense of what's right and what acceptable. And Seymour was one of those.

Miya: Uh-huh. And do you have a sense of where he got that?

Michael: He was gifted.

Miya: It wasn't because he worked at ERA [Engineering Research Associates] and he worked on specific problems that Howard Engström and the folks at--

Michael: Engström.

Miya: Lunds-- no. Engström.

Michael: Engström was a writer.

Miya: Yeah.

Michael: I cannot answer the question in general, you know, and specifics whether Seymour was born that way or had evolved that way during the war or after the war, or something like that. The point is that he was enough in both worlds-- that is applications versus systems. He was at home in both places enough so that his ideas of what the system ought to do for the user were very good. That's a gifted person.

Miya: Yeah.

Michael: Norman Hardy's like that. Okay?

Miya: Uh-huh.

Michael: I mean he isn't a system programmer, but he's one of the best system programmers. He isn't an applications programmer, but he's one of the best.

Miya: So we should interview-- we should do a separate interview with Norman.

Michael: Well, the point is he knows what's going on, and--

Miya: No, but independent of the interview you did for him.

Michael: I can't hear that.

Miya: Independent of the interview that you did for Norman on your Web site.

Michael: I can't hear that. You've got to raise your voice Gene.

Miya: Should the Museum do an interview for Norman like you did for your Web site?

Michael: I think that would be good if you could do it.

Miya: I mean who in computing has heard of Norman Hardy?

Michael: I think Norman is an absolutely gifted person. Okay.

Miya: I don't deny that. The question is the Museum has to figure out who it needs to interview. Those are the kinds of--

Michael: Well, I think you should just [find] him and, you know, his answer will help to define to do.

Miya: Yeah. Okay. I'll do that.

Michael: What I've seen-- you know, I've worked a lot with Norman, and it's just always a thrill. You know, when it's-- it's like he's been thinking about the problem you've just come up with. It's brand new? He's been thinking about it for a half a year. And he -- and his suggestions are right on the money every time. It's just-- just scary. And we had a few people like that. Chuck Leith is like one of those. Okay? And Forrest Basket, apparently, wasn't. Certainly Roger Lazarus wasn't, though he may have felt that he didn't have to because he was in charge of communication and computation.

Miya: Okay. Are there other things you want to say about the Cray-1? How many-- let's see-- how many Cray-1s did the lab end up getting, four or something like that?

Michael: Something like that. Yeah.

Miya: Yeah. So and they were all basically Cray-1 Ss or As and the like?

Michael: Well, the Cray-1 was structurally enough like the 6600 and the 7600 that it could be easily used. But more important, the users felt that making an effort to tune something up really helped them. Okay.

There are some of the users who were, you might say unimpressed with Seymour. Okay. _____ he done a lately [ph?] or something like that, and it's too bad about those kinds of people. But if you take the trouble to look at what he designed, to look at what he's produced, look how it's produced-- to look how it maps to the problems you're interested in-- you can't help but be impressed. Nobody-- the only person I know of who comes close to that is Burton Smith. You know? It's easy to design a computer. It is not easy to design a computer that does effective computation very fast and is intuitively the right ways to do something. Seymour had that-- much more than Burton. Burton has it too.

Miya: The STAR-100 did not have that intuition.

Michael: Hmm?

Miya: The STAR-100 did not have that intuition.

Michael: It had some of it. But remember the STAR-100 was guided by 90 percent of the first two chapters of APL, *A Programming Language*.

Miya: And?

Michael: At our first Salishan conference, back in 1981, we had designed an afternoon off so people could wander around and kick the sand on the beach and stuff like that. But for those who didn't want to do that, there were special little lectures-- you might call them BOFs [Birds of a Feather] of a sort. So one of the physicists at the lab who deals in neutronics, Ernie Pocate, was prevailed upon to describe Monte Carlo neutronics the way laboratory wants to do them. Okay. One of the persons in the audience that are listening to Ernie was a fellow by the name of Neal Lincoln, the designer of the STAR.

Miya: R. Fred Neal. [laughs]

Michael: After he heard the lecture, he said, "You know, if I had known that before we started, the STAR would have looked different." That's all I can say.

Miya: So we need to find a copy of Ernie's presentation. That would be useful.

Michael: Who?

Miya: The Museum should find a copy of Ernie's presentation.

Michael: There was no copy. These are spontaneous talks. No papers were written; no publications. In some cases, an abstract of what the guy was going to say-- but in this case it's just, you might say-- a hallway conversation on a free afternoon. You've been up there. You know Wednesday afternoon's free.

Miya: I know what it's like. The problem is we have to get this across to future generations, who never go to Salishan.

Michael: You can ask Ernie. He's up in Portland. You can ask him to write down the salient points of Monte Carlo calculations in an unclassified way as they apply to Livermore. I'm sure he'd be happy to do something like that. He's a great person.

Miya: Good.

Michael: I'll ask him if you want.

Miya: Yes, do that. No the problem I was trying to get out of you is: What's the problem with the first 90 percent of the APL manual? I mean the APL people would be furious at you guys. Right?

Michael: According to Neil Lincoln and Purcell-- Chuck Purcell-- they should have done 100 percent. In other words, if you made it exactly, APL would have been an easy translation. So there was these ragged edges, which I can't explain too well.

Miya: Yeah, yeah. That's in my questions for CRG, SISAL [ph?] and Steve Skedzielewski. So I have these in mind. Go on.

Michael: The question of if the first two chapters define APL, Iverson's APL, and you say we only implemented 90 percent of it-- it becomes fairly important to know which 90 percent. What is in the 90 percent? Now that is not easy to say.

Miya: Right.

Michael: I'm just quoting Neil Lincoln and some of his colleagues. And I don't know how much more you could say that-- if you don't provide a thorough, complete facility, then when you try to do something, you'll find some places where you don't have the right thing to end it off or complete it or whatever it might be. And that's, I think, what the situation was with STAR.

Miya: Well, I think, you know, that's the way a lot of computing is. That's-- it's a 90 percent solution.

Michael: Yes. But I mean and it's--

Miya: And the other ten percent takes the other 90 percent. Right?

Michael: Yeah. But in this case, they had the book, and it was a conscious decision to let the vector processors that were going to be in the STAR look like APLs. APL is a vector processor.

Miya: Right. Okay, I think that's going to be not well received. It's going to be incomplete because people are going to want to know-- see people are looking for, especially in a lot of these history talks, is lessons learned. The reality is that you're saying, "Well, you know, this conversation took place, but no lesson was learned, except amongst the insiders." That an example.

Michael: Well, that's what scientific meetings are for. So specialists can get together and talk to each other.

Miya: Right.

Michael: That's all this was.

Miya: But there's no record of that conversation or the like. So this is where it's going to be really important for you to do follow-up. Okay?

Michael: The right people got the information. All other hangers-on didn't.

Miya: There are some people who are designing computers now who weren't born at that time George-- or they were in diapers. <laughs>

Michael: Well I don't like to say. I don't know.

Miya: No. That's just simply-- this is why Livermore commands no respect in computers anymore. Right?

Michael: Livermore has made a conscious decision to disassemble all of the hot shots that were there, and they don't have them anymore. There's still some good programmers there, but they-- the ambient-- the ambiance of the place-- what do you get to work on and so forth and so on-- that's changed. It's not like it used to be. Everybody says that, but you'll never get an agreement about what really changed.

Miya: Well, you know, so we're going to have this Cray-1 event. And what's-- what's this Cray-1 event going to be about in terms of if the people are there. I mean they're going to learn about the impact of a machine, which was delivered over 30 years. Right?

Michael: I don't know. Let me give you one example that-- what's learned and how it will influence a given person is up to him. You got a machine, the Cray-1. Let's say it'll do a multiply in seven nanoseconds. Pick a number, okay? Okay. Here's a guy who's an applications programmer or a physicist. And he has 200 billion of these operations that he has to do. Okay. Clearly he's going to be more interested in a machine that'll perform one of them in seven nanoseconds than in a machine that performs one of the in seven microseconds. Okay? That's how people greeted the Crays and other things that Seymour did. He provided raw speed. Okay? And it cost. Ask Bill Carlin. They'll know.

Miya: Right.

Michael: You know, and in order get the speed, he did some things, which are called "not nice" in arithmetic. Okay? Rounding before a division-- things like that. Well, you can't do that without paying the price. But Seymour's dedication and devotion toward high speed impelled him to cut cycles wherever he could.

Miya: Now, let's see-- did Seymour ever formally say the word "supercomputer?"

Michael: Not that I ever heard.

Miya: That's what a lot of other people said too. Supercomputer was always a--

Michael: He said, and in fact we have him on tape, said, "I'm interested in building very fast machines. That's what I concentrate on." But he never called them supercomputers. And as-- I think supercomputing is such a crummy marketing term now that it should be retired from the vocabulary of a real scientist. You could talk about high performance computing, and you can even put numbers to the thing, you know, and so many operations per nanosecond-- or whatever it might be. But everything today is a supercomputer. In fact, you know, it's sort of funny. I have that Macintosh G5 sitting on my desk. It's faster than anything lab ever had while I was there. But it's not a supercomputer. It's a crummy toy. It's not a bad machine. It's just a toy.

Miya: Okay. Shall we move on?

Michael: Could you make it faster? Yeah, I suppose you could, but I don't know whether that'll do you any good. Move on.

Miya: Okay. Now about this time is when I met you. So I'm going to hold you more to what I remember it was about you. Okay?

Michael: About what?

Miya: This is about the time that I met you. Okay? Had you come over to Ames?

Michael: That's a long time ago, isn't it?

Miya: Yeah. <laughs> Yeah. You gave a talk; our wind tunnel blows up.

Michael: Yeah, I remember that.

Miya: And let's see--

Michael: I don't remember what the talk was about, but it was probably reporting on our trip to Japan.

Miya: No. It was just generally talking about supercomputers, but yeah-- talking about Japan, which we should talk about some more. But also, you hired away from me a brilliant computer scientist named Steve Skedzielewski, which I'll have to tell them how to spell his name, since I learned how to do it.

Michael: "Sked--"

Miya: "dzielewski." Yeah, yeah. Now you wasted that time on the tape, but anyways-- so you were not quite a group leader. Right? At that point in time. You guys were just then starting the computing research group.

Michael: I wasn't _____ group leader.

Miya: And you had Jim McGraw and Steve and who else?

Michael: -- and Jimmy and Charlie Wetherell.

Miya: Right.

Michael: And those two helped me decide to hire Steve. Okay? They also were influence-- well, Chuck Wetherell, in fact, was influential in getting me to hire Jim McGraw. And the two of them helped me to hire Steve. And--

Miya: Who else was in the CRG, as well too?

Michael: Well, let's see-- Michael Welcome and--

Miya: Kelly Booth?

Michael: Kim Yates.

Miya: Kim Yates. I remember Kim. Kelly Booth?

Michael: No. He was in-- on the side. He was in the graphics group with-- well, we had-- at one point, we had Art Sorkin.

Miya: Right. I remember. Diane--

Michael: And Diane Smith, and Roger Anderson, and Bob Crawley. And there's one name I'm forgetting too. Anyway, we were a group; and whereas most of the people at the lab didn't respect the group, the people on the outside did. And--

Miya: Why didn't lab respect the group?

Michael: Because it was more oriented toward research than it was towards doing the applications programs that nuclear design needed. What I tried to do was that as each person came into the group, I would ask them to go over to one of the nuclear design groups and spend enough time so that they went through the problem setup and analysis cycle. Okay? I don't know how many of them did that. I think I got Jim McGraw to do it. But you know, it wasn't all just a research group. I remember that after our adventuring into SISAL, McGraw wrote one of the loops in SISAL that would apply to one of the big programs. And it was dramatically faster than Fortran. But you could never get those people to pay much attention to it-- that is, the device designers. Why is that? I don't know, but there are just all kinds of examples. We bought a thing called the LDS-1, Line Drawing System Number One, and it was built by--

Miya: E&S.

Michael: -- Evans & Sutherland. And it sat in the machine room-- was for the device design group. As near as I could tell, nobody ever really used the thing. You couldn't get them out of their offices. And I couldn't understand that. We had to provide the capabilities that those people wanted on the terms that they wanted. "I want to be able to do it in my office." They glommed onto our TMDS like it was going out of style. Everybody wanted one in his office.

Miya: TMDS? What was that?

Michael: Television Monitor Display System. By itself it wasn't very impressive. But when you coupled it with the software that was produced for that system, there was nothing better in the world. And I'm not exaggerating on that.

Miya: I don't doubt you.

Michael: You could get noise-free pictures of your stuff, plus text that you could edit on the screen of a television set-- all in your office-- at the speed of 30 frames a second or more. Those were effective.

Miya: So was McGraw visiting one of the design groups _____ in Jim's career?

Michael: I think I can't answer that really.

Miya: I mean he's an assistant laboratory director now?

Michael: No. I don't think so. I don't know.

Miya: I think he's an assistant laboratory director. Okay. Anyways, --

Michael: I mean I retired.

Miya: I'll go on. Okay. Because see, most people are not going to know about SISAL. So how did you guys decide to go down the road of dataflow?

Michael: Go down the what?

Miya: The road of dataflow.

Michael: Uh..

Miya: Tell us about SISAL.

Michael: Okay. One of the places I regularly would visit when I was on a trip was MIT. So I went and talked to Mike Dertouzos-- he was the head of the laboratory for computer science then-- and he was saying, "Well, you know there's some interesting work going on." And so, I went to Ed Fredkin and talked

to him. And he was saying that they had made a lot of progress in automating the writing of programs-- so that you don't write bugs into your program and so forth. And then Mike Dertouzos told me to go and talk to Jack Dennis. Jack Dennis explained his ideas about dataflow, and at first I didn't really, you know, resonate to it I guess is the way to put it. Yeah, it's pretty clear, but it's very impractical. But in any event, I decided that I would have a high-level meeting. At that meeting, which was held at my house, was Gus Doro and a couple of the Weis physicists. We had Mike Dertouzos, Joel Aaron, and Jack Dennis. And basically, the result was that the lab decided to put some money into Dennis's research. Dennis was showing how, if you could write the programs the right way, you could compute in 10 minutes what it was taking 20 hours to do and stuff like that.

Miya: Is this the house with the geodesic dome, or is it the house before the geodesic dome?

Michael: There was no-- I was living in this 20 room house then.

Miya: Before your current house?

Michael: Yeah. This was before the geodesic dome. Anyway, one of the things Jack was on the hook for was a version of his dataflow language that met some of the constraints that we had to have for nuclear design. One of the key points of dataflow is that they only use the storage cell once. You never, you know, you don't have any side effects so to say.

Miya: Single assignment.

Michael: Single-- right. And one could see, given the kind of calculations that are being done at Livermore, that you would be visiting each place maybe a billion times in the course of a calculation. You can't afford to have that all just single assignment and that you don't change the storage. So out of that Jack and one of his graduate students, Bill Ackerman, came up with this design called VAL. It was a paper design. But he had agreed to come out to the lab for a summer. So he and Ackerman came out, and basically, we locked them in a room along with Charles Wetherell and I think maybe Jim McGraw too, and they evolved a version of this dataflow language, which could live in a real memory and handle the side effect free thing that was the equivalent of single assignment. That's VAL-2 or something like that. Then the funding for Jack dried up. But the guys in the research group continued developing this thing, and they came up with the version that we now call SISAL. SISAL had two characteristics-- it was good dataflow, plus it made use of Fortran I/O. And then to prove that it would do right, we took the inner loops of various of the design codes and wrote them in SISAL. And in all cases they came out faster than Fortran. Well, I thought-- and this is where I was wrong-- I thought that was enough to argue that you want to continue development, because it's faster. The problem is that if you call Fortran and imperative language, SISAL is not imperative.

Miya: It's a functional language.

Michael: It's a functional language. And the people that we were trying to interest-- you know, the device physicists and so forth-- just couldn't do it apparently. But we built SISAL and had a compiler for it, and it used Fortran I/O. And it would do the problems that these people wanted to do and had to do, and it was fast. But it is improper to try to keep the research staff writing these application programs and not doing research. And whereas, the applications programmers wouldn't leave Fortran.

Miya: Whose research?

Michael: Sorry now.

Miya: Whose research-- the computer research or the physics research?

Michael: Computer research.

Miya: Ah!

Michael: Yeah. It was simple, you know. I don't know what else to say. SISAL did the job; it was faster, gave the same results; and some of the people in the big divisions were actually using it. But the greatest success-- it came from Switzerland, New Zealand, Australia, England-- not at Livermore. And when the research group was disbanded, it died—more's the pity. I think it would have been excellent if it had been left to run and was available today. Because the big stumbling block with the machines that are being delivered now is sort of MIMD, and SISAL would eat that alive. It could manage all that stuff. You don't have to put a copy of the operating system in every computer as IBM does.

Miya: I think it's actually still possible to run it. Because, as I seem to recall, what SISAL compiled to was intermediate form 1 or intermediate form 2-- and then that was then compiled to C. So long as you've got a C compiler, you can compile SISAL.

Michael: Well, all I know is that the so-called [IBM] BlueGene/L or all that stuff with the lab supposedly has a copy of the operating system in every machine.

Miya: Operating system or compiler?

Michael: The operating system.

Miya: We're talking about the language SISAL. It's a compiler.

Michael: The compiler is part of the operating system. I mean there's a file management thing, and the only way they have for linking those things together is message passing from one machine to another.

Miya: I think some people would make a big distinction. Dick Watson wouldn't, which I like about Dick.

Michael: I don't know what Dick Watson's position is on this. I don't think he's ever expressed it to me.

Miya: Yes, he did. We had that history talk at Sun Microsystems on operating systems history. And he gave his generations of operating systems.

Michael: Oh. Good.

Miya: And what he basically said was-- well, what he said actually when he visited at Ames was, "Unix set operating systems back." And now he says Windows has set operating systems back. So. Okay. So anyways, let's continue on. Okay. So we didn't say how SISAL was named. So--

Michael: I don't know how it was named.

Miya: Steve told me the story. Steve said what they did-- you guys had two machines. You had the Circus VAXimus, a VAX-11/784, of which I think they made five, of which Ames had one, Carnegie Mellon had one, Johns Sopka [ph?] at DEC, and I forgot where the fifth one was. I might have been NSA or someplace like that.

Michael: We had a Circus VAXimus.

Miya: Circus VAXimus. And then you the Gluteus VAXimus, which was a 780 then 785, which was the host lll-crg. And Steve came over and helped us-- that is Ron Daley's group with the NAS [NASA Advanced Supercomputing Division] get our Unix systems up and running. And it was history from there. And we trounced the competition. And it also equivalently happened with you guys too. Right? Because you guys had your first Unix systems in-house before you had your first workstations. And your users moaned and complained and bitched just like our users did. Right?

Michael: Don't say right. I don't know.

Miya: I know they did.

Michael: All right.

Miya: Okay. Because I want to get to the Cray-2 here in a moment. Okay. So I seem to recall-- okay, the way Steve said that he named SISAL, and used Single Assignment Language -- SAL. So on the Gluteus VAXimus he grepped SAL into the dictionary userdictwords on the Unix system. And three words came up with SAL. And one of them was SISAL, and he thought, "Great. What a great name." And then that became "Streams and Iterations in a Single Assignment Language." And that's how SISAL came to be as a name.

Michael: Well, it was a good thing that he didn't find SALE, huh? <laughs>

Miya: Well, that's a different story.

Michael: I understand.

Miya: Well, yeah. And similarly, that's how Convex got it's name as well too. Steve Wallach grepped for a bunch of different words in the dictionary, because they couldn't use PARSEC. And PARSEC was already taken in Texas, so they came up with CONVEX, and thought "CONVEX. Wow! What a great name."—as an example. Okay. So anyways, one of the perennial problems with SISAL arrays-- because Jack apparently never really thought very well of data structures in VAL, was the question of: Okay, if you have

a variable that only has a single assignment, that's easy. But what does it mean to have an array that only has a single assignment?"

Michael: Right.

Miya: And so the problem there was dealing with, in two-dimensional cases, and just--

Michael: Well, that was the basic research that had to be completed for SISAL.

Miya: Right. Edges, faces for three-dimensional corners and stuff like that.

Michael: I think that Steve Skedzielewski was working on that.

Miya: He in fact did work on that, as an example. The problem is, you know, it was a notation to come out with how you express this stuff. And it forever remained ugly because of the fundamental problem that single assignment doesn't work too well with data structures.

Michael: Well, my overall plan was that I was going to have something, which you now call Mathmatica. But just for a restricted bunch of things, which we'll call partial differential equations. And for that subset of those that deal with nuclear design. And those would represent the output pipe of Mathmatica. And the input pipe was just what-- you know, you write down mathematical equations, and the machine tries to solve them analytically, and if it can't, it builds a difference table and does translation to Fortran to or SISAL. And we were opting for SISAL because it was faster.

Miya: I seem to recall there was a Nobel Prize winning physicist by the name of Robert Wilson, from Cornell, who claimed he was going to do such a thing.

Michael: Yeah. Well, I had a guy by the name of-- ah!-- that was the guy-- Ted Einwohner [ph?] was the other guy in our group.

Miya: Ted who?

Michael: Einwohner.

Miya: How do you spell that?

Michael: "Einwohner."

Miya: Okay.

Michael: And he was working in this general area of automatically producing a SISAL program or a Fortran program. And also, you know, providing you with the systematic memory of what you've done so you don't have to do it more than once ever again. But that died when the research group died.

Miya: But he didn't get along-- I asked you that-- he didn't get as far as Jim and Steve and the others did.

Michael: He was just a very strange kind of person-- I thought a very good person-- but I mean he was strange.

Miya: The lab was populated by strange people.

Michael: He felt he was persecuted all the time.

Miya: It sounds like the lab had a lot of people who felt they were persecuted.

Michael: Well, [I can't help it], but a lot of people. I know he felt that. Okay? And his behavior would stimulate people to persecute him. But he was one of the four people that we-- we used to meet regularly for supper once a month.

Miya: Like we're doing here in a couple hours. Right?

Michael: No. It-- well, yeah. We-- it was called ACME, for Axelrod, Crawley, Michael, and Einwohner. And you know, it was a fun group of very heavy, argumentative people, and-- but enjoying dinner together.

Miya: Uh-huh. Okay. We're going to wrap this up. Let me ask you a quick question. Kelly Booth on the graphics side-- you mentioned earlier-- and that was with Nelson Max?

Michael: Well, they were both in the graphics group.

Miya: And that was about it right?

Michael: Actually, we had a system, which was put together for certain engineers who were miles away from the computer center. So in the computer there was a [Xerox Data Systems] Sigma-7 [mainframe computer]. And it's unfortunate that that's what we had, but that's what we were forced to have. Out where the engineers were was a Sigma-2. The software to drive those back and forth was part of Kelly's responsibility. We had a thing called Gordo, a time-sharing system, on that machine, and it had had the guidance--

Miya: Stop. Okay. We just ran out of tape. Want to take--

END OF TAPE 5.3

START OF TAPE 5.4

Miya: Okay. So we were talking about Kelly Booth working on the graphics side with Nelson Max system for engineers miles away. Is that Site 300? Engineers were at Site 300 or something?

Michael: No.

Miya: They were at a different location?

Michael: It wasn't Site 300. I was in the-- on the square that the lab exists on.

Miya: Okay. They were just at a remote part of the same lab?

Michael: Yeah. It was just-- it'd take a very long wire, okay, and this is again to save people from having to come down to the computer to do their work.

Miya: The computer center was in the southwest corner of the laboratory?

Michael: Yeah. It still is.

Miya: Yeah. Building what-- _____?

Michael: Well no, it's now in the-- more or less central south now.

Miya: Building 113 or something like that?

Michael: That's gone.

Miya: Yeah.

Michael: But you know, it's not the computer center anymore.

Miya: Yeah. Yeah.

Michael: Anyway--

Miya: And the timesharing system was called Gordo?

Michael: Gordo. And the guy in charge of the machine at that point was a guy by the name of Dick Conn. And among the interesting things that Kelly did was to take care of the operating system so that the Kriegspiel could run on the machine.

Miya: Who?

Michael: Kriegspiel is a refereed version of chess where you don't get to see your opponent's moves. You only know if he's in a legal position relative to you, and you know when you lose material or gain material from your opponent. And the computer acted as the referee, and two people would sit at chess

boards-- they can't see each other-- they just move their own chessmen, and the referee will tell whether it's a legal move or not. Anyway, Kelly was one of the three people-- the other being Charles Wetherell and Tom Buckholz. And they put together this Kriegspiel game, if you will. Kelly made some special kinds of movies to support his thesis, which he was getting from Berkeley's computer group-- computer science group. And I'm sorry, but I can't remember what the subject of his thesis was, but he was able to, you know, animate the thing and put in good illustrations into his thesis like that. Another person that did something similar was John Beatty, who took-- well, he produced a thing called Red PP, and that was based on TRIX [ph?] and TRIX-AC [ph?]. And it allowed you to produce print-ready graphics. And that was their contribution in the graphics group. So they weren't formally in the group as far I can-- I think John was a student employee at that point.

Miya: Let's see, TRIX was the editor.

Michael: Can't hear.

Miya: TRIX was the editor?

Michael: TRIX was an editor, yeah, a text-editor.

Miya: Yeah, yeah.

Michael: And what made it very interesting was that the guy who wrote TRIX, Henry Mall also fixed it up so that its output could be viewed on a television set. Okay? And you could edit it from a keyboard, print it, search for things-- extremely powerful. The most popular version of that was TRIX-AC. I think the AC stood for Alex Cecil. And what he did was to imagine what you'd want to do if you were editing a manuscript or something like that. All of the things you might want to do-- search forwards, search backwards, skip this, do that-- all-- everything-- he produced in a thing called TRIX-AC. Then, using those two things, John Beatty produced Red PP. I'm not sure what PP stands for. I once thought it stood for problem program. Maybe that's still the case. Now, Nelson Max--

Miya: Now, wait a minute. Beatty and Booth were among-- in the graphics field-- two of the three Killer Bs. Right?

Michael: They weren't formally in the graphics group. It's just--

Miya: No. I'm saying-- I'm saying--

Michael: -- their interests went there all the time.

Miya: Right. But I'm saying, in terms of relating to history, there's a very famous report on Splines [ph?]authored by the Killer Bs. Right?

Michael: All right.

Miya: Batles, Barsky and Booth. So if you're famous in graphics, you have to have a first name that begins with B, like [Jim] Blinn [or] Blumenthal.

Michael: Brian Barsky was a friend of Kelly's, and I've forgotten exactly where he came from, but he would come out and I interacted with him a lot.

Miya: I think he came from Waterloo.

Miya: Michael: Yeah. That's right. That's why Kelly knew him.

Miya: And that's why I know him. Okay. So that's why they're the Killer Bs.

Michael: Okay.

Miya: So you never saw the TV series Saturday Night Live. So you don't-- you don't-- apprec--

Michael: You're right about that.

Miya: Yeah. You don't appreciate the comedy of a now dead comedian named John Belushi, who also begins with a B. And the joke on those guys from Waterloo _____ Batles, Barsky and Booth. And you call yourself a Singgraf [ph?]! <laughs> Okay. So we don't know what PP was for, and you were going to say some additional things about Nelson Max. Sorry about that.

Michael: I thought that PP stood for problem program, but it's not correct. It doesn't.

Miya: Right. And then you were saying Nelson Max.

Michael: It's like print or something. Anyway, Nelson Max was a fairly well known filmmaker before he came to the lab. He had done a film showing the mathematical enversion of a sphere. And he came to the lab because of-- we had sort of very good facilities, and he could really produce films there that had never been seen before. He produced one that was called Carla's Island, for instance. Another one that showed a tree with leaves and aging as a function of time, and I'd say without much fear of being wrong he has produced more film than anybody else at the lab.

Miya: Right. I agree. Anything else you want to say about Nelson? George? I mean I want Nelson to give a history talk about the history or graphics at Livermore. But it's always on the back burner.

Michael: Yeah.

Miya: I mean they think they have-- that we have too many Livermore talks is what they think.

Michael: Yeah. He's an interesting person.

Miya: Okay. Let's go on now to-- should we say anything about X-MPs and Cray Y-MPs?

Michael: No. That's after me.

Miya: Okay. Nothing on X-MPs or Y-MPs. Okay. What about the Cray-2?

Michael: The Cray-2 showed several typical Seymour Cray trademarks. It shrunk a factor of four. And it was faster. But he had the same problem on the Cray-2 that he had on the Cray-1. He couldn't get the memory speeds to be adequate for the logic and the arithmetic speeds that he had. He was very ticked off by that.

Miya: So he made banked memories. Okay. He made memory banks. Keep going.

Michael: Yeah. Well, I don't know what more to say. He- he didn't do some of the tricks that were perhaps available to speed things up. Everything was just very simple, and in the case of rounding numbers that shouldn't be rounded, it's wrong, but you know, he did it. He pioneered the business of Fluoroinert cooling.

Miya: Right.

Michael: Up to that point, there wasn't any way to tell where you were going to go when you die or, you know, you're going to have to get some cooling done-- how's it going to get done? Seymour solved that with his immersion. And the machine, as usual, is a dramatic statement. It's small and unlike anything that's ever been seen before. It was just beautiful. It's sad in a way, because the memory speed disparity there was even more severe than it was on the Cray-1. So you could compute like fury, and then you have to wait like fury for it to fetch another operand or something like that. The Cray-3 in that same genre had--

Miya: Wait! Let's stay on the Cray-2.

Michael: I'm just going to say on the Cray-3, but you made my thought go away.

Miya: About the memory disparity?

Michael: Yeah, it was about the memory, but I can't remember what I was going to say.

Miya: Don't worry about it. I'll ask you about it later. Okay?

Michael: Oh, okay.

Miya: Okay. So…

Michael: The Cray-2--

Miya: They had this software war! You guys wanted to have your operating system, and Creon and I argued for our operating system. And we won.

Michael: _____ here at Ames.

Miya: At Ames. We had to convince-- because--

Michael: Yeah. I remember that business.

Miya: The Cray operating system is neither COAST [ph?], no CTSS, nor Muddle or whatever the other ones were, and CARLES [ph?] would run on the Cray-2, because they dropped B and T registers and they brought in a local memory. So we convinced them they could run UNIX, which you guys hated. Right?

Michael: I don't know.

Miya: I do know.

Michael: Okay.

Miya: And Dick Watson was well meaning. He wanted NLTSS. He wanted LINX as an _____ protocol. We exchanged a lot a people.

Michael: It was obvious to the people who were working on this stuff at the lab that networking is what's going to happen. And LINX was that attempt.

Miya: It was a noble attempt.

Michael: It was a good attempt. It was just late. The place had fallen apart by then.

Miya: Livermore had fallen apart as a computer place?

Michael: I don't know.

Miya: That's what I'm asking you.

Michael: I don't like to go out there anymore. It's just not very pleasant.

Miya: No longer computing heaven.

Michael: I liked it before, you know. It was just exciting and--

Miya: The gold is tarnished. Yes? Did the gold get tarnished?

Michael: I can't understand that.

Miya: The gold-- the golden era-- the Golden Age of computing at Livermore-- get tarnished?

Michael: Yeah. There was a lot of that. What I think we should maybe draw as a kind of a lesson is that, willy-nilly, having faster components does not guarantee that you're going to have a faster computer. You can't have a faster computer without them, but it's not necessary and sufficient. So you know, that's unfortunate maybe, but that's the way it is.

Miya: But the lab actually did get a couple of--

Michael: Hmm?

Miya: But the lab actually did buy a couple of Cray-2s, unlike Los Alamos.

Michael: Well, again I was not-- I'm retired by then. Okay? But there was no Cray-2 in the Livermore computer center. Okay?

Miya: It was at NERSC [National Energy Research Scientific Computing Center].

Michael: There was one at NERSC. And don't know where the second was.

Miya: Well, the first one was the one-processor machine that Tony Cole has.

Michael: Well, I don't know what to say. You know, It may have all that stuff on it right now, but I don't know about it because…

Miya: And the other Cray-2 is sitting in Visible Storage [at the Computer History Museum].

Michael: Where?

Miya: Visible Storage.

Michael: Cray-2?

Miya: Yeah. NERSC. That Horst Simon gave to the Museum.

Michael: Well, why isn't it in use?

Miya: Because it's a Cray-2. It's old.

Michael: It could be old, but it's still usable.

Miya: No. It's too expensive to run. Right?

Michael: I guess so. I don't know.

Miya: Well, you know, I'm asking. You know, and you get more memory on these things these days. This has as much memory as a Cray-2. Right? The Macintosh. Macintosh Cray-2! Right?

Michael: Yeah. You could label it that. That won't make it that though.

Miya: Uh-huh.

Michael: These things go out to lunch when you have to do I/O.

Miya: Of course. We haven't even talked about disk striping, have we? You remember what disk striping--?

Michael: I hope we're not going to...

Miya: <laughs> Well, when did disk striping first appear on a Cray? Was there disk striping on the Cray-1?

Michael: I can't hear.

Miya: Was there disk striping on the Cray-1?

Michael: What's a zip-drive? I don't remember the term now.

Miya: Disk-striping. I/O to multiple disks at the same time.

Michael: I think that was tried, but I don't know that it-- I don't think it was matter of standard operating procedures.

Miya: But the Cray-2 did have disk striping. George. The Cray-2 did have disk striping. So did NLTSS have disk striping?

Michael: Yes. I'm just pushing it. I can't think of anything that's sort of "important" that it didn't have.

Miya: Uh-huh.

Michael: And one of the things it had that was to its advantage-- it was written by just two people. Okay?

Miya: NLTSS?

Michael: Yeah.

Miya: Who?

Michael: Jim Minton and Donna Deverona [ph?].

Miya: Didn't Dick Watson write any of it?

Michael: What about Dick Watson?

Miya: Didn't Dick Watson write any of NLTSS?

Michael: No. Jim Minton and Donna Deverona wrote it. The specification was the result of Watson and Fletcher and Donnaly and others meeting every Friday afternoon and talking about what they wanted. See after NLTSS, Sam Mendecino wanted to get working on the next operating system, which was going to be NLTSS. So he established a group that would meet on Friday afternoons, and they'd talk about it. The person who came from Sam's group was Bob Crawley.

Miya: Right.

Michael: And he was there, and what they would hear about is how's the S-1 performing, you know but _____.

Miya: Yeah? Just a few more minutes George. Keep going. What else besides Sam Mendecino represented by Bob Crawley and the S-1?

Michael: I can't understand that Gene. Why don't you talk louder?

Miya: No. You're dozing off.

Michael: Well, that's because you're not talking loud. I can't hear you.

Miya: That's because you don't have your ears in.

Michael: I told you they're broken.

Miya: Right. We should have probably cancelled the talk, but you've done pretty well up to this point in time. Bob Crawley. Maybe we should just move on to the Cray-3. Want to go to the Cray-3?

Michael: No. I'll finish that story. Sam put Bob in charge.

Miya: Right. On Friday meetings.

Michael: Okay? Friday afternoon meeting. And people would come and talk about what they wanted the operating system to do. Bob chose MODAL [ph?] as the implementation language.

Miya: MDL?

Michael: -- from Los Alamos. It turns out that since Los Alamos wasn't going to support it, it was not a good idea to go into that, but that was not known at that point. Anyway, Bob didn't finish anything to Sam's satisfaction, so he gave the job of running the group, trying to design the system, to Dick Watson. And however the system got to look is the result of the conversations that were held on Friday afternoon, number 1. And number 2, the kind of actual programming that was done by the guy and this girl.

Miya: Yeah. And I think we sent Andy Goforth over to participate in those. Right?

Michael: I don't know. I'm--

Miya: I'm just letting you know.

Michael: I don't have any records of that, you know.

Miya: Okay. That's okay. I have a few. So...

Michael: All right. Well, that's right.

Miya: Yeah. Okay. So let's move on to the Cray-3.

Michael: The which?

Miya: The Cray-3.

Michael: Well--

Miya: How come you got a Cray-3 hardware manual? How was it that you rated in the world that you could get a hardware description manual on the Cray-3 and nobody else would?

Michael: I suppose it was because I asked for it. I don't know. I got that from _____ didn't I?

Miya: I don't know.

Michael: Well, I don't know either. On the Cray-3, the only one that was ever built, you know, went to NASA-- I'm sorry-- NCAR. And it was unreliable. The reason it went to NCAR is because they were using the waste heat from the machine to warm the buildings in the winter. But as I said, it had this air leak or

something-- leaked air all the time. Always-- memory's unstable. Eventually, Seymour removed the memories, which were American made, and installed--

Miya: Toshiba memory.

Michael: It was Toshiba memory. Yeah. And the memory errors went away. And probably you'd have to say that Seymour was a bad businessman, because he didn't gain any financial advantage when this thing was a success. And you know, it was "loaned" to NCAR, and they made it work. But nobody ever paid any attention to what was done. That's sort of sad in a way-- but--

Miya: So what do you know about the Cray-3?

Michael: Well, just what I'm telling you now, you know. The Cray-3 was in an octagonal, freestanding kiosk-- maybe 30 inches high, 35-- I don't-- it's hard to remember. And--

Miya: You saw this?

Michael: Yes.

Miya: In Colorado Springs?

Michael: I saw the Cray-4 in Colorado Springs. I saw the Cray-3 at NCAR. I -- I don't remember everything about it, but the-- it was an impressive machine. And you know, it-- the packaging was just incredible.

Miya: What was incredible?

Michael: Well--

Miya: What was awesome.

Michael: Diameter about the size of this table, and octagon. And the entire-- I think it was 62 process-- no, 32 processors, each with a couple gigabytes of memory in it.

Miya: No. Four processors. Octets only had four processors. Right? It's supposed to have 16 processors. He announced that to the world. But he never completed 16 processors.

Michael: No.

Miya: You're supposed to have octets. Right?

Michael: I don't know Gene. You know, it's-- this is all after I retired. Okay?

Miya: Yes. But you were privy to all this. You had a hardware manual.

Michael: No. I-- I dealt with Chuck Breckenridge and Bill Sembrad [ph?], and we were just, you know, exchanging information about things there. I can't remember that there was any technical commitment one way or the other.

Miya: What do you remember about the Cray-4 that you saw at Colorado Springs?

Michael: Nothing. Well, again, you know, it was shrunk by a factor of four. And I remember thinking, "Those things can't be wires. They look like spider webs." Okay? But they were wires.

Miya: Okay, we've got to wrap this up in five minutes George. The Cray-3 had a two-nanosecond clock cycle. The Cray-4 had a one-nanosecond clock cycle. Right?

Michael: Right.

Miya: Do you recall any other architectural-- ?

Michael: Well, yeah. They had a Cray-5 and -6 on the drawing boards also.

Miya: Tell us about the Cray-5 and the Cray-6.

Michael: Well, I don't think that ever would-- the 5 would've seen the light of day, but he had it designed.

Miya: What was it?

Michael: It was just measurization over and over. And the Cray-6 would be lost on the top of this table here. It had the same _____ for it.

Miya: Well, isn't that the SRC6? Or is-- George. Isn't that the SRC6, or is that the Cray-5?

Michael: Uh...

Miya: How come we haven't heard anything about the SRC6?

Michael: I don't think it was the SRC6. I don't believe I've ever been inside the SRC, and I don't think it was built when I was at Colorado Springs talking to these guys at SRC. I remember him saying, you know, that Seymour had made a separate deal to be shown the open kimono at Intel. Okay? It didn't have anything to do with reliability I suppose. This stuff was available.

Miya: So let's see-- the Cray-3 was a two nanosecond cycle time, the Cray-4, as I seem to recall you saying-- Hey-- you still with me?

Michael: I can't remember.

Miya: Was supposed to be the size of a NeXT Cube, one foot on a side, one nanosecond. Right?

Michael: Less than a nanosecond.

Miya: Less than a nanosecond?

Michael: It was either a half or a quarter.

Miya: That wasn't supposed to be the Cray-5 or the Cray-6 instead?

Michael: I think there was a Cray-5 identified, but I don't think he was every going to build it. He would work out engineering details on that, and you know--

Miya: The Museum has-- George-- the Museum has part of the Cray-4. Jerry gave it. And it's supposed to fit inside the NeXT Cube. And it was supposed to-- you were saying you could have various numbers of boards and things like that?

Michael: I thought it was 16.

Miya: Was it supposed to be Fluorinert cooled?

Michael: Well, I'm sorry Gene. It's past my time, so I-- I could claim an unfamiliarity with it, maybe even a jocularity.

Miya: I recall that he got rid of the local memory in the Cray-4 to go back to B and T registers. So were B and T registers in the Cray-5 and the Cray-6?

Michael: I can't answer that. I don't know.

Miya: So weren't the Cray-5 and the Cray-6 supposed to be put together with Intel Pentium processors or something?

Michael: Yeah. There was a fourth.

Miya: No. The four wasn't supposed to be Intel Pentium processors. It was his own.

Michael: I said there was three, and that should be right. There was three Serapes [ph?] on the cement there.

Miya: Three Serapes? Three Serapes?

Michael: Well, no-- you know the thing that the Mexicans wear-- on the steps.

Miya: For the Cray-3?

Michael: -- of SRC.

Miya: Uh-huh.

Michael: I'm being incoherent.

Miya: I know you're being incoherent.

Michael: Yeah.

Miya: So. All right. Well anyways. If we've finished this, we've got through hardware. Okay?

Michael: Well--

Miya: Yeah?

Michael: I don't know what I can say about the software, except that, you know, what animates it is experience with how it's used. It isn't just elegant or computer scientist stuff. Its how it's used that determines what you going to do when you write it. Well, all right-- that's one thing. The other thing is, by abandoning the basic philosophy of LRLTRAN 2, we get sort of punish a bunch of people that otherwise wouldn't get punished.

Miya: Uh-huh.

Michael: That didn't come out the way I wanted it.

Miya: Okay. Well, anyways I could-- if and when we resume this, I'll bring up the Cray-4 or the 5 or the 6 again.

Michael: I don't know anything about them.

Miya: Well, you mentioned stuff to me, which is just churning throughout your mind.

Michael: Well, yeah its--

Miya: So okay. You know, we can end for today and--

END OF SESSION 5

START OF SESION 6, September 15, 2006

Miya: Okay. This is September 15 [2006] just before noon and we're hoping to finish interviewing George Michael today. So the last part of the interview we had talked before about the latter models of the Cray-1, Cray-3, Cray-4, etcetera. And we have to do some preparation for the Cray presentation next week. So here's what I have to remind you. You said about the Cray-2, shrunk the hardware by a factor of four. It was faster but had the same problems as Cray-1. The memory was too slow. Made the memory banks do-- didn't do some of the tricks that were available. Everything was very simple. Rounding, pioneered _____, how to get cooling done, immersion. That was a dramatic statement. Sadly, though, the memory-- the disparity caused problems and there were software algorithms involved. You mentioned NLTSS. You gave us Donna Devarona [ph?] and Jim Minton [ph?], specifications done by Dick Watson [ph?], Jim Fletcher [ph?], Jed Donnelley [ph?], Sam Mendecino.

Michael: Jim [ph?] who?

Miya: Jed Donnelley [ph?].

Michael: Okay.

Miya: Right?

Michael: Yeah.

Miya: John Fletcher [ph?], Sam Mendecino [ph?], Rob Crawley [ph?]. There were these Friday afternoon meetings on the language MDL from Los Alamos, which I guess you guys were writing NLTSS or DL. Bob gave the job to-- Bob Crawley [ph?] I guess gave the job to Dick Watson [ph?]. He developed the links, the LINCS network protocol, which came in late.

Michael: Well, I wouldn't think it was Bob [ph?] giving something so much as it was Sam Mendecino [ph?] re-juggling the appointments or assignments. Okay?

Miya: And you said since that point in time, the Golden Age of supercomputing was tarnished. Faster components don't guarantee a faster computer. I guess the machine eventually-- one of the machines went eventually to NERSC. Disk striping was tried with the Cray-1 but it's not standard operating procedure. NLTSS had disk striping. So let's stay on the Cray-2. Are there any other things you want to say about the Cray-2?

Michael: Well, again, he was pioneering a new technology in order to achieve what he was after. You know, and he said two things about the Cray-2 that I remember. One is that the arithmetic stuff hadn't been able to keep or I should say the memory chips hadn't been able to keep up with the arithmetic logical stuff. So it was always wait, wait, wait while getting stuff out of the memory. He tried multiple wave interlaces, and the cooling that they used there was immersion. Okay? And it was a fairly dramatic machine when it first became public. And under certain conditions, it really wasn't that much faster than a Cray-1. But it was bigger and it is certainly an evolutionary kind of machine.

Miya: So was the machine a success?

Michael: Probably more than Seymour intended it to be, yeah. Seymour was concentrating when I last talked with him, you know, all the damage had been done by then. But he had been thinking about the Cray-3, 4, 5, and 6. And I saw the components that he was considering for each of these models. And to get from the Cray-2 to the Cray-3, I don't remember if it was a dramatic change in the cooling technique to get from the 3 to the 4, the cooling technique was to spray the coolant onto the parts and have the heat vaporize the coolant. You got a little extra boost that way and then condense the heated fluid and pull the heat out and stuff like that. Seymour's problem with the Cray-3 was in a sense, he broke one of his own rules, which is to not use unsettled componentry, things that were still in flux. And until he switched memory chips from what he had to the new chips that I believe were supplied by Toshiba, he was having memory troubles all the time. But as soon as he switched, the memory troubles went away and the Cray-3 worked beautifully.

But it was such a struggle that I think in a way its window of opportunity had closed already. And that shows up as, you know, they're bankrupting him and foreclosing him and all these other kind of things. Many of us who were totally ineffectual in helping out on money and stuff like that, we tried, you know, yelling at appropriate people in Washington to supply the necessary funds to keep Seymour going, but they didn't really do it because the difference between, you know, an administrator who's coming from a deep feeling of experience and one who's a special kind of a job expert is too much. They always worry in a sense about their own stuff, and so they really don't like to take chances. And it means taking a chance in the case of Seymour. No one ever felt that taking a chance on Seymour was a bad idea, but the people who were the administrators and stuff like that in Washington just didn't understand.

Miya: Administrators like where?

Michael: Well, National Security Agency on one hand and the Department of Energy on the other. Department of the Pentagon for that matter. I wasn't privy to any of that real inside dealing and wheeling. I talked with various people who assured me that they had tried to get money to Seymour in time to stave off bankruptcy, but they didn't do it. So even though when bankruptcy was declared, the Cray-3 was successfully running and producing good results. It wasn't enough to save his company.

Miya: You said you mentioned you saw the actual components for the Cray-3, 4, 5, and 6.

Michael: Yeah. Yeah.

Miya: I mean, most people have never heard of a Cray-6, much less a Cray-5 or a Cray-4.

Michael: I understand.

Miya: Well, you know, the question is what components exactly did you see?

Michael: Everything. You know, the logic chips, memory chips, stuff like that. Remember Seymour had made a special deal with Intel to get to see their advanced development stuff before it was made public, okay? So he knew what kind of things were coming along that he could use. The characteristics of these machines very roughly, they shrink by a factor of four and increase their speed by a factor of four in each

case. So the Cray-6 was something like a tenth [of a] nanosecond cycle time; something like that, okay? And to quote my friend Norman Hardy about that, when you look at the wiring cable, the cabling that was going to go into that machine, Norman said a spider would be proud of it. You know, the web. It was very small. It looked dusty more than wiry. But I would be perfectly willing to take Seymour at his word when he said it's going to run at such and such a speed and produce so much heat and this spray cooling technology which he had pushed on would work for that. The last one, it would work that way, and then he would have to invent something new again. But he never got the chance to do that. I think it's just generally true and there are people in this country who tend to know what we really lost when Seymour was killed. But the people who are up in the echelons of power, you know, he was just another supplier. Just another inventor no better than anybody else. And I think that's probably not true. He was really, really an important designer. He was a genius.

Miya: Could you stop a second. Okay. You mentioned the Cray-6 as having a tenth of a nanosecond cycle time.

Michael: Something like that, yeah.

Miya: And from the last time we spoke, you said the Cray-5 had a half-nanosecond cycle time. The Cray-4 had a one nanosecond cycle time. So what happened to all these artifacts; these little…?

Michael: Some lucky entrepreneur got the parts that Seymour had collected, and they're available actually if you want to buy them from somebody. The person I dealt with was Bill Simbrat [ph?] who was a Cray salesman, and Charles Breckenridge and they-- I think that some of these parts actually made their way into the DigiBarn in the Santa Cruz mountains.

Miya: Right. We had talked to Bruce [Damer]. What about the Cray X-MP and the Y-MP and the C90 in line. Do you have any comments to say? Anything about those?

Michael: No. They all-- they were, you know, ragingly successful machines. But their impact was in a sense attenuated by the ridiculous speed of these laptops and things like that that were coming along. You need a lot of speed if you're going to run a whole group. But if you're just running yourself, well you're only responsible for yourself. The kind of speed that we get out of a desktop or laptop machine is quite adequate, I think.

Miya: Now, you mentioned Bill Simbrat. One of the things that in terms of my knowing you that was amazing to me was the fact that you got access to these advanced architecture manuals before everybody else did. You got a manual from Bill for Cray-3 and I guess the Cray-4. So it's like, you know, how do you come upon these?

Michael: Well, to the extent they had them, you know, you got to see them. After all, Livermore and NSA were the two technology pushers that Seymour used. In some cases, when he listened to people at the lab, I don't think he listened to the right people. But those things could be changed and corrected and stuff like that. I don't see anything unusual about that.

Miya: Did you have anything to add, actually, since we were mentioning the X-MP and the Y-MP a year ago about Steve Chen?

Michael: No. I had less to do with that than any other machines at the lab, as a matter of fact. They were big production machines. And I would say the vast majority of people came to them through, you know, translators and compilers like some fast version of Fortran or something like that. They didn't come to it by a deep knowledge of how the machine worked. There were people, notably the Harry Nelsons of the lab who did know how the internals of the machine worked. I think that he's a deceptively smooth person, you know. You could learn a lot about the machine just by having conversation with him. He's an interesting guy.

Miya: Harry?

Michael: Yeah.

Miya: Yeah. So let's see. Essentially the Cray-3 was sort of a shrunk down Cray-2, but with four times the number of processors? Because only…

Michael: The packaging was quite different. But, you know, I don't know how you want to represent the Cray-2 in terms of packaging. But the Cray-3 was like an octagon. And you stood on top and looked down on it. And each octant had a computer or more; two computers in it. And it was arranged, you know, so that the spraying so that this Fluorinert onto the hot part could be done and it would condense the juice into a sump and bleed out, restart the cycle.

Miya: The 3 or the 4.

Michael: 3.

Miya: Okay. Because I never saw the 3. You did get a chance to see Cray-3.

Michael: Yes, of course. Yeah. We saw the Cray-3 and, you know, the mock-ups of some of the Cray-4. I don't want to dignify what I saw with, you know, anything like the finished product. It's just that Seymour, you know, how he worked. And he settled things with simple experiments and things like that long before they put the parts together.

Miya: Could you describe some of the experiments? Did they ever show you any of these experiments?

Michael: No. We only had one real opportunity to visit him in his company called SRC. Pat Savage and I went there and visited him. And you know we spent the entire day there talking about this, that and another thing. And looking at how the Crays were being built by the Macintosh's and so forth.

Miya: Right. That's the famous quote of…

Michael: My other machinist.

Miya: Yes. He was using Crays to design-- no, Apple was using Crays to design Macintosh's and Macintoshs were being used by Seymour to design Crays.

Michael: Well, the Macintosh-- the Crays were being used at Apple to design environments from what I would hear. They never were as forthcoming as Seymour was.

Miya: And I also seem to recall if I remember right, you said something about the Cray-4, the one nanosecond Cray-4 with one CPU. And the Museum has boards that his widow gave the museum. And it was supposed to fit in a one-foot cube like a NeXT box.

Michael: Yes. It was a fact, before it shrunk, was still an octagon. But it was shrunk by a factor of four.

Miya: Did they have a case for the thing?

Michael: I don't remember that that well. You know, the largest components or the largest volume that users in one of these Crays, any machine, is the cooling stuff, okay? So you have to arrange it to get the juices where they're supposed to go and pump when they need to be pumped and things like that. So a large amount of how you design things is decided by how you're going to cool it. In the case of Seymour a large amount of things that decide decisions about design come from timing. Okay? And you know, he-- it always surprised me that he was a great, great user of Boolean equations to describe componentry in the machine. And he was very good at it.

Miya: Can you say anything else about the Cray-4 that you saw?

Michael: No. I don't remember anything else. In fact, you know, the Cray-4, 5 and 6 are so much in the distance as far as when they were going to be actually produced, they would depend on the parts that Seymour thought he was going to have actually coming into his control so he could use them. And in a sense, you could expect about now the Cray-6 would be-- would have been, you know, showing up on for sale things instead of-- well, we lost all that because he was killed.

Miya: Right. Do you recall anything about the Cray-5? All I have here in my notes from last [time] is that it had a half-nanosecond clock cycle. And he said the open Intel kimono, you said.

Michael: Well, that was in general he said, you know, the agreement that he had with Intel was Intel showed him their advanced development stuff, okay? It's a very, very useful thing for Intel to do. And they did it. Now, the Cray-5 I can't even say that you know, I have this morally sound feeling that he was actually going to produce Cray-5s for sale. Just like there was a 6800 and a 7800, machines that never even saw the light of day so to say except they helped him work out what he was going to do. I think that the Cray-5 was in that same category. The Cray-4 and 6 looked to me like they were going to be real machines if he had the—Cray-5 was just, you might say an engineering model along the way to test out ideas. He had some strange ideas which I can't reproduce, but it had to do with funny phasing of the power supplies.

Miya: Well, what about concepts like reorganization of memories in order to get more speed out of them?

Michael: What he did on memory, I don't know if this ever was going to change, he provided a 512-way interlace. And that's part of the SRC 6, which is being delivered to some places, though I know nothing

about it. Seymour left a fairly good crew of people at SRC who carried on and are working on machines and…

Miya: Are supposedly using Pentiums. And they're supposedly using Pentiums from Intel?

Michael: I don't know what they're doing. All I know is what I just said; the 512-way interface for the memory, to beat latency. That, in it's own way, is a smart thing because it allows you to take advantage of certain tricks in the compiler so that you're sure that you're going to get the memory cycle when you need it; things like that.

Miya: So let me quote to you, and I want your reaction from this book.

Michael: Boo, hiss.

Miya: Okay. You're ahead of the game. James Bamford here, quoting. My good friend John actually. _____. "The rules changed when it became clearer that Cray Computer Corporation wasn't going to make it," said John Mashey Director of Technology at Silicon Graphics. It's like watching.

Michael: I couldn't hear you.

Miya: "The rules changed when it became clear that Cray Computer Corporation wasn't going to make it"; John Mashey from SGI. "It's like watching your favorite quarterback, who won the Super Bowl many times, but it's not 1976 anymore. His knees are gone and those 300 pound defensive tackles are fierce. While he keeps getting up, it's agonizing to watch, and you really wish he would really quit on a high." What's your reaction to that?

Michael: Well, to be charitable, I would say that that would be a comment made by a person who'd never written a program or ever used a Cray or understood its symmetries and its power, things like that. You know. I had more of an abiding faith in Seymour. I thought he was a great person, was fun to talk with. And I think he showed a consistent cleanliness in thought far, far longer than most designers. And the Cray is as beautiful as the CDC 1604 was.

Miya: You mean the Cray-1.

Michael: Any of the Crays. You know, they're all-- they're pushing technology. I heard some people say Seymour Cray is the Evel Knievel of computer design. He's up on a curve pushing, pushing. There are some interesting comments that he made at a conference that Ted Sterling had pulled together for how to handle the demands of a petaflop machine.

Miya: Oh, Tom Sterling.

Michael: What'd I say?

Miya: You said Ted.

Michael: I'm sorry.

Miya: I know the book. MIT Press. [*Enabling Technologies for Petaflops Computing*].

Michael: Yeah. And I don't think that people listened to him, you know? Basically what he said, if you want to get fast, you have to get close. And there's an enormous amount to be learned about nanoengineering before we fasten on what we're going to do for the bigger machines. And we ought to be doing some of those studies.

Miya: That was lay Copper using bacteria?

Michael: That was a conference that they held. And I think there were other people there who again, largely, just aren't realistic. You know, they talk about well, we'll have to clear the memory every now and then, but the memory is going to be a terabyte and how long will it take the terabyte to clear and be replaced? That means what kind of disks do you have. And you know, when they start talking in terms of thousands of disks in order to achieve some of the bandwidth or some of the capacities that they were talking about then. It's just hard to believe that they stayed anywhere near reality. I think they should have been far, far smarter to stay close to Seymour on those ideas. I think, you know, maybe it's the American sort of sickness that if two cats can do a job this way, nine cats can do it better. That's just not very true always. And even if it is, there's always a place for advanced design and clean thinking and that sort of stuff. Seymour was excellent at that.

Miya: That's Fred Brooks law. The nine cats.

Michael: I don't remember that, but alright.

Miya: No. Actually I like the way…

Michael: I think in a sense you can blame him for OS 360. So I don't think you want to listen to him at all.

Miya: He apologized for that, don't you remember? Yeah. He did.

Michael: No, I don't remember.

Miya: I remember him apologizing for OS 360, which is nice. Especially JCL. But anyways.

Michael: I had the misbegotten experience of having to try and use that stuff when I was at Haverford. That was real, real sloppy stuff. And not very good at all.

Miya: I think the entire computer industry is a neglected child. That's okay.

Michael: The politics at those places forced us to have a 360 [Model] 44, which in itself is a maverick. It doesn't really fit into the 360 line. And they made special alterations in the operating system and stuff like

that. But it didn't change the fact that it would take a couple hundred hours to get the thing assembled. You know, this-- what do I want to call it? Mulish mind of forcing cards to use for job control language and things like that. That stuff is just dumb, okay?

Miya: Guess what?

Michael: It made IBM rich. I can't help that.

Miya: That's true. My next line here is: Describe the debates over interactive computing versus batch processing. So yes, tell us about this, George.

Michael: No. No.

Miya: Yes. Tell us about the mulish minds about, you know, cards; punched cards.

Michael: Look, Gene all I can say is this; if you are on the loose or light-footed appreciation of machines and stuff like that, you realize that they are a hell of a lot more than adding machines, okay? That you can do enormous things with them. And the people that controlled things like the 360 had some fairly beautiful ideas, but also some very weird ideas. You know, I met several of them, and I must say they're great persons. But I don't think the 360 was a paragon of good design or something like that. It had sort of a nice impact in the business world, but I don't think they really cut it through to the scientific world. Who knows? You know, there's no simple statement to be made. Computers are more than adding machines. And the persons who think about them that way are likely to come up with novel things of what you can do with a machine. And we also know that when you're going to do something more than once, having a computer is the right way to go. Fine. I don't think that we did do well with the 360.

Miya: Tell me a little bit about what you view as the importance of academic computer science departments.

Michael: No.

Miya: No.

Michael: The principle importance of universities and colleges is production of students. Trained persons in various disciplines. It is possible because of the nature of computers and cybernetics in general and stuff like that that a large number of the things that are important in these fields will be worked out by academics. But the academics in general don't have to concern themselves too much with robustness, or timeliness or simplicity enough so that persons who are going to have to use it can make use of the things. And so in lots of cases, they produce things which are very nice. Kind of beautiful in some respects, but they're very limited. You know, I think of MIT's compatible timesharing system [CTSS], where they built in, understand, limits in all the tables. Well, it may have been necessary because of the nature of the machine they were using, but they spent $6 million on RPQs for the 7094, okay? It carried an extra core just so you could swap environments; one to the other and go fast. But that's not the way to build a timesharing system. Okay?

Miya: What were your famous comments about the certain architecture in the '80s called a VAX? Being comparable to drugs or marijuana? What was that you said?

Michaels: I was saying to our academic colleagues that the VAX is sort of a form of marijuana to the academic crowd, okay? They think because they can do it on a VAX that all the answers and problems have been supplied. And we were at pains to show by example in many cases that if you have a real computer like a Cray-1 or a 7600 or a 6600, you can do things that wouldn't occur to you with just a VAX. You know, and they're trivial examples in some cases which I'm not going to cite. But when you are trying to do things, the one thing you don't want to have is real limitations forced on you by the nature of the hardware. Because it doesn't allow so many tables. Or it is fast enough to take a square root when you need it or this or that. That isn't the way you're going to come up with ideas about what to try and how they can work and so forth. You know, it's perfectly a sinful waste in a sense. But say the 7600, is an incredibly fast machine if you know how to use it. And there were people who specialized in that sort of thing about instead of running out of the small core, they run out of the large core array and do things that were not originally planned and provide very good results. You could run the 7600 so that it far, far exceeded the scan conversion rate of an ordinary television camera, which, you know, is running 20 megacycles or something like that.

Miya: But should the users have to control the stuff of a large memory and a small memory?

Michael: No. I'm not saying we should have to do it. I'm just saying that given that you can do these kinds of things, there are no artificial barriers being raised up around you because you can't do this or you can't do that. If you want to do it, it can be done. The big machine permits you to do it. A small machine does not permit you to do it.

Miya: So would you say the 7600 example between the difference in the memory are the-- since it doesn't have a cache, technically, is one of your trivial examples you end up telling us one of your trivial examples?

Michael: Well, it was in scan conversion. You know, you start from a vector list and you want to convert that vector list into a raster. And if you do it instead of one thing at a time, but do several things because you've go the registers there to handle it, even though they're out of sequence and stuff like that, they all fit. And as a result, you can overrun an ordinary scan conversion camera. It just couldn't keep up. But you could scan convert at the rate of maybe, oh, I don't remember. You know, it's, you know, you see a camera does things that say 60 frames a second or 60 fields a second.

Miya: Okay. Cat conversion?

Michael: Well, yeah. Whereas, you know, instead of being 60 frames or 60 fields per second, you could do several hundred fields per second. So you could take that extra capacity and so to sale, wisely use it either to provide incredibly fast service to one person or a lot of service or service to a lot of people at no real expense. You just had to know how to program the thing. And there were some of the people, Bob Crawley was one of the best examples that, you know, they'd learn how to use the 7600 for things it wasn't designed you might say for. And it always ran much faster. That's about all I could say.

Miya: Yeah. So that's what you're saying about academic computer science departments and the like. And VAX is just marijuana?

Michael: Well, I think the most important thing in the academic trade is that they supply students.

Miya: Right.

Michael: So, graduates who are well-trained.

Miya: What about the importance of a computational science department as opposed to a computer science department?

Michael: Well, I think it's probably more important if you, you know, have the right definition of computational versus computer. I'd say computational science is more important than computer science, okay, though they feed one another. You know, I think maybe the simplest example of that is the business of having-- of preserving or stopping the propagation of error in your numerical results. If you just start with the simple truth that A times B is not the same as B times A. A plus B does not equal B plus A; things like that. And you start from that and in computational science one learns that given these strange idiosyncrasies of a machine, they can be circumvented or finessed somehow when you're trying to run a big physics program or a big chemistry program or something like that. And it's true that that can be thought of in computational and computer science, you know? After all, where would you put Hamming? I would say Hamming was a computer-- computational scientist rather than a computer scientist. And in fact, he always argued that he didn't want to have anybody who had learned Fortran. He wanted people that could think, he said. Well, you know, the computational point or approach tends to be maybe more experimental. More you know, example-oriented.

Miya: Empirical. Empirical.

Michael: Yeah. And at least from the beginning of a science, maybe that was important. I don't know.

Miya: It's a long topic, I know.

Michael: Indeed.

Miya: For long drives. Okay. One of the great things I think you did was because of the Salishan insignia. Actually, you know, why don't we just go with the next-- we'll stop here and go ahead and change…

END OF TAPE 6.1

START OF TAPE 6.2

Miya: Okay, George, one of the great things is you push across the cost of the balance, that you can't simply have a fast CPU, but you had to have adequate amounts of memory. And also fast I/O as well,

too. So one of the areas that is severely lacking in high performance, or high-end computing policy, was that of storage.

Michael: Was, yeah.

Miya: That you guys had of all the laboratories, the national labs and the research centers, a laboratory devoted to storage as opposed to high-performance computing. So I mean the one that Dick Watson-- I was never involved. Tell us about the short-lived National Storage Laboratory.

Michael: Look, I think it was Central Intelligence Agency. They funded a National Storage Laboratory. It was based at 3M. And they were telling people, for instance, how to handle a tape that'd been stored for 20 years in order to be sure you could read it. How to handle disks to make sure the bits didn't wander and stuff like that. I do not know what happened to that in the long term, but they used to meet regularly, and send out notices to everybody who was interested on the progress they were making in studying magnetic storage. There was a guy down at the University of Santa Clara, Al Hoagland, who's heading up an institute for magnetic storage also [Magnetic Disk Heritage Center].

Then there's this thing that Dick Watson and Bob Koine <ph?> of IBM, cobbled together, which was the National Storage Laboratory. It allowed manufacturers to bring in devices into a real working environment, and see how they performed. I think that Dick Watson is an absolutely great person, and whatever he tries, you know, it should be very good. I don't, on the other hand, know that they got all the results they were after in this thing. We do have today, capacities that're more than adequate for the foreseeable future. The thing that we don't have is speeds. Now we can get some of those things by multiple heads on multiple spindles and stuff like that. And to the extent that the array technology will allow it, you can run as many as you want without error. So in principal, if it's very important, you could be interested in building as big a storage facility as you need, which led some of us to think about a warehouse. You know, if some of the computational facilities that we have right now are to be sequestered and kept away from the average user, for their benefits, storage was one of them. And storage would have, in the sense of a warehouse, would have taught the idea, "Well, we don't trust anybody. If you want something you ask for it and we'll get it if it's there." And it doesn't have to be in a fixed format, but we have to know ahead of time that you're going to use 7-track tape, or 6250 bpi, or whatever it might be. And these tapes and these disks sit in a controlled environment in a warehouse. And you just get them out like going to a library, and checking out a book. Once you got the bits in your hand, and they're over in your computer, you can do whatever you want with them.

Miya: Storage like a cargo container.

Michael: Yeah, all right.

Miya: Like Brewster with his Internet Archive.

Michael: Yeah, to some extent, yeah. I don't know. I haven't heard very much about the Brewster Kahle event, so.

Miya: November.

Michael: You coming?

Miya: I don't know.

Michael: Well, we'll check them in November, yeah.

Miya: Right now they're meeting for lunch, talking status. I know this for a fact.

Michael: <laughs>

Miya: Some of the next questions, until we get to the people, are sort of leftovers from prior discussions and interviews, okay? So, they're going to seem out of sequence. As if they haven't already.

Michael: I'll try.

Miya: So you have this photo, which I think was shown at a retirement roast for you, which has a car standing out on a salt flat in the middle of Nevada. And this 80-foot tall blown apart tower. Are you familiar with the photo?

Michael: The photo I have?

Miya: Yeah, tell us about that. That's Livermore's first test, right?

Michael: One of the first ones, yeah. I don't know that we can talk much about that without bumbling into the security issues of classified issues. Basically it didn't go off the way it was supposed to have gone off. If it had gone the way it was supposed to go, one would expect the entire tower would've been vaporized. And it didn't. And in fact, most of the tower's still there. So something was wrong. And I think that was, in the good sense of science, more was learned from that failure than a dozen successes. So atomic testing was such a funny thing, you know. We don't like to have failures there. But they do teach you more than successes do. And if we had continued on, you know, there was an entire technology developing for getting information out of a hole, underground testing. And it's just different from atmospheric testing, but there could be a mapping that goes from one to the other, and that was being learned. As it is now, people don't want to test at all. And it may be the best thing, because if we have many, many examples of, you know, scientists getting things built and doing things scien-terrifically that were good and so forth, and then they fall into the hands of a bunch of schmucks, you know, in government and stuff like that, and things are ruined.

Miya: I like that word you just used.

Michael: I just made it up.

Miya: Scien-terrifically. Yeah, yeah, I know.

Michael: Oh, scien-terrifically, yeah.

Miya: Yeah. Well, actually the reason why I made this note was more of the fact of Livermore's early days. It was the one-liner you said that the director at the time, I guess it must have been Norris Bradbury said about Los Alamos, looking on at Livermore. What did Norris Bradbury say something like?

Michael: I don't know.

Miya: He said something to the effect of, "When you're done with our tower, can we have it back?" or something like that?

Michael: <laughs> Oh, yeah. I didn't know we were on that subject, but yeah. Look, Los Alamos to me was sort of more like the IBM of atomic, of the atomic world, okay? And being tried and trusted and all that sort of stuff. But sort of short on really juicy ideas and really interesting ways of doing things.

Miya: So if we were to talk to some of my friends at Los Alamos, would they disagree with you?

Michael: I don't know that, because I think that the people at Los Alamos, in general, are very, very nationalistic in the sense that they think that everything that Los Alamos does is great. And it's absolutely true. There's a lots of very good things that go on at Los Alamos. Not the least of which is a visiting scholars program in the summer that didn't last very long at Livermore, I guess, maybe one or two years, and that was it. I think it's very important. In fact, if I wanted to meet somebody in one of those conditions, I would try to get down to Los Alamos during the summer because they would be there, you know, in the summer institutes of one sort or another. But you know, there's some data, too. Timesharing didn't really take off at Los Alamos until they took the Livermore timesharing system. And it's true they did a better job of documenting it, and publishing the manuals and stuff like that. But there's a difference between doing that and creating the thing.

Miya: Yeah, the implication on Wikipedia is that Los Alamos developed CTSS, for instance.

Michael: I hadn't looked.

Miya: I did. And it implies that Los Alamos did CTSS, not Livermore.

Michael: Well, they're that way about lots of stuff, I'm telling you.

Miya: Oh, yeah, I know. They're the same problem. Okay, let me ask you a couple of questions from a unique opportunity-- a series of unique opportunities that you gave me in 1990 or so, which was after the cold war. You invited me as well as some other friends to Livermore Family Day. Which is not something that anybody is allowed to do. Can't take photos. Can't bring personal electronics and stuff like that. But we were actually allowed to wander around.

Michael: Who went to that? You and...

Miya: Let me think. Well, the third time Ed came with us. Ed Hodge.

Michael: Yeah, okay.

Miya: But the first time...

Michael: Just you?

Miya: Well, let me think, but Jean Shuler <ph?> was there, and Carol Hunter were there.

Michael: Well, she works there.

Miya: Well, right, and Carol Hunter were there.

Michael: Carol Hunter, both Carol Hunters worked there.

Miya: So we were all part of going there. I think I ran into Steve and others as well, too. But do you want to say anything about the place you worked? Let's see, I put here for notes. The "trans-uranium facility." I think you said something like every room has an external door, or something like that. Or can you not talk about that.

Michael: I don't know anything about it, I guess.

Miya: The trans-uranic <ph?>, or the heavy element facility?

Michael: I've never been in the thing, I think. I don't know.

Miya: All that you told me was that every room has an external door. Like to try to be able to escape or get out if something bad goes on.

Michael: I don't remember that either, I'm sorry.

Miya: Okay, nuclear chemistry lab?

Michael: Yeah, I've gone to lots of chemistry labs.

Miya: Well, the one that's near Building 111. So you didn't work there? You didn't know to look...

Michael: No.

Miya: I had "tank, armor, and tactical nuclear combat." You might not be able to say anything about that.

Michael: Who?

Miya: Tank, armor, and tactical nuclear combat. Can you say anything about that?

Michael: Well, sure. We had modeled Monte Carlo models that essentially tracked neutrons. And you could put in a geometry that described the tank to any degree of precision you wanted and see where the neutrons went. And you could put in shielding and things like that, and see how it would alter the distribution of neutrons and stuff like that. Basically, if you're in a tank, and you undergo a neutron attack, you're cooked. You might be able to stave it off for a week or so, but you're cooked, I think. And there's a certain kind of finality about anything you deal with when you're dealing with a weapon, a thing. We used to have no precedent to describe the incredible force that those things represent. Just even get a glimmering of it out in the Pacific at the test site they built a thing called a bunker, which houses the high-speed cameras, and they look at ground zero. And the bunkers are made with, in some cases, enormous doors that can't be moved, you know, except by machine, and the concrete that holds everything together is six-foot thick, and there're all kinds of shielding that is put on the thing. And after the test, you can't find anything. It's all gone. You can find a broken box of concrete thrown three/four-hundred yards away, 100 yards away. It's an awesome force.

Miya: Do you want to say anything about other parts of the lab like the HEAF, the High Explosives Application Facility, or MFE in the civilian areas? Or SHEVA, NOVA or NIF?

Michael: They're not very complimentary, so I don't think so.

Miya: Oh, c'mon George!

Michael: You know, I will tell you this way, I felt that the initial thrust that was being made in Project Sherwood, this is about the controlled release was strongly experimental. There was hardly any numerical computation that preceded their experiments. That's changed a lot with the help of SID. The MFE, Magnetic Fusion Energy Computer Center was created, and it took our timesharing system <clears throat> and it built a central thing to which everybody had access, and all the subscribing people had PDP-10s or 20s, and VAXimus and stuff like that. So they could, you know, do their small things on the small machine and then ship it over to the big machine. What was found very early on is that nobody fiddled with any of those machines, you know, that're right in your own bailiwick. They would go to the big machine. Things like editing a text file so that you can produce a new program. The idea was that you'd ordinarily do that with a DEC editor. When you were satisfied with it, ship the editor over to the big 6600 or whatever it might have been, and it'd run there. What was always done was that people developed the stuff on the big machine, they didn't use the small ones at all for this kind of stuff. You know, the smaller machines were used in almost all cases as experiment controllers. Again, you know, I mean, it's a quiet way of showing that machines, big machines are incredibly powerful, and you can do an awful lot of things that you hadn't otherwise thought of. You know, <clears throat> running around the whole world, you got people who never wrote programs, but they're in charge of the computing facility of something like that. And they're running things so as it makes the thing look slick or something like that. That, to me, is nonsense. What should be is that you run things so that the people who are using it can really, really get their work done without any distractions or any untoward distractions. But a lot of the administrators just don't understand that.

Miya: Yeah, we covered that in the security topic. I was impressed after you hired my friend, Steve, away from me that...

Michael: He wasn't your friend.

Miya: <laughs>

Michael: We brought him up here initially, and we had several people at the lab interviewing him before he came over here to be interviewed. And whether you liked it or not, the lab is a hell of a lot more interesting than this place as far as...

Miya: No, no, I mean down south. Not Ames, but JPL.

Michael: All right. I don't know.

Miya: No, no, I mean we'll get to that. And we'll get back to Sid here in a moment. The thing that impressed me was to learn that you guys had DEC LSI-11s as store stock items. You could simply go to store stocking and draw $7,000 LSI alumni.

Michael: Well, we had the LSI-11s, and PDP-8s. LSI-8s, whatever they were. A storage item. It was anything enlightened, it was a simple arithmetic thing. You know it's cheaper to do that than it is to try and build a special controller every time you want to run some sort of an experiment.

Miya: It shows that you guys had money.

Michael: Huh?

Miya: It shows that you guys had money!

Michael: I give you my standard remark again. We had four things. We had money. We had real problems. We're not looking for Fibonacci numbers, or the next prime or something like that. We had very good people, and perhaps the most important thing, we had a management that knew enough to stay out of the way. You know, tell us what's to be done, and then let it happen, let it get done. That's the recipe for a good laboratory, but most people fail utterly to get that kind of message sunk into their heads.

Miya: Can you say anything about the labs? I think it was a first success, which was a Polaris warhead, which was designated a W-47, or is that too sensitive?

Michael: Well, <laughs> what do you want me to say? You know, it was a success because it was a project, rather than just a bunch of people producing components that are going to be integrated later on. I believe it was Jack Rosengren <ph?> who was the chief physicist on that job. And there were a bunch of people who had big programs that ran on the various machines that did explicitly what needed to be done in order to design those warheads. And it's not just the bomb itself, but you have to design the carrier and the tin cans and stuff like that. And I remember roughly that Jack was given a reward on behalf of everybody in the Polaris project. That was a long time ago.

Miya: Yeah, but it also used a term which was different from what's basically known to historians like BADASH <ph?>, in terms of gun method and implosion. The term used was "linear implosion" which can mean a number of things. Can you say anything about that?

Michael: Hm uhm.

Miya: Okay, next. Tell me a little bit about working with organizations. I want to do the two major computing societies, but I also want to do, not just at the national level, but also the conference that we helped put together, and locally SIGBIG <ph?>.

Michael: <laughs>

Miya: You laughed! With Mary Fowler and me. And which you attempted to foist on me. But tell me what it's like in terms of working with the ACM.

Michael: Well, I can't really tell you much, Gene. It was a perceived need to have a sounding board with that kind of pressure voice on the thing, you know. Large machine, supercomputing, SIGBIG, if you will.

Miya: In contrast to SIG-Small.

Michael: Yeah. Or SIG-Ordinary. <laughs> And But there wasn't anybody who was able to spend the time necessary to pull all that stuff together. So that it was a functioning thing. It's not something you can have a dinner meeting once a month or something like that. It takes more than that to get those things moving. We didn't have anybody who could do that, as I recall.

Miya: Well, it wasn't that you not only didn't have that, but...

Michael: Who?

Miya: It's not that-- it wasn't as simple as that, as I recall. You know, Mary didn't have the time, and I had a big "rising sun" painted on my back, and a lot of people didn't like that perception.

Michael: Well, that's what I'm saying. People didn't have the time. They couldn't spend the time to handle the organizational questions for SIGBIG. SIGBIG had to be more than just a dinner meeting once a month or something. And it's the same is true. In general, well, what we found at the lab was that they were willing to "tolerate" is the right word I think. They were willing to tolerate staff members belonging to various pressure group things outside. Yes, you can belong to the American Physical Society. Yes, you can belong to the Acoustics Society, or the Optics Society of America, and things like that, in addition. And they'd tolerate that. And they'd tolerate you getting to go the meetings every so often. And they'd even tolerate you-- they used to tolerate you becoming officers in those various agencies. But even while I was still at the lab, and this is prior to 1993, say, the bean counters were sort of being critical about how people would spend their time on these other kinds of things. And it didn't matter if it was an ACM, or IEEE, or the Computer Society, or SIGGRAPH or any of that. No matter what you wanted to do, they objected to you spending time on that stuff, where you should be spending time working on "The Bomb," or something equivalent to it.

Miya: You want to add anything-- it just occurred to me-- to either, since you did mention SIGGRAPH, or the Cray User Group, CUG, independent of the ACM or the IEEE?

Michael: Well, people were allowed to participate with the Cray User Group because we made use of Crays and we actually got a lot of useful stuff out of those meetings. And also you got a lot of programs that helped.

Miya: You mean, internally developed programs, or programs from other places?

Michael: Well...

Miya: The reason why I ask this is because this is why Al [Kossow, CHM Software Curator] came in here a little while ago. They want to find Cray-1 programs and they can't find any, except one that I gave them.

Michael: Well, they don't know where to look, okay?

Miya: Where should they be looking?

Michael: Well, I would nominate Harry Nelson, number one. In fact, you could probably get the entire stack loop from that software from him or Frank McMann, or Lance Sloan. They wrote those programs for those machines. The principals are probably still around.

Miya: No, the problem is most of those programs have been modified for subsequent architectures. What they really want is something for the Cray-1.

Michael: Yeah, I understand.

Miya: Yeah.

Michael: I'm suggesting that...

Miya: I'll point them...

Michael: Send them to Harry Nelson.

Miya: To Harry, Frank and Lance.

Michael: There's a guy by the name of-- you know, that damn-- well, I shouldn't say it that way. Don did not do what I thought he was going to do about this Cray retrospective. I found what I thought were two good speakers. Nelson and Bill Bennett. Both those guys, you know, they were not neat and tidy programmers, they're nut-and-bolters. They write in machine language to get things done. You know, Bill is a very influential guy. First in our labs and then down at MFE. Then he went south to the San Diego supercomputer lab. And you know he helped set up the timesharing system there and so forth.

And a lot of that stuff is on Cray-1s. And I think that they're all retired now. And I believe you just need to ask.

Miya: Well, we'll get to these people a little more significantly here in a moment.

Michael: Okay.

Miya: So let's see. Why don't we just-- let me just check topics. You want to say anything about SIGGRAPH, since we were both in SIGGRAPH?

Michael: Yeah. When we first started SIGGRAPH many years ago, I think the last interesting meeting I was at was at Bowling Green.

Miya: Bowling Green in...

Michael: It was a long time ago. You know, now SIGGRAPH is too big, so you can't get around to talk to the people you want to talk to. And you can't even get into the technical sessions unless you buy special tickets for the thing, and so forth. It's just gotten way too big. It's worse than COMDEX and things like that, where the only thrust is to make money!

Miya: Okay. Anything else about these professional organizations?

Michael: We, I think helped SIGGRAPH a little bit also for the first supercomputer meeting. We sponsored Maxine Brown [?] giving a fairly extensive keynote talk about the role of graphics in supercomputers. Which he showed an enormous number of very good examples. And even to this day, you know, that was 1988, okay? And to this day, the graphics that you find that go along with the supercomputers is pretty elaborate stuff. That's about all I could say of that. It's good stuff.

Miya: Did your kids follow in your footsteps in computing?

Michael: Who?

Miya: Your kids! Follow in your footsteps in computing?

Michael: No.

Miya: Like your sons and your daughters.

Michael: Well, I think that's not quite true.

Miya: Go ahead.

Michael: Karen is pretty good at it. And the baby, Sarah, had a different attitude when I-- she had real mathematical talent. When I braced her with that that she had to become a programmer and stuff like that, she said, "Listen, when I need that kind of stuff done, I can easily find somebody to do it. That's not a big deal." <laughs> So, that was great! <laughs>

Miya: So you are happy, actually that some family members sort of followed in your footsteps?

Michael: I would've liked it if they'd gone into mathematics. You know, it isn't exactly true that-- but in the very beginning the people who were critical would say, "Studying a computer is like studying a slide rule." And that's wrong! But it gives you the idea of what the attitude was like.

Miya: Still is.

Michael: Well, for a large number of people, yes. But it also-- just look at all the kinds of stuff that you can get done with a computer now. Most of it is good ideas that are not fully implemented correctly. But they're there in potential, vestigially.

Miya: A couple of years ago, you told me about, as we were driving through Nevada, a tape drive that was four miles long.

Michael: Oh, yeah. I think it was more like ten, but all right.

Miya: Ten miles long!

Michael: I don't remember exactly. It was the first attempt to produce at a 100 megacycle recording frequency. And the tape itself was on a reel that was about, say, what like four foot in diameter. Something like that. And it was air-driven, and you put the tape on there, and then you bring it up to speed, and in order to do that, some two miles have gone by up onto the takeoff spool. Then you got about four miles that you can record in. And then you got another two miles on the end to slow down, to come to zero speed.

Miya: Or something like that.

Michael: Yeah. And that was at Lincoln Lab. And it was built for the Air Force.

Miya: Out in the Nevada test site?

Michael: No. Is it Lincoln Lab?

Miya: It was actually located at Lincoln Lab.

Michael: Yeah. I saw it running there.

Miya: So should the Museum go attempt to look for one of these four-foot diameter spools?

Michael: I have no idea where that thing went.

Miya: Or who made it or anything like that?

Michael: We have it here?

Miya: No, no. Or who made that?

Michael: I don't know who made it.

Miya: You simply saw it. All right. I have a bunch of other small quick things. Let me go down our quick list of people.

Michael: Go.

Miya: The first person you're going to say stuff about. Sid Fernbach.

Michael: Okay. I had a lot of trouble with Sid. He wanted to help me get fired and things like that. But in retrospect, I don't think he was wrong. I think I was a...

Miya: Loose canon.

Michael: Undisciplined enough that it upset him. He didn't like it. But on the other side of the thing, it's clear. He was brought over from a post-doc at Stanford, in physics. Edward teller brought him over and said, "You're going to run the Univac." So Sid went off to Philadelphia and learned about Univac and so forth and so on. And so he was our first computer center director or computation director. And at that time, computing was a subset of theoretical physics. And even to this day, I would say he was the best director that they had. And I think I could then spend an hour listing stuff that he was terrible about. And still he was the best.

Miya: He had a lot of visions, which before most people did, about the contributions that were being made with computers for computers by computers. I mean, you organized this *Advances in Computation*, a publication which went year after year after year, and he organized many of these what you call, special interest groups that we were in some cases paid for by the manufacturer. I can't remember all of them, you know, but one of the ones he liked a lot was to get all the laboratories within the Department of Energy, and help them collect together a stable of mathematical routines, that anybody could use. So they had to be movable, and they had to be Fortran-able or this/that about, and _____ and so forth. He did that. And I think he had the vision.

Miya: And he justly has the award named after him.

Michael: Well, he had nothing to do with that.

Miya: The IEEE award. That's why they give the Fernbach Award, right?

Michael: Yes, that was the thing that Al Brenner did. I helped him with that. We originally had slightly different ideas about the Fernbach award, but it looks like it's worked out okay.

Miya: Any last thoughts about Sid, or should we move to the next person?

Michael: Go to the next person.

Miya: Okay, this is the most important person here. Seymour Cray.

Michael: I think we've said so much about him already. You know, he was a genius. Some people said he was a packaging genius. Whatever it was he had-- he was really good at clean thinking. And the stuff he produced was beautiful stuff. And I would say that among most people in this country, they do not appreciate what they lost when Seymour was killed. They do not appreciate the brilliance of his designs. You know, you've heard some people say he's the "father" of supercomputing and stuff like that. Well, I'm not sure he wants to be the father of that. I mean, we always had it in mind that the machine we bought for the lab's use was the fastest available for the machinery, for the problems we had to run. A different set of problems might indeed have produced a different result as far as which machine is fastest of best. But given that, all you can ask for is that you keep true to your mission and have it happen.

Miya: Okay?

Michael: Yeah!

Miya: Steve Chen.

Michael: No remarks. I didn't know Steve that well.

Miya: Cray Research as a whole?

Michael: Well, again, I think the greatest virtue was that it was a clean, and unencumbered with a lot of administrative stuff that has no contribution to make at all, and in fact, I heard TJ Watson, Jr. say, you know, "How come that guy [Seymour Cray] can produce a 6600, had 32 people to do it. I have this thing that TJ Watson research labs at Yorktown with thousands of people, and we're not producing anything like that!"

Miya: Where'd you hear him say that?

Michael: At the lab!

Miya: At Livermore?

Michael: Yeah!

Miya: Okay. He came to Livermore.

Michael: Yeah. He wanted to see for himself what the 6600 was.

Miya: So you guys showed up with a 6600.

Michael: I didn't, but Sid did, yeah.

Miya: Were you there when he saw it?

Michael: I was in the group that heard him say this, yeah. I don't remember the exact venue, but that's roughly what he said.

Miya: Did he say anything else besides what's documented in the letter that he wrote?

Michael: No. I don't remember. I only remember this, you know. It sort of tickled me. He says, you know, when they asked him, he asked, "How many people worked for Seymour?" And they say, "32--that counts the janitors." <laughs> This was when he first moved to Chippewa.

Miya: Well, I actually found it wasn't merely Seymour and Watson who said that. I found out Leo Szilard said something very similar. Said, "You have to be able to tell the janitor."

Michael: Leo Szillard would say anything about Seymour. I don't think so.

Miya: No, no. He said, "If you have a new scientific idea, one of the things you had to do was be able to explain it to the janitor." <laughs>

Michael: Yeah. Oh. <laughs> Everybody, I mean, he had a rough, tough and smart group of people, including 32 people, including the janitors, and he had people who were devoted to the idea, and was not just a salaried post. You know, they worked for hours of-- and get paid. The women of Chippewa Falls took on the job of wiring. And it was a labor of love in a sense.

Miya: Yeah, actually, I've kind of encountered that. The women that worked for Cray actually sort of have a special sense of that level, too. So that's something actually that should probably be studied. Okay, say some more about Bill Simbrat <ph?>. I've only met him I think the one time you introduced me to him.

Michael: There's not much to say about Bill. You know, he was a replacement for Chuck Breckinridge in a sense.

Miya: Well, Chuck's next.

Michael: Well, I guess lump them together. Both of these persons are smaller stature persons, okay?

Miya: Right.

Michael: And I have heard it said that was deliberate choice to have people who were short dealing with Sid, who was short. Other than that, they were-- I dealt more with Bill than I did with Chuck. See Control Data bought season tickets to the opera and to the concerts, stuff like that. And nobody that wanted....

END OF TAPE 6.2

START OF TAPE 6.3

<Audio begins abruptly>

Michael: And one of them goes over to the console and says to the machine; is there a god? Is this machine going to answer everything now?

Miya: This computer that was built by scientific men.

Michael: And the computer went blank.

Miya: The computer went blank to the cleaning woman.

Michael: It frightened the woman. She left. But in the morning, when they came to work, the regular staff, they found the machine floor covered with paper, and it was printing out over and over again "There is now. There is now. There is now."

Miya: I just hope the context of that can be captured, your joke. Okay. Anyways. So much for computer humor.

Michael: There's one where the minister asks the computer, no, asks the computer well, is there a god? And tell me about god. And the computer just doesn't answer. It doesn't say anything. He keeps probing it different ways, asking the question. Finally it says "She's black."

Miya: Okay. Alright. We actually captured two George Michael jokes. Partial and we have full.

Michael: And there's the one about…

Miya: Oh, a third one. Go ahead.

Michael: You know, this admiral comes in and this machine is going to answer any questions and stuff like that, and they tell the admiral yeah, go ahead, ask anything you want. So he says well, what's my name? And the machine tells him. He says well, that's pretty good. He says where's my father? He says your father is buried in a churchyard in Chester, England. Ho, ho, ho, ho, the admiral exclaims. That just isn't true, you know? My fathers alive, yet. So the staff were very upset by this and they went and checked around and so forth and they learned that, it depends on how you want to tell this, you know, well, yes, your father is buried in a churchyard in Chester, okay, and your mother's husband is on a boat off the coast in Labrador, you bastard.

Miya: Okay. Alright. Let's get back to the history a bit. Okay. You were talking about Control Data as a corporation. And they apparently got you guys season tickets to the opera. Is this correct?

Michael: No, no, no. They bought a season ticket…

Miya: They is Control Data, though?

Michael: Control Data, you know, every big company has a marketing budget, okay? And how they use that marketing budget is probably their business. Control Data chose to buy a couple of season tickets. Okay? Bill Simbrat [ph?] had control of them. And he couldn't find anybody to go with him. So several times, he and I went to an opera or to a concert. We didn't go to the regular big fancy concerts. We went to those rehearsal concerts, which are much better. You know, people are relaxed and they can show you even though they aren't even trying, how to extract some meaning from music that otherwise isn't there. And well, you know, he was a fairly scrupulous in attending to the needs of the lab and stuff like that. I honestly don't remember whether there was much friendliness spent or wasted between Sid and him. Probably was alright. See, the president, Bob Price was also a lab employee back in 1953. And he would come out to the lab now and then and, you know, I remember seeing him several times.

Miya: And Bob is the third?

Michael: Price who was here [at the Computer History Museum] who gave a talk [June 12, 2006 at the Computer History Museum, "An Evening with Robert Price: The Control Data Story in Conversation with Mel Stuckey".].

Miya: Yeah. I wasn't there. But the thing is now he's third down from what we're at right now. I'll elevate him up if you want.

Michael: Well, you don't have to elevate him. I'm just talking about him. He has a degree in mathematics and maybe it's because I knew him, you know, that I didn't see any of those super attractive for being a president or something like that. But apparently Bill Norris thought the world of him. And Bob is a great person. And so he got to be the second and final president of CDC. CDC, a lot like DEC, you know, it came up with a good product. And then they hung onto it too long. And just lost all of its sales and so forth.

Miya: So you want to say any last things about Simbrat [ph?], Breckenridge [ph?] or Bob Price?

Michael: No. They're all friends, okay? And I much admire them.

Miya: Bill Norris.

Michael: Who?

Miya: Bill Norris whom you just mentioned [Founder and president of the Control Data Corporation (CDC)].

Michael: I didn't know him. I met him but, you know, that doesn't make you know him.

Miya: Bob Thornton?

Michael: On the other hand, you know, I knew all of the guys that were the company officers at DEC. Great bunch of people.

Miya: We'll get to the DEC guys here in a moment.

Michael: I don't think we need that.

Miya: Okay.

Michael: They were great people, you know?

Miya: No. Okay. Bob Thornton; do you want to say anything about him?

Michael: Who?

Miya: Bob Thornton [ph?]. Was a <inaudible>.

Michael: Oh, Seymour's assistant. Well, I think that, you know, that he was an assistant when they made the 6600. And in some sense, he got an enhanced reputation because the 6600 was a very good machine. So he was given the job of making the STAR-5, and it was beyond him. In some respects, he didn't do the job. But when he left CDC, he went directly into Network Systems Corporation and they produced the first high bandwidth…

Miya: HYPERchannel.

Michael: …channels that you could hook onto a machine. 25 million bytes a second.

Miya: They just removed the HYPERchannel newsgroup from YouthNet.

Michael: From where?

Miya: The people who run YouthNet just removed the HYPERchannel news group. It's gone. Okay. So how's Bill Thornton , right? Or Will Thornton? What was Thornton's first name?

Michael: Jim.

Miya: Jim Thornton.

Michael: Yeah. He and Seymour authored the book on how to design a computer.

Miya: Yeah. The 6600 design.

Michael: Yeah. And then he went on to do the STAR. And it didn't come out very well.

Miya: And then HYPERchannel.

Michael: Then when he went to a new company called HYPERchannel or Network Systems Corporation, that was done well, and you know, they sold a lot of those things to people who had high-speed machines.

Miya: It was barely appreciated.

Michael: Well, it was a good machine.

Miya: Yeah. It was a good thing at the time.

Michael: You know, they give Metcalfe a large amount of credit for Ethernet. This thing was better than Ethernet.

Miya: It was faster, but Ethernet was more reliable.

Michael: More reliable?

Miya: Hey, you could take a fire axe to it, and it didn't get to Ethernets not a non-functioning network.

Michael: Okay.

Miya: Okay. Same thing with token rings too. Okay. What about Neil Lincoln? Our good friend Neil?

Michael: Dear Neil. Well, to me, he came-- he was the spokesman for the STAR. And he's a fairly flamboyant kind of a guy. And fun to deal with. And you could, you know, insult the STAR in front of him and it didn't bother him at all. And I think that perhaps the best anecdote about him is at the first Salishan meeting, we had designed it so that Wednesday afternoon was free. And during that time, Ernie Plechaty was prevailed upon to give a talk on Monte Carlo techniques.

Miya: How do you spell Plechaty?

Michael: P-L-E-

Miya: P-L-E…

Michael: …-C-H-A…

Miya: C-H-A…

Michael: C-H-A-T-Y.

Miya: C-H-A…

Michael: P-L-E-C-H-A-T-Y.

Miya: Okay.

Michael: Plechaty. Okay.

Miya: Okay. So Monte Carlo.

Michael: Yeah. And so Neil sat in on the lecture. And at the end of the lecture, he stood up and he said wow, if I had known that before we started to build the STAR, it would have come out differently. Okay. That's I think a nice anecdote about Neil and…

Miya: You've said it before, actually. Which is okay.

Michael: Oh, he told the story about how the new company got its name. ETA <inaudible>.

Miya: Go ahead. You want to say it?

Michael: His son helped produce that thing. Somehow ETA is the beginning of a sequence of frequent--shown in the frequency of character table, ETAOIN.

Miya: <Inaudible>

Michael: Yeah. And so he said why don't you just take that part of it.

Miya: Yeah. And that's on Wikipedia just so you know.

Michael: It's what, now?

Miya: It's on Wikipedia.

Michael: Okay. Good.

Miya: Okay. Let's jump a manufacturer here for a moment. Peter Capek?

Michael: Great person. But you know, well, he and a bunch of other people who I'm not going to name at Yorktown are always worth visiting when you go there. And that's what we did.

Miya: Okay. Well, he's retired now.

Michael: He's retired, yeah.

Miya: And I should probably email him to see if he's coming out here in November.

Michael: He should be here in November, huh?

Miya: I don't know. I don't know if he has the money. Gene Amdahl.

Michael: Yes. Gene Amdahl. Gene Amdahl. Okay. When there was this new thing, ACS, advanced computer system being worked on at IBM, Jack Bertram was in charge of it and so forth. And Norman had gone over there and was working on it and stuff like that. Gene came in and he altered the direction of that program enough so that it lost all of its interest. And he was asking the question; which is it better; to have a machine that's very fast or one that's reliable? That's oversimplifying what his question, but that altered the design of the project Y machine so that it was now, you know, going to be a fast engine for him in 360 code, 370 code. And I think that he thought in a sense that he was headed off in a direction he didn't like, so he formed his own company. And well, the last time we talked at any great length was when we were at Gordon Bell's birthday party. I chatted with Gene for a long time, and he's definitely mellowed. And I think it's impressive that the Amdahl Computer Corporation is kind of a mini-IBM in a way. I'm surprised. I wouldn't have chosen to do that. He's a great person.

Miya: He's playing a lot of golf. That's why he's probably mellowed. No, that's what we talked to him about. His wife was telling me that he was playing a lot of golf <inaudible>. It's not far down the street from Decobar [ph?] there. I have to laugh. <inaudible> Okay. I won't get any other IBM people here in a moment. Okay. What about the DEC Company folk as you've just called them?

Michael: Well, when DEC started out, they were exciting, okay? Again, they didn't have a heavy management structure and in the time that they started, the American whatever company [American Research and Development Corporation] it was that financed them, you know, it was their first venture capitalists. I met that guy on an airplane; Jesse Cousins. And he said how they, this company, American something or other, takes a position on the DEC board, but they didn't want to be up in front. And so DEC could run things the way they wanted to. In the beginning, when you went to the mill in Maynard, you know, it was always routine. You went and visited Ken Olsen and then we would go see Harlan Anderson and Stan Olsen and Ben Gurley. I don't remember others. And I got my first lessons about how to draw DEC diagrams from Ben Giurley okay?

Miya: Anything else you want to say about them?

Michael: No. I think that it was a kind of a compliment to say that Ken Olsen wouldn't fire anybody. He felt we had a responsibility to those people. That's much more in the spirit of how Europeans and Japanese look upon their workforces.

Miya: And also Hewlett Packard. And also HP.

Michael: Really? Well that's good.

Miya: But times have changed. I suspect Seymour was probably that way as well, too?

Michael: Yeah.

Miya: Was Amdahl that way in the beginning? Was Amdahl an exciting person?

Michael: By himself he was, yeah. He spun off ideas at a furious rate, but you know, some of the clues were that he forced the 360 organization on a project Y, and it's probably not a good idea.

Miya: Was project Y for Yorktown?

Michael: No. That was just ACS.

Miya: Right. It was just a codename for ACS.

Michael: Yeah. And it was headed in the wrong direction. In a sense, I happened to be over there one day. I walked in Norman's office, and there's these two ten foot high stacks of paper. These are manuals for that machine by Systems Development Corporation.

Miya: By SDC. Okay.

Michael: No. That's terrible. That's not in the capacity of man.

Miya: Right. You want to say anything else about the DEC Company folks besides being exciting in this chain of people?

Michael: Well, no. I think that they just did things the right way. You know, all of them were MIT-trained and all had done respectable things initially.

Miya: Do you think it was MIT that made them that way?

Michael: Huh?

Miya: Do you think it was MIT that made them that way?

Michael: Yeah. MIT is an incredibly good school.

Miya: You want to say anything about MIT?

Michael: It's an incredibly good school.

Miya: Besides that.

Michael: No. I think that you know, mostly two things. There's a negative input on this thing, and they tend to hire their own students more than they should, possibly. It's not considered very good from the academic point of view. All the people who ever ran the AI lab, for instance, all came from MIT.

Miya: Right. That's a common criticism of other technical.

Michael: Art Winston [ph?]. Marvin Minsky.

Miya: Well, we'll talk about Marvin in a moment. What's the second thing you said-- wanted to say, besides hiring their own students?

Michael: They have always managed to maintain incredibly good standards, but I think that's because the graduate students are given free rein, you know? Yes, the guy-- Pat Winston [ph?] went as an undergraduate, then as a graduate and studied there. But now when he's running the AI, he's running a bunch of students who are given free rein to do things the way they want to. And they were, you know, it's just-- they're really great people.

Miya: Okay. Well it turns out the next two people I'm going to ask you about went to MIT. And that was Ivan and Burt Sutherland. Yes?

Michael: Well, I think they're really good people, okay? I got to know Ivan while he was a graduate student at MIT. And I thought that his movie on Sketchpad was very good. And we had something similar which has long since disappeared. But it was, you know, Ivan did a much, much better job. And again, he's a person who could see much further into the future than the average person. And so is Burt; the same way. I you know, I dealt with Ivan when he was head of that OPT office in <inaudible>.

Miya: Oh, information processing technology.

Michael: Whatever it was. IPTO [Information Processing Techniques Office]. And the only piece of sadness in a sense was that we showed him some of the work that Norman and I were doing. And he said he'd supply a million dollars, okay, if we would work on that project, okay? And there was this guy, Arthur Hudgins [ph?], an administrator type physicist, and he said a million isn't worth our time. And he wouldn't let us take it.

Miya: Was he at the lab or-- this Arthur?

Michael: He was at the lab. Now, maybe it turns out that it was a wise thing that he didn't allow us to take it; I don't know. But I always felt that, you know, the way we were going to do things was dramatically better partly on the business of whether you're dealing with a hot bay or a small computer, I think it was much better than dealing with these small computers. But we didn't get to do anything.

Another guy I met on occasion was Phil Peterson. And when I met him, he was at Lincoln Lab. He's the guy who produced, I don't know if you ever remember seeing it, the CalComp plot of the Mona Lisa.

Miya: Yes. There's one downstairs.

Michael: He scanned a 35 millimeter slide with an eyeball that was attached to the JX2. And then the densities that he measured were mapped over into this number system where each number was, you know, had a relationship with the density it was supposed to represent. And so you put the pen down on one corner of the drawing. You don't read it again until you're at the other opposite corner. And the other end of the drawing. And Phil and John Frankovich [ph?] at Lincoln Lab, maybe others that I don't remember now, but they produced an eyeball which we produced at Livermore, right? And it was very good. I think we had a far more flexible thing, but it was plagued with a lot of noise. I could fix that now.

Miya: Back then you weren't able to?

Michael: Well, I wasn't so much interested as it wasn't permitted.

Miya: Okay. Let's talk about some of our-- well, I don't know about the first guy-- friends at the lab. Alright. You ready? Paul Woodward [ph?]. What's so funny. What are you laughing about?

Michael: Well I don't know. Why are you dredging up all these names?

Miya: This is a list of names that you and I came up with that you and I agreed that I should ask you about.

Michael: Paul Woodward came to the lab from I don't remember where or when. But he was a, you know, a numerical physicist, computational physicist, whatever you want. And he's interested in astrophysics. And he produced some running programs that were very, very influential in the field of astrophysics. And you know, he just did this because he was a fertile kind of person. He hauled me down to Socorro, New Mexico, when we went to the national very large array [VLA] place down there. And that was a very, very, very impressive trip. We were down there to tell those people stuff that we knew about storage. And the people there had a storage problem, which was, is still, you know, way, way out on the fringes. Not many people can handle that much-- that number of bits. But now the lab is deeply involved in that sort of thing in the person of Don Dosa [ph?].

Miya: Okay. George. You said that Paul Woodward [ph?] went to Minnesota, right? Okay. And you were going to say that he basically studied shock propagation and stellar atmospheres. And he compressed time to get some sort of a result <inaudible>.

Michael: Yeah. I can't remember the details now. He came up with a technique for doing the calculations more rapidly than otherwise. And that meant they could study more of these phenomenon. And they do.

Miya: So they could study it more.

Michael: You know, he does that.

Miya: Yeah. Charlie Weatherall [ph?].

Michael: Ah, dear Charles. He first came into the lab I think as a summer hire. And I think of him as one of the really smart computer types. You know, he was important to us. And he made the research group elegant. And between him and Jim McGraw [ph?] they were able to crowd Jack Emmis [ph?] and his graduate student Gil Hildigrand [ph?], no Bill Ackerman [ph?].

Miya: Bill Ackerman.

Michael: Bill Ackerman into a specific thing that, you know, they-- Ackerman was doing.

Miya: What dataflow?

Michael: They produced a dataflow language valve. And I think that without Charles and Jim it wouldn't have happened.

Miya: Yeah? George? George? You're talking about Jack Dennis, Bill Ackerman with Charlie and Jim McGraw getting valve.

Michael: Yeah. They were producing the dataflow language.

Miya: Right. Right. You want to say anything else to that?

Michael: At that point, you know, we were sort of committed to trying to use dataflow. Out of that stuff came finally SISAL, and SISAL in a sense was successful everywhere but the lab. I think the guys did the right things in what they did, which was to build the SISAL translator so that it was compatible with Fortran I/O. And they did that. Now, I think that we sent Jim McGraw [ph?] and Steve Skedsielewski to England for the summer so they could work with Gerd [ph?] and others on that point.

Miya: You tired, George?

Michael: Yeah. I'm getting sleepy.

Miya: I know you are. I can tell you we just have an hour and 20 minutes. We need to finish, okay? Hang in there an hour and 20 minutes. Okay. We're still talking about Charlie Wetherall one of the really smart computer guys. And he did this book that you really like a lot, which I lucked out to get a copy of; _____ Super Programmers. Tell me about it.

Michael: Well, I had the inspiration of having learned about Mary Shaw at Carnegie _____. She ran a thing called the immigration course where beginning students in computer science would first have to go to her class and learn how to program. And learn that they were, you know, programming in various languages and stuff like that. And so she had a series of problems that came along that way. I was using

those to start our beginning students at DAS that way. And Charlie came along, and between us, we produced a bunch of even more interesting programs to train a new programmer. And I think it's safe to say that if you could get through Charlie's programs, you were pretty good.

Miya: It's a pretty tough book. Looking at it.

Michaels: It's not tough. The one problem that I remember is a Fortran program that reproduces itself.

Miya: Right. I remember that one.

Michaels: That's a toughie.

Miya: Yeah. See, you agree with me.

Michael: Well, for one problem. Anyway, that was Charlie's magnum opus. I'm sorry it didn't make a bigger splash on the book scene, because I think it's clearly one of the best books that's been produced. It talks about programming for people, not learning how to write Fortran. That's important.

Miya: I have to say I agree with you.

Michael: Go ahead.

Miya: One of the things in my library that I appreciate sitting right next to your concrete mathematics, to which my friend Darrel Long says, what an intimidating book.

Michael: Yeah. Anything that Donald does is intimidating.

Miya: Oh, I don't know. I actually appreciate <inaudible>.

Michael: I think he's super. I always remember the criticisms of Hamming [ph?], you know, call him an _____ man.

Miya: What? Hamming [ph?] called Knuth an _____ man?

Michael: Yeah. He thought that Donald shouldn't be wasting his time on setting typography.

Miya: Well, I told him that, actually. Don just smiles.

Michael: Well, see then you could join Hamming [ph?].

Miya: Right. I know. Well, I liked him.

Michael: I think Hamming [ph?] was one of the truly unsung greats of this last century.

Miya: Well, you learn something every day. Well, Booth [ph?] and Hamming [ph?] are further down the list. Okay. So you want to say any last words about Charlie besides _____?

Michael: No. I think that-- no. I don't want to say anything else.

Miya: Okay. You mentioned Jim McGraw [ph?]. Any last things you want to say about Jim McGraw?

Michael: No. He entered my group as a very bright and very aggressive and very impolitic kind of person. And I remember he used to come into my office a lot and we would have what was called a curve filing session, okay? We're filing down these curves so he wasn't so upsetting to all the people. Because he sort of called it like he saw it. And he did lots of stuff to upset an awful lot of people. Rightly or wrongly, okay? Now, he's entered the realm of management now, so I don't know what's happening.

Miya: I see him every now and again. Any last things you want to say since you've mentioned them with regard to Charlie about Jack Dennis and Bill Ackerman?

Michael: Well, I think that in a sense, Dennis is a perfect example of a beautifully trained and brilliant academic who has too much experience with running very large programs and trying to produce lots of answers and stuff like that.

Miya: Lacks experience or has experience.

Michael: Lacks it. You know, he produced a lot of interesting stuff; this single-assignment language. And I for one learned an enormous amount from him okay? I didn't like the fact that he quit MIT in order to go try to be in business. And everybody has got the right to do that, I suppose. I didn't like it.

Miya: That's okay. I shared an office with him for a year. Bill Ackerman.

Michael: The only thing I can remember about him is he has a great laugh. When I was, you know, I like to play with words a lot. I was telling him that I had to come in here and watch Ackerman's function. You know, there's this function.

Miya: Yes. I know Ackerman's function.

Michael: Then he broke him up.

Miya: Yeah. Whether other people who view this know what Ackerman's function is, that's okay.

Michael: Well, I'm always impressed with how, you know, he could do so much. He was just an incredibly good guy.

Miya: Okay. Steve. Tell me about Steve Skedzielewski.

Michael: What can I say about Steve? He's…

Miya: He's getting married again in January.

Michael: Is he? I hadn't heard that.

Miya: In Maui.

Michael: Connie.

Miya: I've never met Connie.

Michael: I have. A very nice person. He had a rough road to hoe.

Miya: Right. Steve had a rough road to hoe.

Michael: And I think of him as a quietly competent person. And many times I was stuck on something, I could ask him and he'd help me out. And as in many of the young people who are, you know, so talented, he had wanted everything in his life worked out so that click, click, click through life. So they were going to have this kind of a house and that kind of a child and all that sort of thing. None of that worked out, but he kind of weathered the storm I think beautifully. He's a good guy.

Miya: Oh yes, I too. But I will never forgive you for taking him away from me.

Michael: You don't have to forgive me. I'm not interested in that.

Miya: I know that.

Michael: But I didn't take him away from you. You never had any chance to begin with.

Miya: I know that he didn't have any chance, but I needed him at JPL. Okay. Any last things about Steve?

Michael: No.

Miya: John Gerd; you have two minutes.

Michael: John Gerd called me up when he was here in this country, and we found quickly that we had mutual interests. So that was the start of us going over to England. I went over there with Chris Hendrickson [ph?] and Tim Rudy [ph?]. And we toured various English universities including, well, John was then at Manchester as I recall. And the name of the game was the study of functional languages in general, but single-assignment languages in particular. And as these things go, you know, we went over there and they would invite us to give talks about what we were doing and so forth and so on. And we did

that and it was a lot of fun. But London; they had a funny notion of what a Yank is, okay? And so they were going to drink us under the table one night. We met at the Northumberland bed and breakfast place and went out to dinner. Well, we had numerous pints before we went out. And we had numerous pints while we were at dinner. And we came back and the British have this rule; if the guest wants to keep the place open, he can stay all night if he wants. So we stayed in the pub all night long. And we had more pints. And when you tally it all up, I gave up, everybody gave up except-- including John Gerd and his friend Ian Watson, except Ronan Sleek didn't give up. And neither did my bride, Heidi. And she drank without getting up to go to the restroom or anything like that, eight pints of beer, of ale. Big pints. The 20 ouncers. And she sort of put the English crew in awe there. They called her eight pint Heidi.

Miya: Ah, that's not something we learned about your wife.

Michael: Well, it's just great.

Miya: Okay. Are we at the end now?

Michael: John Gerd was…

END OF TAPE 6.3

START OF TAPE 6.4

Michael: Occurred so it was John and Ian both marvelous young men, we had the privilege of helping them get to be tenured faculty at Manchester and since then of course John's gone off to Oxford I guess it is where he met his most recent wife and he comes over her periodically and he stays with Chris Hendrickson when he comes over here. But then we always get together for a supper to talk and Rushby, John Rushby whom we met over in England is now at Stanford Research, he comes over for the thing too so it's a kind of a little get together, it's very social, there's no work involved.

Miya: Okay so John Rushby at SRI?

Michael: I think that's correct yeah. Anyway we watched the British put together a brilliant crew of people on this business of single assignment languages and functional languages and operating systems and stuff like that and just as quickly disassemble it because they said they were done with it now. It was stupid, telling them it was stupid didn't help, I mean they didn't change their way at all.

Miya: It was just the English way of doing things.

Michael: Well it's stupid, you know they were saying which is exactly what's wrong here, we're not going to fund academic research anymore you guys are now going to have to be funded by industrials who will fund you and will control how much you do and where you look. So, you know, you're no longer doing research where ideas lead you, you're now doing something which they think may have a sales potential, that's baloney.

Miya: Okay that completes John Rushby.

Michael: That completes John yeah.

Miya: I won't ask you about John Rushby.

Michael: Well nothing to say about him, you know.

Miya: Yeah its okay and I won't delve deeper into Ian Watson, Sharon Lee and Floyd Danielson?

Michael: Sharon Lee was my second secretary and I would not have had her if the first secretary had been so to say willing to work full time but she wasn't so we had Sharon Lee and I must say people were jealous of me because, you know, Sharon Lee as a young woman was quite attractive; her husband Floyd was a pharmacist out at the Veterans home south of Livermore and he's a very a taciturn type person as you know but she's flamboyant as hell and a very good-hearted person and responsible for typing all the initial interview tapes, transcribing on this.

Miya: Like these interviews maybe, she did the equivalent right?

Michael: She types the interviews that I conduct with these people okay and then she nails them down, but once she's through typing, now it's an electronic record and we don't have to go back to tape ever again.

Miya: She was your attractive, flamboyant secretary?

Michael: Yeah she was now going to join a dance crew troupe up in Banten, Oregon where she lived.

Miya: Right and who I happened to run into when I was flying to Hawaii. Okay next, Frank McMahon?

Michael: Norman told me a rule once that anybody who played the game Go very well was a good potential programmer so another rule that I had which I got from Ed Fredkin I think is anybody from MIT who had failed because of being attracted by the Model Railroad Club was a good programmer. Frank McMahon fits into that category, you know, he went to MIT but he didn't get through, I think he busted out because he spent so much time with Model Railroad Club and so I was trying to get him to work on my stuff okay and I was outvoted so to say or out drafted by Hans Bruijnes who took Frank and Frank became a very good compiler writer okay and that's what he is yet. He's retired now and he doesn't have much humor in him but he's a very, very good guy.

Miya: So he's not a great humor but he's a great person to talk to I think.

Michael: Yes.

Miya: So Frank did not get a college degree but you guys hired him still?

Michael: He was hired as a computer operator and so I was able to say I wanted him, I had several people who came in as computer operators and I got them to be programmers or better, you know, yeah, he was one of them.

Miya: Yeah I appreciate him for the Livermore Fortran Kernels, (LFK).

Michael: Well he's now, you know, partly responsible for this business of performance measurement and he handles the stack group library, things like that, well he did that when he was still working.

Miya: Lance Sloan?

Michael: Another person you might say is a Frank Walton/Frank McMahon-type person, okay? Lance Sloan, as I'm told, had the distinction of having the highest graduate record exam scores at the lab, they went out to the Department of Applied Science and took their courses out there, Lance was Hertz fellow, smarter then a whip but he didn't have what you might call the completion drive or something like that that he had to finish everything the way he wanted it and so forth. So I don't think he ever finished a degree program and I don't know whether he was considered a super programmer or not, he was a very smart cookie and it was good to deal with him in that respect, all the time.

Miya: Harry Nelson?

Michael: Yeah Harry is a good friend and what do I want to say; he's--- I think single handedly made the STAR whatever production capability it had came through him okay.

Miya: Useful.

Michael: Yeah he was useful and he's also a skillful games man, you know, mathematical games and he has an international reputation in that respect and, you know, he's got a fairly slow drawl type of delivery and it irritates a lot of people from what I could tell because they think he's running a trip on them or something like that but I don't think that's Harry. Harry is a highly ethical person, I'm a great admirer of him, you know.

Miya: I know Harry enough, I agree with you, that's a good characterization of Harry. Want to say any last things about Harry?

Michael: No.

Miya: Tommy Thompson?

Michael: A brilliant person, he's what you would call a computational physicist and that is a distinction between a designer who will take the capabilities of neutronics and nuclear radiation and blah, blah this and that and put together some kind of a device. He, Tommy is a user, a computational physicist.

Miya: A designer from a user thing?

Michael: Yeah and he's gifted with much deeper insight then most people have.

Miya: Do you mean most designers or just people in general?

Michael: Well I mean he's just he's a brilliant guy and I appreciate people like that, I like dealing with people like that.

Miya: You referred me to him when neither you nor Cliff could speak publically about the computational uses for weapon design and you were able to give me Tommy and Tommy actually did a very good job in front of a bunch of opponents, you know, he was just there merely to educate them as to the computational process and because he worked on defensive weaponry as opposed to deterrent weaponry, he had an appreciative audience, they appreciated the fact that he worked on anti-ballistic missile systems, anti-ballistic missile warheads specifically as opposed to counterforce or countervalue warheads.

Michael: What I find unhappy, not about him --- he has these insights okay but they don't listen to him, the managers don't listen to him in a sense that they implement or correct something that he points out to them okay.

Miya: Is he retired or his he still working there?

Michael: I don't think he's retired but I don't know how heavily he's working now; I haven't talked to him in over a year.

Miya: That's better than me; I haven't talked to him in 20 years. Computer requirements for missile defense is what I have here, but that's okay, you know, I don't know if you want to say anything about computer requirements for missile defense, nothing, okay. Chuck Leith?

Michael: He was one of the three people who most influenced my life at the lab; the other two is Norman and Dana Warren.

Miya: Dana is next.

Michael: Anyway Chuck, you know, is probably one of the smartest people at the lab and he got to the lab well in the early stages of the Manhattan Project he was assigned, even though he was in the military, he was assigned to work at Oakridge and he did that with Herb York, then he came back to the lab and the original things that he did at the lab while classified are fundamentally important in computational physics.

Miya: So he was part of the Manhattan Project?

Michael: Well in the sense that he worked down at Oakridge, yeah.

Miya: Well, was he part of this special engineering detachment thing?

Michael: In the sense that the Army attached, you know, gave him the job instead of putting a gun in his hand or things. I don't know of any formal arrangements, you may.

Miya: No I don't know--- I've never met Chuck, you merely mentioned him a lot.

Michael: He is brilliant.

Miya: Yeah I can tell, like my friends.

Michael: He left the lab just to go back--- he wouldn't accept the degree in physics, a PhD in physics, he went and got it in Mathematics from the University of California in Berkeley and then he came back to the lab. He went from the lab as one of the most knowledgeable persons who understands computers and he had an enormous amount of influence on the setting up of the NERSC, not NERSC but the NCAR setup at Boulder and the only problem is that they gave him administrative work to do there as well which he doesn't like so he left and came back. His burning interest is three-dimensional turbulence; he's the guy who in my opinion did the first successful implementation of the so-called Louis Richardson primitive's equations on a computer. The key here is you set the thing to running and there's a big noise spike that it starts because everything is wrong and that noise spike damps out and it continues to run forever, errors do not destroy it. That was one of the first requirements for having a model of the atmosphere running for predictive purposes, that's in addition, you know, to the classified work that he was doing.

Miya: Which you can't talk about?

Michael: The work on weather is not classified, though they required--- yeah I guess it's a way to put it, these administrators required that he not talk about his un-weapons work related stuff because that would give congress the wrong idea, it's just bull.

Miya: A political problem.

Michael: But in any case one thing he said to me which I've always remember, he said, "I believe in the element of futility of all human activity."

Miya: I like that idea, I will write that down.

Michael: In the early, early days he and Norman, Bill Lindley and I would go back to the lab at night and work and then go to have breakfast at a place called the Hungry Truck out on Highway 50. It's the pancakes out there were about 15 inches in diameter, god.

Miya: The Hungry Truck?

Michael: It's gone now.

Miya: I know I'm trying to think of where Highway 50 came in, Highway 50's like 60 miles away from the lab right?

Michael: You can if you want to try, I don't know, I don't know.

Miya: The Hungry Truck.

Michael: Yeah a guy by the name of Bill Busic ran it, he's a flaming socialist and he loved to come and sit with us and argue, you know, we would have maybe a more conservative view of things and he was pushing socialism.

Miya: Okay so that's Chuck Leith, the first of the three most influential people on your life right?

Michael: Yeah.

Miya: The next I have here and I don't understand why we wrote it this way, maybe you did or I did, Dana and Grace Warren followed by Grace Hopper?

Michael: Not me.

Miya: I'm not sure why.

Michael: Well because I would meet her there.

Miya: Oh, you would meet Grace Hopper at Dana Warren's place?

Michael: Let's put it this way, Dana was a philosophy major undergraduate and then his father was a missionary in Japan but then he went to Japan after graduation and then he returned to this country to study physics, I think it was done at Harvard or maybe it was Yale, I don't remember. In any case he met Grace Warren well Grace Adams was her name then okay, she was a member of the Adams family, Samuel and John Adams, all the famous revolutionary things and she was in the branch that was called the 'Western Adams,' okay? Grace Adams' roommate in college was Grace Hopper, they were always called the two Graces and when Grace Hopper came to look at the work that was being done on the Univac and stuff like, she would always stop to see her ex-roommate and we would go over there and read a play, okay. Everybody took a part in a play, a Shakespearean drama, whatever and everybody would read, you know, in the process I got to talk to her about COBOL which I thought was atrocious.

Miya: What did Grace Hopper say about COBOL?

Michael: She thought that the businessmen liked it that way. I said well I don't really think that what businessmen like has any bearing on anything.

Miya: And what did Grace Hopper say when you said that?

Michael: I don't remember.

Miya: What else about--- you say work things about Dana Adams, this is the second most influential person in your life mind you.

Michael: Yes, you know, he was a physicist okay, a fellow of the American Physicist's Society.

Miya: What he designed primaries and designed secondaries? [Stages of a nuclear weapon—Ed.]

Michael: Well I don't know how you'll take that but beyond the physics, what he showed me is that you could be a good physicist okay but you can have all kinds of other interests.

Miya: He was a Renaissance man as our friend Robert Gibney would say?

Michael: Oh well, he was one of the most knowledgeable Shakespearean scholars I ever met, okay?

Miya: Okay so he's the reason I have to blame when we quote Shakespeare.

Michael: Yeah well he could quote it by the hour and he could give you the interpretations too. He had a peculiarity which I always enjoyed quite a bit, he didn't like store bought mayonnaise, but Grace his wife would make home made mayonnaise okay and she'd put that in a separate container in this box, little tin box that he carried his lunch in. Also in there was a tea cup and a tea bag okay and every noon he would go to the fountain down the hall from our office and put hot water in the cup and he'd stand there and wait and argue with people and finally he was satisfied there was enough hot water in there long enough so now he'd throw that water away then make a--- put in a fresh load and put the tea in.

Miya: Right that's English.

Michael: So it's preheating the cup yeah. <Inaudible> actually imitated him, it broke him up. But anyway he was a Renaissance man, that's a good word for him yeah.

Miya: So he was multilingual as well too?

Michael: Well, yeah, I suppose I mean he knew Japanese. He liked Japanese pornography too.

Miya: Was he a computational physicist or just a general physicist?

Michael: Well I don't know if we could even make that distinction in the early days, he was a physicist and he and I actually tried to produce a two-dimensional version of the equations for hydrodynamics. Ours was not successful, there was an earlier one produced by Bryce DeWitt that was not successful and actually I think Bill Schultz and Roland Herbst, Mike May, maybe a few others actually produced the first successful two-dimensional model.

Miya: I think I might have put that down too. Okay, well how about our good friend, well if you have more to say about Dana and Grace Adams or Grace Hopper.

Michael: Well.

Miya: Norman and Ann Hardy?

Michael: Well Norman is also a genius, one of the first persons I met at the lab, who would come to the lab because we had computers, okay? Other people came to the lab because it's a place to have a job or they were interested in physics.

Miya: He came from where?

George Michael: He came from Berkeley and he was interested in equations that you could solve on a computer and those equations are the equations of mathematical physics which <inaudible> and I think he more than any other single person influenced the direction of computing at the laboratory to head down the trail of resource sharing and finally timesharing aims and so forth.

Miya: Do you want to say anything else about our good friend Norman and Ann?

Michael: Well I guess so, Norman really was other-worldly okay, he didn't really pay much attention to him, he bought a car from a mutual friend and eventually it got to the point where he was going to go off to IBM to learn about the Stretch and the Harvest, stuff like that and he decided to leave the car with me. So I found 232 Hershey wrappers in the front seat, I found I don't remember how many, maybe 10 pay checks in the back seat that he never cashed. I found books that were rotting in the trunk, "Continued Fractions" by [Hubert Stanley] Wall, these are big books and also I found a whole pile of shirts okay. When he finished wearing a shirt, he'd just throw it in the trunk.

Miya: Anything else you want to say about Norman and Ann?

Michael: Well when he went to England one of the persons he met was a young lady who had taken a test that IBM gave that showed she could be a good programmer. So he was teaching her how to program and it was Ann that he found and she came back here, the committed marriage and made a couple of babies, very lovely person.

Miya: Oh yeah she's very friendly, I like her.

Michael: I don't know why they got divorced but...

Miya: Well ,yeah that kind of happens. Anything else you want to say about the two of them?

Michael: About those two, no.

Miya: Let's go to some of our lesser friends, lesser personalities, not quite as impressionistic okay, you ready? Mile Bosch, Cliff Rhodes?

Michael: Why do I have to say anything about him?

Miya: You agreed to have; this is a list that we agreed on of our friends.

Michael: I was impressed with how smart Cliff is okay and I was also shall I say not impressed with his discretion or discretionary use of privileged information, he used it the wrong way in some cases at least wasn't friendly, but other than that, you know, I think that he had a incredible ego.

Miya: Cliff?

Michael: Yeah well, you know, I learned about how good Cliff Rhodes was from whom, from Cliff Rhodes. I'm not arguing that he wasn't good.

Miya: I know what you mean. Anything else?

Michael: No.

Miya: Dick Watson?

Michael: I think that anything that Dick wants to do.

Miya: He should be allowed to do it.

Michael: He should be allowed to do it, yeah but he's not an accommodating kind of person okay and not as bad as Cliff in any respect but he's in the same direction.

Miya: I think Dick is a historic person not only from his time at the lab but at SRI and the lesser known parts of the ARPANET.

Michael: In those particular roles, he functioned extremely well I think and he was critical of Crawley because Crawley is lackadaisical and wasn't in Dick's opinion, he was not going anywhere when they were trying to build this new operating <inaudible> okay. So he went to Sam and kept hounding Sam until Sam finally agreed and they removed Bob as head of that little project and put Dick in.

Miya: Anything else George, anything else about Dick?

Michael: No.

Miya: John Ranoletti?

Michael: Okay, what can you say, he's gifted though probably not as gifted as Norman or Harry or people like him <inaudible> them and in a sense, you know, he took the job of being head of the Computation Department for a variety of political reasons which I can't really enumerate but under his tutelage things went along alright okay but mostly things were subordinated to him getting his PhD and I'm not sure that that was good for everybody.

Miya: Yeah I heard that too; anything else you want to say?

Michael: No I liked John quite a bit.

Miya: I do too; he was <inaudible> members, Burton Smith who we'll see next Thursday.

Michael: Burton to me was the second best computer architect in this country if not the world okay, the first being Seymour and then Burton, Burton saw stuff in the HEP1 and HEP2 and this multi task thing.

Miya: MTA yeah.

Michael: MTA yeah that those are good things, I believe in every case they missed the window.

Miya: Right, I do too.

Michael: And the fact that he has left Cray and has gone off to Microsoft indicates that there's some really bad stuff going on at Cray. What do you think?

Miya: I think that we should joke with him and ask him how Bill Gates is doing.

Michael: Well, you know, I followed Burton's footsteps at NSA okay and they recognized he was made a fellow very early on, that's very rare <inaudible> and given privy to the deepest secrets that NSA had and in some cases you can maybe guess what those secrets are because you can look at the so called Horizon Computer which is MTA and see what he was coming across with. He quit NSA because they said they would not allow him to work on that machine, he wants to build it.

Miya: Anything else about Burton?

Michael: I think of him as a Renaissance man who's very down to earth about lots of stuff okay and to me I think it's just great to be able to talk to one person about a whole bunch of different thing and that's certainly true for Burton, he's good.

Miya: Want to include his wife in that or do you not know her nearly as well?

Michael: No I've only met his wife once.

Miya: Oh I've met her more than you, amazing.

Michael: Where?

Miya: She's a librarian at the University of Washington. Dave Cook?

Michael: Well he participated in the development of compilers and stuff like on behalf of the laboratory when we were going through this exercise with single assignment languages and things of this sort and we gave him a grant or a contract, I don't know what you'd call it to work on a compiler and I don't think he ever delivered it and it was, you know, whether or not it was part of the University of Illinois, I don't know or was it part of Cook and Associates, I don't remember. But what can you say he had this flair for this thing that he could do and he did it. Again I don't see him in the same capability range; I don't see him in the capability as Burton or something like that.

Miya: Dan Slotnik?

Michael: Interesting person, he introduced the idea of single assignment, single data path--- multiple data path, the SOLOMON, the SOLOMON was designed to solve a certain Air Force problem but when we found out about it, Norman went back there and told them about floating point arithmetic, this is Westinghouse in Baltimore and so they could see that you could build a SOLOMON to do floating point arithmetic and that's great. So we went ahead and wrote a specification to do that and it turned out that Westinghouse threw Slotnik out about that point and, you know, he surfaced, I guess, at the University of Illinois and he was pissed by that so even though the Westinghouse proposal was good, he wouldn't touch it. He took the Burroughs proposal to build the ILLIAC IV okay and I think he left a great mess on NASA's doorstep when he brought that thing here. I mean it was just--- even though they got nothing done, Pirtle and when _____ so forth, <inaudible> got more done on using that machine.

Miya: Steve Lundstrom?

Michael: Well Steve picked up the ball at Burroughs and tried to deliver the high performance computer that we didn't get. We got the STAR instead and I think he became a vice president at MCC but they didn't treat him well I think there, any case he left there. I don't think he's worked since.

Miya: Well he just does small consulting gigs, I see him every now and again, I should go by his house and say hi because I like him. Oh I know what I was going to ask you, you mentioned that Dan Slotnik received Norman at Westinghouse, most people don't think of Westinghouse as a computer company.

Michael: Well most people don't think of Goodyear Aerospace either.

Miya: That is true too yes; so Westinghouse did produce a few computers right?

Michael: So Goodyear.

Miya: Right I've seen the Goodyear produce computers in fact I think I saw one of them flying over the other day.

Michael: No I wasn't thinking of that, I can't remember the guys name now, who is at Goodyear?

Miya: Ken Batcher and a whole bunch of other people, Jerry Potter and others, I don't know if you met Jerry Potter or not. I was reading about Jerry Potter the other day. Okay next person, Jim Brown, UT Austin?

Michael: Well we retained Jim Brown as a consultant, a general consultant I would say, you know, he had the credentials in physics and in computing, he had some good students, not the least of which was Elizabeth Williams.

Miya: She's next.

Michael: Yeah I never could find her or Ralph Carlson for that matter, they never showed any spark of interest, you know, it was just a job.

Miya: Well let's stay on Jim Brown for a moment, so wasn't he in the Chemistry Department at UT Austin or something like that?

Michael: Not that I remember no.

Miya: He did a TRAC, Texas Reconfigurable Array Computer right?

Michael: I don't know.

Miya: Okay, but he was just a general consultant.

Michael: The TRAC to me means something like Snowball text processor that Henry Mall did.

Miya: But it wasn't the parallel processer at the University <inaudible>.

Michael: No I mean it's a software product.

Miya: Yeah, no I meant the Texas parallel computer.

Michael: The Texas reconfigurable thing I don't know anything about.

Okay, so anyways, finish up with Jim Brown.

Michael: We retained him, you know, he'd come up and we'd just talk, I don't remember anything except maybe, you know, he would send students up to our place then.

Miya: Of whom Elizabeth and Ralph were among two.

END OF TAPE 6.4

START OF TAPE 6.5

Miya: One of the things you remember about the Cray.

Michael: Yeah was that with the development that had been initiated at the Supercomputer Research Center which is now called CCS was the PIM [Processor-in-Memory] chip and that was just for Boolean functions, Boolean operands, but I didn't see any reason why we couldn't have it work for floating point numbers as well and that would have given us say 10 to the 14th floating point operations per second, that's pretty fast but no one could ever, you know, they actually had the PIM chip working and there's this guy at CCS has still got it.

Miya: Ken Iobst.

Michael: Iobst, right, and he still has it.

Miya: In his garage as I recall, I talked to him at Salishan remember?

Michael: No I don't remember, but alright.

Miya: About 7 or 8 years ago, he's going to give it to the Museum when he gets--- but he's not running it or anything, he doesn't have any money. He picked it up for an incredibly cheap price like $10,000 or something like that. Okay, so that Cray-3 SSS, I think is what it's called and they built a follow on, I think it was the T90 MSSS.

Michael: I didn't ever see that.

Miya: I haven't seen it either so but yeah I remember talking to Ken; it's just a matter of time. Speaking of Ken, speaking of CCS, Elizabeth Williams, say all the things you want to say about her.

Michael: I got to know Elizabeth when she was at Los Alamos and she was a very ethical person and a very conscientious person. I never found her with a burning interest in mathematics say; it was just a thing she had to do and she could do it and I tried to, you know, not help but answer the questions she had when they came up and stuff like. So for a while I was her consultant down at CCS and I was a great admirer of hers, you know. I really liked her.

Miya: I know you did. Let's see, did you spend time at CCS like a year or something like that, was that right or SRC at the time?

Michael: Well it was SRC when I started yes and I didn't.

Miya: She was there for a year.

Michael: I was there for a while I don't remember it being a year.

Miya: Should we say something about that or?

Michael: No.

Miya: Skip Ralph Carlson?

Michael: There's another guy, you know, he doesn't ooze the interest in mathematics that I think a PhD mathematician should have.

Miya: Have we said enough about Elizabeth though?

Michael: I think so, you know, she was a great person and she was doing something that was important to do.

Miya: Okay well we won't add additional stuff about Ralph Carlson. Edward Teller, George, your boss, Edward Teller?

Michael: I'm listening, I'm just thinking about Edward Teller; Edward Teller was probably in my lifetime the number one political physicist in the United States. He was an extremely outspoken person and from other things that I could read and/or hear about, he didn't really worry about anybody but himself. Yet the country owes him a lot, just as it owes a lot to Oppenheimer and Seaborg [?] and I think the best thing you could say about Edward is that, you know, with him being improperly influenced by Lowell Wood, he would have been better off listening more carefully to alternate voices okay, alternate opinions.

Miya: Was that it?

Michael: Well, yeah, I don't see anything else, you know.

Miya: The next person you have here, which I think is related to Teller is Fred Wooten from Department of Applied Sciences at UCI.

Michael: Yeah, well at what?

Miya: Fred Wooten about DAS.

Michael: It's not UCI it's UCD.

Miya: UCD that's right.

Michael: Okay, Fred also did some diagnostic physics in weapons testing at the lab and when the idea of constructing an department of applied science was first broached, this was an Edward Teller idea, Al Kersbomb and maybe Harold Smith and maybe Fred Wooten too for that matter. But at some point Wooten got to be chairman of the department too and be candid in saying that it was important to have him on our side when we were trying to set up the so called Computer Computational Science

Department at Livermore at DAS but he was very nice to me and, you know, I told him what we were going to do and how we thought we ought to do it and he corrected a few things and that's what we did, that's enough, gentle, sweet person.

Miya: Next person, Lowell Wood?

Michael: Interesting person, given that there's such jealousy and competitiveness all up and down the physics temple, Lowell may have succumbed to a couple of temptations in the sense that he advocated ideas which weren't right about how physics works or how to do this or how to do that but as a person who has breakaway ideas, I think he's probably one of the leaders at the lab, you could always. You're not going to find prosaic nonsense with him, he's going to be out in front leading, okay, and I think he is responsible for bringing more bright young people to the lab then any other person.

Miya: Do you want to say anything about the computing requirements for strategic defense with regard to Lowell?

Michael: Well this is 1969 when I got back from Haverford and got pulled into a meeting and it was basically what are we going to do with the STAR. Lowell proposed a couple of problems or, you know, said what had to be done and I could see in a sense how the STAR would solve them, like it could process as an arbitrarily long vector memory-to-memory so that it could, you know, look at a scene on the ground and run the filters over it and stuff like that and do that in real-time and better all day long, wouldn't have run out. He was responsible for instilling a great esprit de corps in the young people that came to the lab.

Miya: Is that it?

Michael: On balance he wasn't very friendly but I don't know, yes that's certainly enough.

Miya: Stan Ulam and his wife Françoise?

Michael: Yeah, neat guys, of course I met him at Los Alamos a couple of times, but we got to know each other at a NATO summer conference in Achen and the only thing I regret there is that I didn't get his autograph on my copy of his book, *Adventures of A Mathematician*, you know.

Miya: The Misadventures of a Mathematician.

Michael: Whatever you say, I think he was a great guy and his wife was an incredibly warm person and we could use a lot more mathematicians in this country like him, would be really good.

Miya: Anything else you want to say?

Michael: Yeah, I was going to say we got him in a sense because Harvard has this Junior Fellows program and he could fit into it.

Miya: You mean Livermore or Los Alamos?

Michael: No, the country; these refugees from Europe could come over here and didn't suffer okay, they got good jobs and stuff like that, that's important and I'd like to see more of that happening, I don't see it.

Miya: Anything else you could think of for Stan Ulam or Françoise?

Michael: That's about enough, yeah.

Miya: Frank Kwo?

Michael: He was a summer consultant to Waldo Magnuson at the lab and we just naturally, Crawley and I and Frank and Waldo formed a nice quartet okay and Frank came from Bell Labs and he's a real politician, a very suave individual and I enjoyed my first ever visit to Japan through his eyes because he could read the signs and translate them, he knew the language that well.

Miya: Did I ask you enough about your visit to Japan in all these interviews?

Michael: I don't know.

Miya: Maybe I didn't; anything else you want to say about Frank?

Michael: No, I don't know where he is now, he's probably at SRI.

Miya: No he's retired, I think.

Michael: Retired, okay.

Miya: Actually I see a bunch of names here who are coming next Thursday, I forgot to add they're basically Los Alamos people, I'll just do them right here and now. Jack Worlton?

Michael: I think of him as a great scientist, you know, a person who's learned and has led in the early days of computing and I think he was recognized down at Los Alamos by virtue of the fact they made him a fellow and, you know, cut him loose to do what he wanted to do. He is a very, very good speaker.

Miya: And he wears bolo ties.

Michael: That's okay, I like him.

Miya: Yeah I do too. Anything else about Jack?

Michael: No.

Miya: Bill Busby?

Michael: I was just going to mention next Thursday he'll be here, but then he's coming over to my place Friday morning… we'll have breakfast together, you interested?

Miya: I don't know if I can make it, we'll see. I have a three-day meeting from Tuesday to Thursday.

Michael: On what?

Miya: Commercial space, I have to help.

Michael: Commercial what?

Miya: Commercial space.

Michael: Space?

Miya: This is irrelevant, it's my life not yours, we're talking about your life, don't worry about it. Bill Busby?

Michael: What I called him was the 'Soft Talking Texan.'

Miya: The what?

Michael: A soft-talking Texan, he's a math major, again I'm not sure how animated he his by mathematics but he has a degree in that and he was one of the principals in management at Los Alamos and was my partner in crime when we set up the Salishan meetings, I also went to him for some help with the setting up of the Supercomputer Meetings. What I learned through all that stuff was that the quality of your goals is not going to save you but the persons that you pick to run interference for you will save you or could save you. I helped Bill in the sense I wrote a recommendation for him when he left Los Alamos and became the head of computing at NCAR and I even had some particular inside juice because it turned out that Chuck Leith was somehow involved and I could say directly to Chuck that Bill's a good guy and he won't do anything that's crazy on you like that.

Miya: Well speaking of crazy things, weren't the three of you going to found a company, BMW wasn't it?

Michael: Yeah that was I think Jack's idea, I don't know that I ever had anything to do; I would have loved to have done it but I don't know.

Miya: Don't you think that name's been taken already?

Michael: What BMW? Well not by as good as us, we're better.

Miya: Okay Bob Ewald?

Michael: What can I say about him, you know, he was a great person, to me very important when we set up the supercomputer meetings because I was able, through him, to get Seymour to give the keynote speech at the first meeting.

Miya: Anything else? Bob Horomoto?

Michael: I don't have anything to say about him.

Miya: Richard Hamming?

Michael: I'm a great admirer of Hamming.

Miya: Hamming's alternative musings.

Michael: Yeah well he also said it's an absolute surprise how well mathematics describes nature or something similar to that, okay, and I liked his attitudes which we discussed many times about like if you had taken or used Fortran you couldn't get into his class down at the Naval Postgraduate School.

Miya: He's a dear friend when he was still alive.

Michael: No he's not a bureaucrat he was a…

Miya: Dear friend.

Michael: Oh yeah.

Miya: He was a dear friend okay.

Michael: Yeah alright, the point is it's I think what the Dutch guy, has the same attitude, you know you burn your brain on Fortran or something like that, you won't think of what good structures are like.

Miya: Oh, Dijkstra, you mean?

Michael: Dijkstra, yeah.

Miya: Nick Metropolis?

Michael: He's a great person and I didn't get to know him nearly as well as I'd like to. We always visited when I went to Los Alamos and, what should I say, I learned a lot about the inner workings of their arithmetic processors through Nick, okay, but in a sense he was from a different generation than I.

Miya: Well actually, you know, you're talking about Richard Hamming there for a moment, you were surprised how math describes nature, actually I think who gets credit is Eugene Wigner… from whom he was copying.

Michael: Oh really?

Miya: I think Wigner even has a more impressive paper, that's one actually I can understand, it helps to know how to integrate two dimensions.

Michael: Well, you know, the basis of the sciences was developed in the 1800s and later okay… [it] is the basis for extending science through computation today, that says the scientific truth is scientific truth, it's not going to change. We're finding new representations now and maybe some new insights because of that but so whether it's Wigner coming up with this discussion or Szilárd, any of those other guys it's--- they share those very, very common heritage.

Miya: How about Michael Tatusose?

Michael: The golden Greek, he is mostly--- to begin with he was responsible for me pushing into the data file language whole adventure and I'll tell you one thing, I was at MIT which is a place I always stopped if I'm on the East Coast and I talked with Mike and I talked with Ed Fredkin and I talked with Jack Dennis and I came back and I talked with Gus Dorro the AD and the long and short of it was that I set up a dinner party at my house when we lived in the big mansion thing and the dinner party would have let's see, Gus Dorro, Bob Crawley, Norman Hardy and myself. Would have Mike Tatusoso, Joel Erin or Joel Moses, Jack Dennis and the purpose of this thing was to get together and hash out a computing program or a research program or a little of both so that we could get fast, you know, the thing from the point of view of the lab was that we're going to have computer structures that will naturally lead to much, much faster computations. The point of view from MIT and stuff like that is they're going to have a stimulus that will, you know, animate their efforts when they try to produce this faster thing or this bigger thing, whatever it might be, so it looked like a thing that everybody could win on.

Miya: Jim Warren?

Michael: I don't know what to say about him.

Miya: Ken Olsen, although you said some stuff about him already.

Michael: I've already said something about him.

Miya: Harlan Anderson?

Michael: Already said something about him.

Miya: Donald Knuth?

Michael: I haven't said anything about him.

Miya: Other than calling him a minutiae man by Hamming.

Michael: That's what Hamming said, yeah, and I personally can stand and look at my bookcase at the number of books he's produced, I don't have any of the books on typography but I have all the others and papers that's he's published and it's just incredible how much that guy can put out. Because I can't do it, I'm in awe of that kind of stuff. We invited Donald to come over and give a distinguished lecture, he didn't do a very good job of that but then after we were through we adjourned to somebody's house and Heidi made a spaghetti and meatballs feast with big pieces of garlic bread and some homemade wine and it blew down a lot of the water. He loved it.

Miya: Why don't I do the names real quick and you can decide who you want to say something about in the last two minutes. McCarthy, Minsky, Feigenbaum, Dijkstra, Bob Hughes, the Dennings, Peter and Dorothy, Bob Taylor, Steve Squires, Bob Kahn, Bob Cooper, Maurice Wilkes, Brian Randall, Bill Burke, Ed Fredkin, Bob Jastro, Herb York, Sid Caron or Norris Parker-Smith?

Michael: We can't cover those today. I would like to say a few things about some of those guys.

Miya: Say something, you have two minutes.

Michael: Well what was the first one you named?

Miya: John McCarthy.

Michael: I think that John gave me the courage so to say to go ahead with the Octopus Plan when I described it to him at MIT one visit and he did that okay, he gave me that courage.

Miya: McCarthy was still at MIT not Stanford?

Michael: At MIT, right, and since then, you know, we've had him on a cleared basis for the Lowell Woods thing and I helped him with going around the lab seeing this, that or another thing. I think of John as a absolutely elegant mathematician.

Miya: Minsky?

Michael: Arrogant.

Miya: Arrogant okay.

Michael: Feigenbaum, my mother said don't say anything if you can't say something nice.

Miya: Dijkstra?

Michael: Arrogant but I liked him.

Miya: Bob Hughes.

Michael: That's our Bob Hughes?

Miya: I have no idea who this Bob Hughes is.

Michael: We have a guy Bob Hughes, we call him Bob Huge, he's a 6'4 black guy and he was one of the persons who helped produce Fortran.

Miya: He's African American?

Michael: Yeah.

Miya: Most people don't know that, okay.

Michael: Well I think of Bob as an absolutely great gentleman okay and, you know, he was sent by the lab as the lab's representative to John Backus for working on the Fortran thing. I've forgotten exactly the role, I have it written down in my interviews what is the role that Bob Hughes had in this whole process and he was a good compiler writer.

Miya: Peter and Dorothy Denning?

Michael: Denning?

Miya: Yeah.

Miya: Do you want to go to Bob Taylor?

Michael: Yeah I think that I can't say much about any of those except, you know, I knew him and I liked Bob. First met him when he was kind of Ivan's assistant at IPTO and then, of course, I met him after he came out to Xerox PARC and I think he gets a lot of credit and he deserves it, okay, for making sure <inaudible> good, stuff like that.

Miya: Steve Squires?

Michael: I don't like him too well, well, you know, I think that he was an aggressive enemy of the Cray machines and stuff like that. But he was perfectly willing to pour 20 million dollars a copy down the rat hole to get those Thinking Machines built and I don't understand that, okay?

Miya: Bob Kahn and/or Bob Cooper?

Michael: Well I think Bob Kahn.

Miya: He's going to be honored in a month, you know, as a [2006 Computer History Museum] Fellow?

Michael: That's alright he's a good guy and a visionary.

Miya: How about Maurice Wilkes and/or Brian Randall?

Michael: I knew them both, you know, and in fact when we were in England we spent several days with Brian, okay, and Maurice spent several days at my place in Livermore when he was here as a DEC employee or whatever, just like Grace Hopper was a DEC employee also.

Miya: Bill Burke?

Michael: That was a great, great person and I find myself thinking about him all the time, you know, it's like with the case of the loss of Seymour, you know, I don't see how to compensate for those losses.

Miya: Ed Fredkin?

Michael: Ed is maybe a genius also but I got to know through MIT and through DEC and we bought a lot of the stuff from his company eventually and he was a good guy, okay? I think maybe a little too gentle in some cases with his own kids but that's his business.

Miya: Bob Jastro?

Michael: Bob Jastro was one of the two people who were intimately involved with the lab in 1952 and he is one of the persons who recruited Bryce DeWitt who was the first person to write down the La Grangian expressions for two-dimensional hydrodynamics, okay? I'm trying to get that worked up into a unit that will be published on my history site. Jastro I can only remember one thing: he had a powder blue Oldsmobile convertible in '53, when they were scarce. We left a note on the seat of his car saying we're sorry that we scratched your car but we can't afford to have to pay for it, something like that, okay? When he went out and we watched him then from the window, he went out and he picked up this note and, you know, stopped and he looked around and he walked around the car, not once, three times okay. He never found out who did that as far as I know. I interviewed him down in Los Angeles.

Miya: Okay, you said Bob Jastro, 1952 involvement, recruited Bryce DeWitt right, Bryce wrote the first 2D La Grangian expressions for hydrodynamics.

Michael: Well, you know what the La Grangian is?

Miya: Yeah.

Michael: Okay he wrote two-dimensional equations of motion which is hydrodynamics.

Miya: Alright, this was Bruce, though not Jastro, who wrote that though right?

Michael: No, he did this.

Miya: Jastro did this too.

Michael: Not that I know of.

Miya: The question is about Jastro, but he hired.

Michael: Jastro was trying to do computation on computers that we were going to buy and he was in New York doing it with, I mean Sid was there at one point and perhaps Bryce was there too I don't know. Edward Teller was the guy who sent them there and, you know, he didn't stay at the lab very long after that and so for the next 50 years I have no knowledge of him. Then I interviewed him and he was nice but even though we had a date, he wasn't ready to do much so we didn't have much of an interview and he was working as the head of Mt Wilson, astronomy environment or telescope up there. He recruited Bryce DeWitt, Bryce DeWitt was big in the country as a quantum gravity-type person, but there are people trying to trace the genealogy of Bryce DeWitt from when he produced the La Grangian of these hydrodynamic equations of motion. He wanted to equate that until Bryce became fairly famous in quantum gravity, you know, what did thinking about this geometry thing do to his head in the process, things like that. It's all tied together and if you've ever gone to my site at all?

Miya: Yes.

Michael: Okay well I don't have this one completed yet but it'll be Bryce and Jim Wilson the astrophysicist talking about Bryce's development of the first time of these equations. There was nothing wrong with the equations, there was a question about how best to difference them, two reasons, accuracy and stability. You have that problem, but a big problem is that in order to get any reasonable data out of this model, you've got to have a large number of points on one hand, and you have to run it for a very long time on the other, and both those things are difficult, were especially difficult with old computers. Univac had a thousand words of memory, that's not enough.

Miya: Sid Carron?

Michael: What can I say about him, he and Norris Parker-Smith, Norris, you know, produced or had something to do with *High-end Computing News* magazine.

Miya: Is that all you want to say about Norris?

Michael: Well, yeah he was at the lab before he started that and I got to be fairly friendly with Sid because he was on the board of the supercomputer conferences and, you know, he's a fairly light-footed guy so I don't know where next I'll see him.

Miya: Herb York?

Michael: Herb York was a graduate student at Berkeley in physics and he answered the call that Lawrence laid out after the war, where he and Chuck Leith had been at the Oakridge laboratories with the hardware. Now after the war, he's obeying the call of the wild from I don't know whatever you call that place.

Miya: So York came to Livermore.

Michael: Yes that's because Laurence asked him to come out there and head the place up, okay?

Miya: So he was the first director?

Michael: Yes.

Miya: Is there anything else you want to say about that, I think we actually have a whole bunch of your commentary about him earlier?

Michael: I don't remember.

END OF SESSION 6

END OF ORAL HISTORY

