Oral History of Gordon Bell, part 2

Interviewed by:
Gardner Hendrie

Recorded July 19, 2005
San Francisco, CA

CHM Reference number: X3202.2006

Hendrie: Continuing his oral history for the Computer History Museum and I think we left off, if I read the end of your last transcript, you're still in Australia and you thought it was about 1959. I'm not sure about the date.

Bell: Yeah, right. Well, actually I left Australia in August of '58.

Hendrie: Oh, so it was not '59, okay.

Bell: No. But anyway--

Hendrie: We'll get the dates right.

Bell: The tail end of Australia. One of the last things I did in Australia, and I don't know when this was, was I wrote a program for the DEUCE that proposed to Gwen Bell. You ran that program from the toggle switches and you entered yes or no and there was a little flowchart program with a loop in it. And there was a display on the Deuce which was actually part of the memory that you could put messages in essentially line by line so that 32-bit lines bits would come down in a scrolling fashion. And so, it was a 32 by 32 points.

Hendrie: Display, so you made the characters?

Bell: Yeah, so it was a 1,000 point display and then you sort of made these little dot matrix characters on there. I don't know how many lines, not very many lines, that you do 32 lines.of probably 5 x 7 or maybe smaller like 3 or 4 by 5 or 6.

Hendrie: And it was basically showing what the memory is?

Bell: Yeah.

Hendrie: This was a bit memory map.

Bell: It was a memory map.

Hendrie: A bit memory map.

Bell: Yeah, word by word, and so words would shift or scroll down from the drum and you could sort of slide these words down through there. So I wrote this program and then Gwen and I left Australia somewhere mid-August, arrived back in the U.S. in November. But anyway, let's see, I guess there were a couple of memorable or one memorable incident coming back. The first stop was Singapore and Gwen had a classmate there, a Chinese architect classmate, who we spent a few days with. His name was Bill Lim. And then we went to Thailand and spent a few days in Bangkok and then to Burma. Very few people had ever gone to Burma so we had a few days in Burma, then Turkey, Istanbul, and then Athens and then through Europe. And we spent until probably I think November, mid-November doing all of that,

sort of crisscrossing Europe, being in London several times. I don't know, I think we were in most of the countries. We were in Holland, Germany, maybe Austria, Italy. We weren't in Yugoslavia because we took the boat from Athens back to Italy and then up. The memorable incident from a computing standpoint was we had written the SODA program to make it easier to program the DEUCE and to allocate software to storage and have a single level store. Essentially, it was the equivalent of what Manchester did to make a two-level store into a single level as and I think I mentioned that earlier.

Hendrie: Yes, you did describe that.

Bell: And then we went to visit the National Physical Laboratory in Teddington, outside of London. I don't remember whether I mentioned that.

Hendrie: No, you did not.

Bell: And, god, the name will come back, but anyway we -- ah *as the name returned to my memory* -- went there and visited Wilkinson, James H. Wilkinson, who was the famous numerical analyst. I went in there and talked about the program that we had just written. By the way, the paper ultimately got published in the British Computer Journal.

Hendrie: Oh, okay.

Bell: That was I think actually my – our -- first program and then our first publication. We were sitting around having tea and then Wilkinson said "What can you do with this program?" And I said, "Well, you can really write a matrix routine in like ten or 15 minutes. It was just basically a few lines of code. And he said, "I can do that with the DEUCE code" and so . . .

Hendrie: He could write it just in . . .

Bell: Anything, yeah.

Hendrie: In straight assembler, okay.

Bell: Right. And then I told him about this when he came to the museum in Boston in '79 I think and gave a talk. He said, "Yeah, I was pretty cheeky then." But he spent a lot of his life working at Stanford as a visiting scholar.

Hendrie: Oh, really?

Bell: Yeah.

Hendrie: Okay, so he eventually emigrated.

Bell: Yeah, but he's a great numerical analyst.

Hendrie: Oh, that's a good story.

Bell: Yeah.

Hendrie: And then eventually you came back to the United States.

Bell: Right.

Hendrie: Now, did you know what you were going to do when you came back to the United States?

Bell: Let's see, at that point I don't think I knew exactly what I was going to do. I'm not sure whether I had an offer from MIT to come back as a staff engineer for research. The Division of Sponsored Research had essentially staff engineers, and I think I may have had an offer from Ken Stevens. By the way, in fact, I sort of woke up in the middle of the night thinking maybe I ought to bring my computer over to this interview because I recall having seen the offer letter from MIT or Ken in my scan pile, and so I would like to query my memory to see, in fact, exactly what the offer was and when it was made and all of that as a test of my own MyLifeBits system that holds my entire live.

Hendrie: Yes, all right, yes.

Bell: And then I know the original DEC offer is in there too. But anyway, I interviewed there at MIT, and I know I interviewed at EPSCO. 2008 Addenda: At some point, I interviewed at BBN with Ed Fredkin and JCR Licklider (Lick) and they didn't hire me. The reason I remember this is that Ed has quoted it to me several times. My point in bringing this up is that I have been extremely lucky in the choice of jobs and BBN was clearly not a good move for me. So I have been lucky in not getting a job.

Hendrie: Oh, with Bernie Gordon?

Bell: Bernie Gordon's company and so I basically decided . . .

Hendrie: So you basically came back with some ideas of where you might go and where you would be going?

Bell: Yeah, and I had some offers before that. I think Philco in Philly was a potential. I had interviewed there. I don't know if I had an offer or anything like that, but certainly GE because I was a co-op. As a co-op student I would have an offer there. I also had an offer from NCR that included a written intelligence test.

Hendrie: Now, it was Stevens at MIT; what was his position?

Bell: Okay, Ken Stevens, who is ten years older than I am, was head of the Speech Research Lab and he was my thesis advisor when I made the sound level recording meter for recording histograms. And actually it was Dick Bolt, of Bolt, Beranek and Newman [BBN], who had started me out. He had stimulated me to write the thing, and then Dick left MIT and Ken took over as my advisor and I finished

the project. And then, when I came back, Ken hired me to work on speech, so my first real job was a staff engineer at MIT working for Stevens and then working on speech and eventually on instrumentation at MIT's Instrumentation Lab. In fact, what we started with was a filter bank and, I don't remember, there are maybe 18 or 24 filters. These are tuned filters and they were at 200 or 300 Hz wide or less depending on where they were in the band and that covered the speech spectrum. And then you looked at how much energy was in each of those bands and so you got essentially a time series of what the sound was. And basically that's the fundamental technique that almost all speech I think still uses today, which is frequency.

Hendrie: That's where the data is.

Bell: Work in the frequency domain, frequency versus time analysis. In fact, the thing that people normally see is a display called a sonogram, which is a map of the intensity of each frequency in the Y axis. So you see the darkness of the frequencies versus time. And so, you look at this plot and people can actually read sonograms and determine what someone is saying. I mean not a lot of people can do that. 2008 Note; I saw this used in Australia to single out rare birds in a rain forest.

Hendrie: But it is possible to do.

Bell: It is possible, and so given that, I thought, okay, well this will be great because, I mean, people can read this stuff. I'll work on a program that can read speech.

Hendrie: Oh, so you're saying, yeah, so . . .

Bell: So, I would do speech analysis, and we had the filter bank and then I worked on an instrumentation where you basically took a loop of tape with a saying on it. Oh, I think these were five second loops or something like that, and so we went around this tape recorder and the thing was you glued its end together or taped it together for a loop.

Hendrie: Yeah, right.

Bell: And then on the stereo track put a timing mark, which was the sampling time, and then used that sampling time to then….so you made 24 trips around the tape and in doing that you….then I had a stepping relay that would go and move to the next.

Hendrie: Move from filter to filter to filter.

Bell: So essentially you get a sonogram, a plot of frequency shown by intensity or blackness versus time. That was my first task and then we built and started working on speech analysis. And I don't know where exactly the idea came from but we invented a process which we called "analysis by synthesis", which is fundamentally a technique that people still use to recognize a phrase. Fundamentally it's still in use on things like you're going to make an ideal synthesizer for a piano and you want to get what the waveform is. So you basically take the waveform of a piano striking a string and then you take that and you synthesize what you think are the components of that, the perfect tone and then there's all the harmonics and any kind of distortions. And so, you basically start throwing synthetic terms at it until you get the same sound as the input.

Hendrie: So fundamentally it's the same spectral band with all the same energy levels.

Bell: Right. And so fundamentally you "analyze" the content by synthesizing content and then you know the parameters of what that synthetic term is. And then you can sort of apply that back to the vocal tract where you move the vocal tract parameters around, and then you see if that's what gave you those terms, those signals.

Hendrie: Oh, all right, that's fascinating.

Bell: So in a sense the technique recurses or . . .

Hendrie: It's a trial by error recursive just getting the errors smaller and smaller each time.

Bell: And so different ways of getting which errors, what error do you want to look at, and so that was the basic method of getting the parameters of what's going on. And then we wrote a classic paper on that because it was also being applied at the same time for some handwriting analysis.

Hendrie: Okay, oh interesting, all right. Do you remember what year or at what time you were doing this work?

Bell: Now that was. . . So I started at MIT in January '59, so part of the constraint here was Gwen had to finish her Master's degree at Harvard, so I didn't want to go too far from Harvard .

Hendrie: Yeah in different cities.

Bell: Right. And so, I figured MIT was just the right thing because there was a project orientation. I liked the people and got that program working. Then a little later I went over and worked in another part of RLE at the Instrumentation Lab and worked on a thing called pulse mode computation, or pulse hybrid computation using analog components for simple inner loop like multiply and add. This was totally a diversion that shouldn't have been because of the analog nature. It just happened to work on the TX-0 because it had no multiplier.

Hendrie: RLE?

Bell: Research Laboratory for Electronics.

Hendrie: Got it, okay, got it.

Bell: I think it's still named that or may or may not still be named that. But it was located along Vassar Avenue in Cambridge called Building 20 and these were World War II constructions, plywood constructions, were noisy and it was filthy. I mean lots of truck traffic and you'd go away for a few days and your desk would be covered with sand when you came back. But, anyway it was a great place.

Hendrie: So, what did you do there now?

Bell: So, then I went over . . .Well, I guess the important thing about the speech stuff research was that I thought, "Oh, my god, this is not for me. This is basic research and science." I like to build stuff. I want to be an engineer.

Hendrie: Yes, okay.

Bell: And I've done the easy part, which is to get a program that, in fact, could get to this point of recognizing, of giving you what the components are coming out of the program.

Hendrie: I can give you the data.

Bell: Yeah.

Hendrie: Now, we need a little science.

Bell: Now, what are you going to do with it, how are you going to look at that, and I did it to understand speech. I think maybe we had a few trial recognition programs and several components to segment, or extract pitch of or something because you know where the energy is, and then did some segmentation and stuff like that. And then at this point in late 1959, I said “this is just too hard.”

Hendrie: And it did prove to be too hard didn't it?

Bell: Yeah.

Hendrie: 20 years and decades went by before people figured it out.

Bell: My estimate was 20 years, but I had no basis for that estimate, and then it's been about 40 actually to get it to a useable general recognizer, although after 20 recognition was in use.

Hendrie: Exactly.

Bell: People were making products in 1980-ish or so. So, along about mid-'59 I then switched to the Instrumentation Lab headed by Frank Reinjes, a principal engineer who worked on radar at MIT’s Radiation Laboratory. I was working on pulse computation because it turns out the TX-0 didn't have a fast multiplier. In fact in the speech program, because it had multiplication involved in it, I did multiplication by table lookup of logs.

Hendrie: Oh, really?

Bell: Yeah, to get the speed and so . . .

Hendrie: And this program was written, ran on TX-0?

Bell: Yeah, right.

Hendrie: I didn't ask you what computer this ran on.

Bell: Oh, yeah, this all ran on TX-0, so I was a proficient programmer on TX-0 and wrote this fairly good-sized program for the day to do speech analysis. Others used it for several years.

Hendrie: And the instrumentation was tied to TX-0?

Bell: Right, and then that was all part of the speech lab. And then I went over to this other lab, did this project -- again I think it was the Instrumentation lab -- and built a multiplier that would allow you to basically do sample and hold, basically put values on a multiplier and then pick those off and then feed the results back in. And so it was called hybrid computation and it was for a particular program. It was auto correlation or cross correlation of radar data, so I had a lot of data and needed to do it.

Hendrie: Okay, so fundamentally the multiplier, it was a digital computer but you used an analog multiplier?

Bell: Yeah and I think it had an integrator too, so you could use . . .

Hendrie: Oh, my goodness, yes.

Bell: So, anyway, we were able to do radar signal processing in real time. There was a paper report published on it in August, 1960. And then at the time the TX-0 was coming up and they needed a director to run it and I interviewed for the job.

Hendrie: Oh, my goodness.

Bell: With Wes Clark. I think Wes was kind of "overseeing" the TX-0 from Lincoln. Jack Dennis ended up doing that.

Hendrie: Oh, he ended up getting to be chosen to be the guy.

Bell: Right, to run it and sort of be the faculty advisor and whatever.

Hendrie: Yeah, right.

Bell: In fact, it probably undoubtedly had to be a faculty, or should have been a faculty member 2008 Addenda: In retrospect I was really lucky to not get the job. I have been lucky to have been rejected by two jobs (see my addenda re BBN), neither of which would have been good for me. If I had accepted either one, it would have been a dead end.

Hendrie: The PDP-1 had not been built yet had it?

Bell: No, no.

Hendrie: So, they didn't have a PDP, Jack Dennis didn't have his PDP-1 yet?

Bell: Right, and that's how I got to DEC. In that process, Jack wanted a tape unit on it so I designed a tape controller for the TX-0.

Hendrie: Oh, all right.

Bell: And I needed modules for it.

Hendrie: Ah, yes, exactly. You're not going to do it from scratch. You just did logic design.

Bell: Right, so I ended up getting modules from DEC, doing a design, and that's how I met the people at DEC. There was one particular circuit that you "needed" that turns out to be very useful and it's hard to do any other way, and DEC was the only one who had it. It was called an integrating single shot so that every time you pulse it, it would start and take a fixed amount of time from the pulse input. So, as opposed to just a normal multi vibrator where you hit it and it is unclear how long before it stops.

Hendrie: It gives you a fixed amount and you can change the capacitor and it will do that.

Bell: This thing actually integrated and so from the last time you hit it it would generate a pulse.

Hendrie: Ah, so it integrated and depending on how long it was between characters on the tape.

Bell: Yeah, right. It was the restart of all of this that . . .

Hendrie: Ah, okay.

Bell: That was critical for a tape because you didn't have the sort of nice properties.

Hendrie: I see and 3C didn't have it in its module either?

Bell: It was a very tricky circuit and I think they had invented it for the tape, the same thing for the tape controllers at Lincoln Lab for the TX-2. It is also called a boxcar circuit.

Hendrie: Okay.

Bell: So, anyway, I did that and it turned out that just in that whole process I ended up getting a job offer in May of '60 from DEC.

Hendrie: Now who did you interview with?

Bell: I interviewed with Ben Gurley. Ben was the guy, my primary interface there because it was about building this tape controller. In essence, I asked him, “What do you think of my design?”

Hendrie: Right.

Bell: Dick Best was there as chief engineer from doing circuits, and then in that process I met Ken and Andy. I met the whole team and got a job offer and I was number two as a computer engineer working for Ben.

Hendrie: And you decided to take it.

Bell: I took it, yeah. It was exactly what I wanted to do.

Hendrie: You were going to build stuff.

Bell: I was badge number 80. It didn't have the problems that I had seen as a co-op student at GE where there was a sea of desks. Everybody had their own little office, which was uncommon at the time because I remember my friend Bob Brigham from Bell Labs was visiting and he said "Oh, I'm impressed". The offices were made of hollow core doors. You know DEC was in a civil war woolen mill building. One company was milling doors and we bought rejects for partitions.

Hendrie: Yeah, of course, with the lanolin floors.

Bell: And so a company that made doors was in the Mill Building and they'd just go down and buy a bunch of hollow core doors and put these doors up for partitions.

Hendrie: Oh, okay.

Bell: There weren't doors on the door partitions but they were sort of all cubicles. Anyway, so that's how I ended up getting to DEC and one of my first jobs I think was at . . .

Hendrie: But did you ever finish the Mag tape?

Bell: No, I didn't finish it.

Hendrie: Oh.

Bell: It turns out that Bob Spinrad who was at Xerox PARC still cusses me for having to debug my controller, and in fact he said he ended up as my technician to finish the debug. So, it got wired and ultimately worked.

Hendrie: It ultimately did work but you--

Bell: It ultimately worked but I . . .

Hendrie: You had bailed.

Bell: I bailed out, yeah, just in time. There's nothing worse than having to debug a tape controller.

Hendrie: Debug something that interfaces an electromechanical device?

Bell: Yes, and tape is by far the most difficult. In fact, of course, it's always the job you give to the youngest and least experienced engineer.

Hendrie: Right.

Bell: When a new person comes in, you say okay, what do you want him to do? A tape controller.

Hendrie: Right, it usually isn't critical about exactly when it gets done. It's never on the critical path. You just have to have one.

Bell: But, anyway, I think I designed one of the units there. Another friend, Jack Brown who recently died, I think ultimately debugged that one too. We had two tape controllers. One was a programmed controller. In fact, the TX-0 one was I think a highly programmed one and so it was a good experience. All these things were good experience in terms of the I/O. In fact, that's how I got such a fondness for the I/O and also doing as much as possible in software.

Hendrie: Yeah.

Bell: Because you could trade off program for work in hardware control.

Hendrie: Okay, so it was, yeah, so TX-0 did a lot of the work.

Bell: Yeah, and the PDP-1 had good interrupts so you did buffer assembly and then you sampled the data and then brought it in with a program.

Hendrie: Okay.

Bell: And then we made one like that for PDP-1. It ran through the direct memory, DMA channel. And then when so I got there in August of '60, I wrote some programs and worked on various logical design problems.

Hendrie: Now, had Ben finished the PDP-1 yet?

Bell: Ben had made the PDP-1 number one and it was sort of a brown speckled one with a remote console and it had, of course, a huge number of wires coming to the separate console. It was a separate console just like larger machines including the 7090 and the CDC 3600.

Hendrie: Oh, my goodness.

Bell: With a table and then almost very much like the PDP-1 console today but yet just remote, so you had drivers and switch filters and stuff like that to deal with the remoteness. The worst problem was the connectors were unreliable after you used them a few dozen times.

Hendrie: All the stuff you have to do.

Bell: And I think I was the one who said, “God, let's get rid of this console, get this thing back out of there and put it on the computer” because it had this big bundle of cables for lights and switches. Nothing was doing much and there was a particular connector on it. I learned to hate connectors too at that point. You'd open it and close it a few times and it was a leaf, one of these kind of leaf connectors and multiple pins and lots of problems.

Hendrie: Oh, wow, okay. So now were you told what you were going to do when you were hired?

Bell: Just different things. I know I wrote a floating point package for the PDP-1 and I also was sort of instrumental in starting DECUS. I said first off we've got to share programs among the users. There is so much to write.

Hendrie: Yes.

Bell: And so we got DECUS going, probably my guess is in '61 or so. There weren't very many users at that point until after PDP-1. I guess we made a couple of these remote console ones, and then we streamlined it and put the console on the end like the PDP-1s we have kept... BBN was delivered the second one.

Hendrie: And that became more the production model.

Bell: And I think it was probably number three that was that way and I doubt there's anything left except the pictures of the early one.

Hendrie: Of the early one. Now, at the time that the PDP-1 was originally built there was supposed to be a PDP-2 and a PDP-3. Now was anything going on with that at the time?

Bell: The PDP-2 and 3 or the three's . . .

Hendrie: I actually have a brochure for the 1, 2 and 3.

Bell: Right.

Hendrie: That's how I knew that from my . . .

Bell: The PDP-2? I don't have the brochure on the 2.

Hendrie: Yeah, well this is just a combo brochure for the line.

Bell: Okay.

Hendrie: 1, 2, 3.

Bell: Okay. The two was left to be a 24-bit machine. The three was a 36-bit machine, and we had a proposal out and there was an architecture for it and it basically was kind of a big overgrown PDP-1 with index registers. And anyway what happened was we had this proposal and we sold one of these.

Hendrie: A PDP-3.

Bell: A PDP-3 to the Air Force Cambridge Research Lab, U.S. Hanscom field.

Hendrie: Oh, my goodness.

Bell: And I don't remember what year that was. It might have been . . . it was probably in 1962 or 1963, something like that.

Hendrie: So you did actively promote this.

Bell: Yeah.

Hendrie: But you didn't build one on spec.

Bell: Right, there was none built. It was all clearly vaporware.

Hendrie: Yeah, well that was the way people did it in those days and still do.

Bell: Yeah, here's this computer we have a dream about. Do you want to buy it?

Hendrie: So, did one get built?

Bell: No, but I have a wonderful story about how we dealt with the order we had for the 36-bit PDP-3.

Hendrie: Yeah, I want to hear the story.

Bell: Harlan Anderson and I went over to talk to the guy buying it, Charlton Walters, who was doing speech work research, and on the way over -- I remember the spot on Route 2, just as you make that corner that turns going back to Boston and you keep going to Bedford -- we had an epiphany. Harlan and I were over there and I was going to be the project engineer on it. I, or maybe Andy said, “Hey, the company is taking off. We have no business building another ~!@#$%^&*()_+ computer. We can't do this. This is a lot more work than we thought. It's this stuff called software that takes so much work.” We were just beginning to understand what a computer company looked like and what you had to do to be successful.

Hendrie: Yes, and the other stuff you have to do.

Bell: I can build this thing but, oh shit, what are we going to do?

Hendrie: All the other, we got to support and . . .

Bell: So, I don't know. I think Andy or I, one of us, reinvented. Well, we'll give them the 36-bit machine, but it will be in the form of two 18-bit machines.

Hendrie: Okay, oh wow.

Bell: So, we go over and tell him, "Charlton, here's the deal. We want to give you two machines. It will give you a lot more power and it will let you break up your analysis thing. It will provide your redundancy. We'll make the connection between them so they can be used together."

Hendrie: So they can talk to each other.

Bell: Yeah, and "We'll give you all of this capability." So, we made up a little nice story about the benefits and were able to convert the 36 bit PDP-3 order into two 18-bit PDP-1s.

Hendrie: With some special hardware so they could . . .

Bell: Yeah.

Hendrie: So they could be programmed to work together.

Bell: Much less hardware than we would have had to build otherwise.

Hendrie: Oh, my goodness.

Bell: That was the PDP-3.

Hendrie: That's as close as the PDP-3 ever got.

Bell: Well, not exactly, because some company, and I don't know remember the company's name, actually took the spec of the PDP-3 and they bought DEC modules and they built a PDP-3.

Hendrie: Oh, my goodness.

Bell: They didn't understand software either, or the curse of having a one-of-a-kind computer.

Hendrie: Oh, wow.

Bell: So, there was one PDP-3.

Hendrie: It existed.

Bell: One existed. I went over to see it.

Hendrie: Do you remember where it was?

Bell: Yeah, it was in Waltham along one of those streets parallel to 128th on the east.

Hendrie: Oh, okay.

Bell: I don't remember the company's name but I remember seeing it.

Hendrie: Okay, that's pretty interesting.

Bell: Right. But fortunately, we got a big order from IT&T that allowed us to continue and grow. And then, I don't recall when we made the deal, but it was sort of halfway through oh maybe the summer of . . . it would have been '61 or so because I know it was the new blue machine. We made a deal with ITT which was to build them a switching system called the ADX7300 and that was to replace a torn teletype tape switching system.

Hendrie: Oh, a torn tape teletype switch, ah.

Bell: Right. And interestingly enough I always wondered about if you ever ran two torn tape systems together, that is whether it totally spoke, whether just one torn tape system in the whole world or whether you actually had Europe and the U.S. in which case packet switching would have been invented. Thus it was invented many decades earlier than the fathers of the Internet ARPAnet claim.

Hendrie: Maybe you could explain what a torn tape system is. I happen to know but I don't think people reading this or watching this tape do.

Bell: Yeah, anyway, back in the days when people sent messages by teletype, you basically either talked on the teletype directly or you made a tape with the teletype. And the code had all evolved from

Morse code, but anyway there was a code on the tape and the code was in that case a five-bit thing called a Baudot code. Those five bits were used to encode 32 characters, and you had the upper case alphabet and some control characters and stuff like that. And they were read in parallel and then each character was what you had typed. And so you had a tape reader to enable you to send the messages stored on the tape. Instead of calling everyone directly, you put a message address on the tape that you first sent to New York or another switching center. The sent tapes were repunched at the tape reader. So I want to send a message from one place to another so I basically send it to a torn tape center. A new tape is created there. At this center someone read the destination address, took the tape to a reader at the center, and sent it off to the final address. This is similar to the way FEDEX works -- all messages are collected in trucks and flown to Memphis, are sorted in Memphis, and then flown back to their destination and delivered.

Hendrie: Now, this is a switching, this is really a switching system.

Bell: This is a switching system.

Hendrie: But manually the switching process is very manual.

Bell: Right, and basically on the header of the message you say, "Well, I want to go to somebody" and the way you did that was that a tape would go to the center and the tape would be labeled in some way where it's going and that that tape is for a type M machine and so on. So you'd basically just take the stack of tapes to a particular destination reader and you'd cue queue them on up for output.

Hendrie: And you'd send them out on a line going . . .

Bell: On another line.

Hendrie: On a reader. They come in and get punched.

Bell: They come in and are punched.

Hendrie: Are punched. Now you have the tapes and then you move it to somewhere.

Bell: And then you got the tape and then you move that to a station that's going to read and put it out on a telegraph some other place.

Hendrie: So, it's a one stage packet switch by definition?

Bell: Right, exactly. It is a packet because . . .

Hendrie: The packet is the tape.

Bell: That's right and I don't think there were any practical limits on how much tape you want in your message but these messages were short. These were telegraph like messages or Western Union kinds

of messages that you paid by the word. And so, basically ITT -- International Telephone and Telegraph is what it stood for at the time -- and John Ackley, who was an MIT classmate of mine, was the project engineer and was pushing this. He was in New York and we made a contract to basically build this switch so basically a tape could come in or the stuff would come in to the PDP-1, was stored, queued, and then sent out to the appropriate line when it was free. It is, of course, nothing now as that's what all of our hubs and routers do on the internet. We got rid of all the paper tapes and operators on roller skates moving tapes about in New York and other centers.

Bell: I was the project engineer for the ADX-7300 for IT&T. And what that entailed was basically handling the telegraph. Well, two things -- one, handling the telegraph lines, which was taking in the 5-bit characters and then which were transmitted serially and then assembled into a parallel, 5-bit parallel character, and then brought into the computer and stacked placed in a word. And so that involved two things: One is the receiver and the transmitter on one of these lines; and then the other part was once it got into the computer, then multiplexing those in separately and putting them into the machine. So you essentially have all of these 5-bit characters arriving, and that was just sort of a standard multiplexing to bring those in and there was a kind of a multiplexer there.

Hendrie: Did you assemble them into 18-bit word and then...

Bell: No.

Hendrie: So, they came in one 5-bit character per word initially?

Bell: And they were then assembled, program assembled.

Hendrie: Program assembled, yeah.

Bell: And because they were coming in at 75 or 100 bits per second, so you've got a lot of time.

Hendrie: Now, the PDP-1, what was its performance?

Bell: It was five microseconds cycle time, so instructions took either five or ten microseconds.

Hendrie: I just wanted to get the perspective there.

Bell: Right, that's the spec on the PDP-1. And then it had memory size, the basic size was 4K words, the starter memory was 4K and then it ultimately got up to I think 64K words. And what we had just developed had been, well, we called it a sequence break system which was actually a program that was an interrupt. So, at the time, the basic machine had only one level interrupt. So anything would happen on the outside. You'd go and take that, whatever it was, and then interrupt and put it into. . . and take care of it in some fashion. In the case of this, we had 256 lines coming in, so we had a bit of multiplexing to do to get the 256 lines to all come in there and put them in some kind of order. So one of the options was a 16-line interrupt. And so I extended it to be a 256-line interrupt, so there was essentially interrupts on interrupts.

Hendrie: Okay. Then you had to do its software. You had to deal with the...

Bell: Yeah. So an interrupt would occur and then you'd get a number and say, well, I think we went to 16 places and then within those 16 places...

Hendrie: Go 16?

****Bell:** We had four places that you could go. So we had a unique place to go once an interrupt occurred. And so there was that mod to extend that to a greater, bigger interrupt system. And then, the interesting part was...

Hendrie: And you did this mod in hardware?

Bell: Yeah, it was all hardware.

Hendrie: It was all hardware mod.

Bell: Yeah, and then you wrote the routines to handle whatever is going to be handlable then. And then the other part was dealing with the telegraph line. And that was when, quote, I invented the UART.

Hendrie: Oh!

Bell: And the UART, people had been using single shots to that to deal with . . . I mean, that was the way it had been done. When we finally found out about them, people said, "Well, these don't work very well because you've got these analog potentiometers to set."

Hendrie: When you found out about the single-shot approach?

Bell: Yeah, right. And so I basically did a sample clock approach. And that was the method that is, in fact, in use. So you basically break the line up into a multiple of either 7 or 8 samples and then you take a sample from the time, you take the transition, and you go through a little state system to sample this serial telegraph line. And then you convert bit by bit, put that into a parallel buffer and then take it in. And in a telegraph line, as I recall, there were various codes used. But for whatever reason, there was a 7.42 baud, bit -- bauds were the actual bits on the line of which you got five good ones. And so one was the start character and then you had . . .

Hendrie: So there was a starting, a pulse of energy that was the start?

Bell: Six. Right. So you had 2.42 or others. And so we ended up with rounding that up to 7.5 as the signal. So basically you had a start 5 and then 1 and 1-1/2 baud at the end as a stop signal and then you started it again. And so in a sense it would kind of resync based on the next start bit.

Hendrie: Whenever the start came.

Bell: Whenever the start came.

Hendrie: And then it would just clock along.

Bell: Right.

Hendrie: You knew where the bit times were based on when the start was, and you'd sample through this and find out what the bits were.

Bell: Right.

Hendrie: Now, did you design a special module to do this? Or did you just combine a bunch of logic?

Bell: Initially, that was done with just logic.

Hendrie: That's a lot.

Bell: We had 50 modules which got us eight lines, 50-deck, 5000 series, that's a 500 kilohertz clock.

Hendrie: Modules.

Bell: Modules got us eight lines.

Hendrie: All right.

Bell: And then when the PDP-1 or PDP-5 came out, at some point I'd been arguing for a long time that this ought to be a module. In fact, of all the things I've kept as parts, mostly everything I've really given to the museum is because I just don't have enough space to put the stuff. And I do have some PDP-6 modules, and I also have a receiver and a transmitter. And these were extended module. So we got the idea of making these long modules.

Hendrie: Well, so they would be long in the sense they had multiple connectors on them or they stick way out?

Bell: No, they stick way out, because you didn't have a connector. You didn't have a pin problem, you had an area problem because they had 8 bits. They had 8 flip-flops and clocking and the counter and this other was maybe 15 bits or so of...

Hendrie: A lot of transistors of flip-flops on it?

Bell: Flip-flop. Yeah. And so that was put on an extended module. Ed DeCastro stuck that in the PDP-5. I think we made it for that just because we wanted to keep the size down, although I think by now it was like 6 or 8 DEC modules, a few standard modules. But it was still much more elegant just to plug

these in. And people loved them. You know, people who were building their own systems loved them. And that's sort of how I fell in love with the serial communication, because in the process of building and testing the lines and stuff like that, I wrote a test program which had basically simulated this message switching system. And we had a bunch of old telegraphs that IT&T had set up and so we were playing with these things all the time. And so I got to love these things because they were so reliable. This is in contrast to the Soroban, which was an IBM modified typewriter that was used on the PDP-1. But it was a full typewriter, beautiful typewriter. But it was constantly breaking down because it was being driven, you know, the computer was driving it faster than any human could ever drive this thing.

Hendrie: And it wasn't really designed for that?

Bell: It wasn't designed for that.

Hendrie: So you did not use Flexowriters, which was the other choice in that era?

Bell: We had never connected a Flexowriter -- we used Flexowriters for offline preparation. But Flexowriter, you know, again, was not a preferred...

Hendrie: I know. Well, your competitors used Flexowriters.

Bell: Yeah. And they were both...

Hendrie: They were both bad choices.

Bell: They were both bad. In fact, I think we connected the first, used the model 28, which was a 5-bit Teletype, probably the most homely thing you could imagine. We painted it blue and that was what I used on the PDP-4, because I wanted to get rid of the typewriter on the PDP-1.

Hendrie: With something more reliable.

Bell: And I think -- oh, geez, I can't remember -- I think actually we had lucked out at that. We lucked out on the PDP-5 just on the 5 transition that the teletype ASR-33 had come out. I'm quite sure that's what we used. And it was right at the cusp of when it was coming out, and that's what Ed put on the PDP-5.

Hendrie: And this is what you used?

Bell: And so it had the virtue of making the thing low cost. You got a tape reader. You got a tape punch. You got an I/O device. And so from a computer standpoint it's ideal. You've got the whole works right there in one little package. And it was pretty reliable. It wasn't as good as the older Teletypes, which were really meant for heavy duty use. But, in fact, it worked quite well. And that's the device that was used for a long time until actually that drove them out of business with the LA, with the Decwriter, which is a matrix printer.

Hendrie: All right. Let's roll back. Yeah, let's rollback to this project that you're working on.

Bell: Yeah. So fundamentally, the project was completed.

Hendrie: Now, how long did this take you to?

Bell: I don't know maybe six months or something like that. And meanwhile, you know, the company was developing. Half of the PDP-1s were sold to IT&T under the ADX label. And, in fact, it was one of the key things that I'd say "saved the company" and made it a computer company, because just getting enough orders there was the key thing. Up until that time, the PDP-1 brochure -- which, you know, we have copies of -- had every possible device that you could imagine in there. And we ended up selling one of each one. And Lawrence Livermore bought this for an offline device, essentially kind of a 1401 pre-processor for 7090. And it had the tape units, of course. And it had a Remington Rand tape unit, because nobody used Remington Rand. And so we had made a special controller for Remington Rand. It had a high precision scope and a camera and this was a 4000 x 4000 point plotter. Absolutely beautiful scope for film -- actually, we had a film reader on there -- and a scope would be used for reading and writing. It had a connection to -- which we didn't do -- but a connection to a high-speed card reader. And I can't remember who made that card reader. George Michels of Livermore arranged to buy it...

Hendrie: Being George Michael, yes?

Bell: I remembered seeing it on the top floor of where all the debugging was taking place. And this poor guy was there from the card reader company and debugging logic on the PDP-1, and things would go awry. So occasionally you'd hear this screech and you know the cards were all jammed into this one little space. And so we kind of renamed it the high-speed card folder.

Hendrie: So, basically, you had all the data processing?

Bell: Yeah.

Hendrie: You had to do all the standard data processing peripherals sooner or later?

Bell: Yeah, so Teletype was an important company because that interface there got us used to Teletype. And another thing that we removed was we had initially a thing called a Tally punch, which was the first punch of the PDP-1. And then again, it was an errant device. I mean, it was like, not a nice device. You know, mechanically it would get out of line. It was asynchronous. And so we were delighted when we could buy the Teletype BRPE punch, which is on the PDP-1. Probably still in op. I haven't talked to the PDP-1 restoring guys, but I think they got that working. They needed belts of course, but it was a beautiful punch.

Hendrie: Teletype made remarkable electromechanical equipment that was really reliable and...

Bell: Yeah, it absolutely worked.

Hendrie: They had great mechanical engineers. So, you've gone and you've finished this switch for ITT. Ironically, DEC’s matrix printer, the LA36, wiped them out though.

Bell: And then that's about the time we met when I started working on the PDP-4. And I think you triggered the PDP-4, which was to say, "We want a control device." You were working for Foxboro at the time, and you wanted a control device for a process control computer. And we said, "Yeah, we can make one of those." And I became the project engineer of the PDP-4.

Hendrie: All right. So that was the next thing you did?

Bell: Yeah, that was exactly the next thing I did.

Hendrie: And who decided what the PDP-4 was going to be like? Is this Andy?

Bell: No, I did. I don't know how the decision was made, but among other things that went into it was that we had gone from 5 Megahertz to 500 kilohertz modules. And 5 kilohertz were good enough to make the computers, so we didn't need these faster modules. And then also they were lower cost instead of transistor logic, it was transistor capacitor logic. And so, you operated at a lower speed but at considerably lower cost for the modules given the cost of the transistors at the time. And for whatever reason I decided that it wasn't going to be PDP-1 compatible. This was before I understood about programming and programming cost and investment. Which, by the way, there are still people who don't understand that. That is probably the most misunderstood concept in computing and in digital computing today. People just don't understand the value of software, the integrating effect of software, and why you, whenever possible, use somebody else's interfaces where there's an installed, or where there's a collection of software that you have access to or that you can get. Changing the instruction, changing the architecture is always tempting for whatever reason, and it's virtually always a bad idea. And in this case it was a bad idea. And there are lots of reasons why it was, "Oh, well, I hate ones complement arithmetic." It turns out that in PDP-4, it was going to be twos complement arithmetic. So floating point works a lot better and stuff like that.

Hendrie: Okay. So you made that change.

Bell: Yeah, I made that change.

Hendrie: From ones complement to twos complement.

Bell: I had made these this idea for very inexpensive index registers, called auto index registers, which you could indirect to memory and bump up the count. And so you basically can index through something in not too bad a shape. And so you save an index register. So I didn't need an index register and all the attendant logic.

Hendrie: Was there an index register in the PDP-1?

Bell: No. No.

Hendrie: No. There was not an index register, so this got over the marketing problem.

Bell: Yeah.

Hendrie: Your competitors had index registers.

Bell: Now we've got to put index registers in. And then there were a few other things that were "better" in terms of how it worked. I think one of the things was I gained a bit with all this fooling around and so it could directly address more memory.

Hendrie: Okay. Because you had shorter instructions, fewer instructions, plus you could encode them into -- because you were not compatible -- so you could fit them into a smaller number of bits and gives you another factor of two more memory.

Bell: But you still had to use bank switching to go beyond the 8K. And, we thought, oh, well, nobody would want more than 8K of memory at this point. And so, the first application, we can put in a bank switching on it. But anyway, I don't know how long it took me to realize this one point about the value of an installed base of software.

Hendrie: I don't think anybody understood it at that period.

Bell: Well, one of the quotes that Wilkes made, and I don't know when I heard him make it but I thought it was very early, he said “It wasn't very long before we began to realize the value of the software that we had created would vastly outweigh the cost of the machine”. And, you know, it's one of those things that's absolutely true. I was with a group a couple of days ago, and Chuck Thacker and I were there advising them about stuff. And then it was like, okay, we're going to use one of the CE versions for font. And they said, "Well, we just want to make this little change." And Chuck said, "Do not touch that. You cannot deviate from what Microsoft expects as a platform, because you cannot afford to do that." We've got, you know, there's one other very large PC vendor. He said, "I will not recommend their stuff because they go off and they make something that's 'better' and it turns out they can't track the software and can't support it and all, and they never get it right. And this match between hardware and software is one that you . . . there's just no tolerance for ambiguity there." You know, I think, probably kind of at the root of maybe the Jobs decision to use Intel. Well, they used plain old PCs.

Hendrie: I want to go back to one other. I didn't quite understand the difference of the upgrade of the modules and the memory or what trend changed there between the PDP-1.

Bell: The thing that made the sort of the rationale for the PDP-4 was that we could, in fact, make a machine that was half the cost of the PDP-1. So that was the rationale.

Hendrie: And so what modules did you use?

Bell: And the 500 kilohertz modules had just come out, which increased the density and reduced cost.

Hendrie: Oh, they were denser also?

Bell: They were somewhat more dense. And I don't remember where all of it came from. But in fact, the bottom line was that I was able to get the PDP-1, PDP-4, I think in something like one bay of seven or eight "crates of 25." I believe they were 25 across, versus three bays that I think were like 12 high.

Hendrie: And 25 across?

Bell: Yeah.

Hendrie: Still the same?

Bell: So it was a huge . . .

Hendrie: Reduction in the card count.

Bell: Reduction in the card count. And I think...

Hendrie: Were they 1 Megahertz on the PDP-1?

Bell: Five Megahertz.

Hendrie: Five Megahertz and you went down to 500 kilohertz modules?

Bell: Right, and then at a clock speed increase of -- again, something you don't do -- the cycle time went from 5 microseconds to 8 microseconds. Not a good idea. Kind of every time DEC came out with something that was worse and cheaper, it didn't really work out very well. Cheaper and faster's always good, but not . . .

Hendrie: But cheaper and the same is the furthest you dare go. You never want to go with cheaper and slower.

Bell: But anyway, the PDP-4 turned out to be the resulting architecture that went into the 4, the 7, the 9, and the 15.

Hendrie: Yeah. It had a long life.

Bell: It had a long life. But nevertheless, it probably should never have been. So what you have in this interview is a confession of my screw up. And if I'd just implemented the PDP-1 in these 500 kilohertz modules and probably stuck in some more instructions, you know, we probably would have done just as well.

Hendrie: But that's what you did.

Bell: That's what I did. And there were a bunch of ideas in there. The auto index registers and then the way subroutine calling was done. There was a trap that you could define, user defined instructions. And that idea basically came from Atlas, called Extra Codes, so that you could go in and trap some place and create another op code. This came from Atlas and we went on to use it in the PDP-6.

Hendrie: And you had in mind that maybe this would be useful in doing some of these special applications like building things like the torn tape system or something you didn't even know what it was.

Bell: Right. And that those were ways of call. You know, I'm having trouble remembering exactly how that manifests itself in the software. But that was a key way that we ended up doing "defining" more op codes.

Hendrie: Did you make any improvements in the interrupt system from what you learned?

Bell: I'd say I simplified it quite a bit. And then the other thing that was put in there was the ability to go in and index register, index memory so that there was certain process control, you would have an interrupt and it would go and you could dispatch to a certain place and then auto index that for doing histograms. And that ended up to be very valuable for the physicists when they were making counters.

Hendrie: Or pulse-height analyzers?

Bell: And the other thing was that direct memory access control logic was added. I'm pretty sure it was done there, but I don't remember when. There's the 4, 5, and 6, and there were a collection of ideas in the I/O system that all came out of that.

Hendrie: Exactly when you put DMA into the...

Bell: Well, DMA was in the PDP-1, but that was the device specified the register, specified where it wanted to go and the data. And in the 4, I can't remember whether on the 4 and the 5 exactly how those were done. But the 6 I know was done so that all the device state was in memory, so that basically a device would say, I've got a word to go in memory and it would go to a particular place and it would increment -- it would decrement the count and increment the address. And it would basically put it into, with two cycles, you could go into memory. And so I think there may have been three cycles. You do the same thing in a word count and a pointer in PDP-4. So the purpose of the control logic was to eliminate the need for block control in the i/o controllers.

Hendrie: Okay. And so you've got an interrupt.

Bell: I think maybe the 5 worked that way.

Hendrie: And then the computer set up where to go?

Bell: Set it up so you could make very low cost, fast block transfer access….

Hendrie: Devices.

Bell: Devices. Yeah. Because they were built to have lots of devices and to be able to handle them nicely. So those were all I/O refinements, interrupt refinements that made DEC a very desirable machine for connecting stuff. And then I think the key thing on the 5 was realizing that these should have been

busses, and making the 5 be a bus architecture for I/O. So that made it still easier to get stuff. And that was kind of an I/O bus and then there was a memory bus, because otherwise on the 4, where you had the commit space multiplexers, in fact, we had a very elegant way of putting stuff in and patch paneling it in. But it was still fanning in devices rather than having a bus where you basically paid nothing for the I/O until you connected to it

Hendrie: For the multiplex. It was a distributed bus, a distributed multiplexing scheme is what it fundamentally is.

Bell: Right. And so that idea was used in the 5, it was also used in the 6. The 1 and 4 had been radial.

Hendrie: Okay. Good.

Bell: And then, in fact, I extended that notion. I think I had two big ideas -- the other one was the Unibus. And the Unibus was extending the bus even further, so that you have everything on a single bus. And that was, by the way, invented at Carnegie.

Hendrie: Oh, really?

Bell: Right. The interesting thing is, when you invent something, you can remember exactly the day and the circumstances of . . .

Hendrie: Of where you were.

Bell: You know, I was talking to Ted Codd of IBM who invented the database. And he was our contract monitor. He was monitoring an IBM research contract at Carnegie. And he was asking me what I was doing. And I was writing the book with Allen Newell that we called Computer Structures.

Hendrie: Yeah.

Bell: And so I described sort of this general structure of, okay, here's the selected set of switches that are linking memory to processor to I/O to this periphery. And so you had this sort of fanning out, and then I said, "Well, gee, a more general structure is, I just have switch and everything connects into that, and then you can have this bus that…" you know, and I was sort of explaining it to him.

Hendrie: And it just came to you by your explaining it to him?

Bell: It just came. Yeah.

Hendrie: I think when you explain to people like that, you try to figure out, you're trying to explain the fundamental of what's going on and in trying to do it, you suddenly get the insight.

Bell: Exactly. And the general registers for PDP-11 came out of this kind of explanation at Carnegie. I was talking to Alan Perlis and talking about the use of registers and the next registers and accumulators.

Everybody was oohing and ahhing about the B5500, B5000, and I said, "Oh, you know in the 6 we were able to create a stack as part of the general register structures." I was trying to remember just how far we had gone there. We had not really gone to a stack or rather put the registers in a stack. We'd stopped just short of that. And in the case of describing that to Alan Perlis at lunch, I said, you know, "They should do everything. You should have the program counters, you should have the accumulators, you should have the index registers, the stack pointer, everything, and its how you control them."

Hendrie: Determines their use.

Bell: Determines their use, not having special ones that you can't make fungible. And again, then I worked with a student, Harold McFarland, who ended up doing . . . You know, he's one of these students that were basically there and sat in class and was designing computers all the time. And so, I worked with him, and then he took a job at DEC one summer. But the PDP-11 is another story. But I think the important next story probably is the PDP-5 or maybe finish the 4.

Hendrie: Yeah, why don't you finish the 4.

Bell: The 4 was designed for process control, and I think mainly it was with you and one was for a large Nabisco bakery in Chicago.

Hendrie: Yeah. I don't think I was yet at Foxboro, because I remember I got assigned to decide what computer we were going to use. Foxboro had not acquired the RCA Computer Group yet when the 4 was started. So the 4 was in the works and you had maybe one customer. And well, I think we should probably change the tape before we get into that one.

Bell: Okay, sure.

Bell: Right. During the interim we had talked about another thing, the Foxboro decision to use the PDP-4. And Ken's reluctance to put a drum on it probably stemmed from the fact that we had put a drum on the PDP-1 for BBN. Anything that had a drum on it was probably suspect. But we had made and, in fact, I did the original design of this magnificent drum for a PDP-1 that was a swapper so that in one cycle you could read a track and write the track into the memory and write memory onto another track simultaneously. This was by no means a trivial feat just because you had to synchronize the core with the slots on the drum and the slots or words had to be swapped.

Hendrie: Did you actually do it with one memory access for each word?

Bell: Yes, right. You would read out one word, put it on the drum, and then write back another word from another track on the drum.

Hendrie: A different word.

Bell: Yes.

Hendrie: So it was really efficient in terms of memory.

Bell: Oh, yeah. You could do a swap in one rotation.

Hendrie: It was literally a swap. It wasn't like you'll read the whole memory out and then you'll turn around and write it back.

Bell: Right, because this was for timesharing. You basically wanted to be able to swap an entire user's 4KW working memory in 1/30th of a second and have another user in place. You could move somebody out and somebody else in and . . .

Bell: And you did that for BBN.

Bell: Yeah.

Hendrie: Who at BBN was writing a timesharing system?

Bell: This was right at the beginning and in fact it was one of the . . . there's a whole story, the whole collection of stuff around the PDP-1 and around timesharing at that point because timesharing was just hitting in those early '60s. MITs' CTSS that Corbato designed for the IBM 7090 had started to work. We had the swapping drum that we had created for BBN and then it was also delivered to MIT.

Hendrie: Jack Dennis, did he buy one too?

Bell: He bought one.

Hendrie: And he did timesharing.

Bell: Yeah, and that was also . . .

Hendrie: The same drum.

Bell: And so at that point we sort of said this is hard to do because you're trying to read and write, you've got a lot of heavy currents going on everywhere for reading and writing, and so it was a high noise environment.

Hendrie: So it was hard to make the equipment work.

Bell: Right, and so we said okay, if you want just a plain, old drum we can do plain, old drum, and it was refreshing just to get a drum that you read or write to but didn't have to do swapping because swapping added a lot more registers because you had two parallel channels going -- just read or wrote, but not simultaneously.

Hendrie: Exactly. Simultaneous, synchronized, yeah.

Bell: So the drum really was part of the whole timesharing experience to make memory swaps fast. In fact, those drums were the device of choice when you're making timesharing systems.

Hendrie: So that was a . . .

Bell: The SDS 940. You used a drum and . . .

Hendrie: I think everybody did because this was prior to widespread disks.

Bell: Yeah, and that was . . .

Hendrie: That's what there was.

Bell: Yeah, and I think the original CTSS [compatible time-sharing system] used a drum and they swapped 32 k words on a 7090.

Hendrie: So that accounted for Ken's drum reluctance to put a drum . . .

Bell: **"**I don't want another drum," yeah.

Hendrie: There were a lot of headaches with that. I didn't realize you'd done the swapping drum, that that ever came up as one of your projects.

Bell: Right. Yeah, I'd forgotten about it too. It was part of . . .

Hendrie: The stuff you did during the PDP-1, time you were working on the PDP-1.

Bell: Right. And the other thing was that then BBN bought a large PDP-1 for doing hospital control. Jordan Baruch was making a real time hospital control system, medical control system, and in that case we had a drum swapper on it, and we also had disks because it needed a huge data base, he needed a data base there. This was in 1963, '64 period I believe because the PDP-6 had just started and the memory system was built to test the PDP-6 memory modules.

Hendrie: But he was doing it on a PDP-1.

Bell: He was doing it on a PDP-1, and in a way we debugged some of the memory bus stuff on this machine because it was doing high speed I/O into the memory while we were also running a processor in the other memory bank. And that was part of the PDP-6 -- it had that multiplex parallel memory structure.

Hendrie: So you had independent memory banks and you could be DMA or doing some controlled I/O while you were running code out of the other one.

Bell: And that was kind of where we first used that before it came out on the 6. Let's see, back to the 4 . . . But, I think we may have finished the 4, so ultimately it got out to Foxboro. Jay Forrester was on the board, and board concern at the time was about whether we are liable if this computer stops and dumps flour into the Chicago River. It was at the Nabisco baking plant.

Hendrie: Yes. That was one of our customers at Foxboro.

Bell: There were two that I think were kind of targets, that we wanted to make sure we could do. It may have been a power company. Now I recall the other one was Corning Glass.

Hendrie: Yes. We did some power companies and I remember Nabisco also.

Bell: And then . . .

Hendrie: The board did not realize that in some sense none of these were direct digital control. These were always just setting set points on analog controllers so there was almost always, if there could be, big trouble in River City. There was some backup control system in the analog domain -- classically it's the catalytic cracker which can explode under certain conditions, but there's a slide valve that comes down and breaks up two parts of the reaction, keeps the catalyst away from the petrochemicals and the most known number of refinery mega explosions have occurred when the slide valve stuck. You would never have a computer control the slide valve.

Bell: I was trying to think of any more on the 4's. And then I think the interesting part of the 4 was when we were invited to go to Chalk River Canada to bid on a reactor control computer. Ed DeCastro was in special systems at the time, and I don't think anybody else went to Chalk River and talked to them about the control, about what they needed in this system. Chalk River was a system that we sold. We sold a PDP-1 there for pulse height analyzers and counters and stuff like that. Ed was there to say okay, what is it you need, how many scalers, what, how many x to do the special front end system design.

Hendrie: Yeah. They built a whole bunch of special front end equipment.

Bell: So there was a fair amount of special hardware, and the spec was something like, well, we're assuming the PDP-4 was going to fail on this I/O and so this other part can't and it's got to be maintaining a certain state. So on the way back I said let's build a tiny, tiny, tiny computer, a 9-bit computer, that's going to just maintain the state, just have the program that's going to do all of this stuff and then feed it over to the 4 and vice versa for control.

Hendrie: Rather than doing it with hard wired logic which was the . . .

Bell: The normal way you did the job.

Hendrie: Yeah, special systems had done memory testers and testing systems like that.

Bell: Oh, yeah, and we had very complex systems designed . . .

Hendrie: . . . with hard wired logic. DEC had a lot of experience in doing complex hard wired logic.

Bell: That was the way you designed digital systems before we realized the general purpose computer can do everything. The computer industry got started that way simply because the computer companies would go off and make the integration before there were sort of system integrators that would do that same function. So we were driving back that evening to the airport and it was, I remember, very, very cold and we were sort of sitting in the car talking about this, and I said this ought to be a computer and we got back and started the design. And then as it got evolving, it took on this character and went from 9 to 12 bits and in the process even stopped at 10 bits.

Hendrie: Do you remember why it did that?

Bell: No. It was like, well, it's only 30% more cost and we get so much more out of it than just having it be barely able to do it the job. We feared we might have run out of memory or something.

Hendrie: It had a built in A to D, didn't it?

Bell: That's right. This may have been the reason we went from the 9 to 12 bits. The way we got the cost down was it had a built-in A to D using a combination of hardware and software. The accumulator there had a D to A attached to it and a comparator to incoming analog signals. So with the program you basically went off and successfully used that to generate the signal and did a compare and, using various forms of trial and error, determined the input signal voltage. So it was very inexpensive and in fact we did everything to take cost out of the design. The program counter was in memory and so it had just one accumulator. That was the whole state of the machine. We used some of the ideas that were in the 4 in there, like auto index registers in memory. Also, it used a bus for easy I/O connection, unlike its predecessors. And it used 2's compliment arithmetic that made multiple precision work better.

Hendrie: You'd already figured that out on the 4, the arithmetic. Did you know anything about the CDC 160 at this point?

Bell: Yes. The CDC 160 was a 1960 machine so we had that to look at. I'm pretty sure we had the LINC to look at. So we had a bunch of computers.

Hendrie: Lots of places to steal . . .

Bell: Yeah, exactly, and then in the case of the 160 the big decision really was rather than having an address that was relative off of the program counter for short addresses or to have them in banks, and so we went to a bank structure. I think there were 128 word pages. So in a funny way they were very small, little pages that you flipped in and out, and I still don't know whether that was a good decision or not overall. I think overall it probably was. It gave you these places because later on there was a huge amount of swapping that went on in these small pages, and so they were used in a funny way as overlays which would have had to be done in the program and the boundaries may have been a good thing to save, but that was the key on the PDP-5. It was a nice way to think about programs.

Hendrie: That became the PDP-5. Wasn't it called the Program . . .

Bell: Data controller.

Hendrie: Something else, 12, the 4, before it became an official computer. I think I have a brochure for it being called a . . .

Bell: A controller.

Hendrie: A controller 12.

Bell: Right. We knew it was a computer. And then it was officially named the PDP- 5.

Hendrie: It came into the series of DEC computers.

Bell: Yeah, but it wasn't very long because in fact Ed had designed it as a computer and with all of that capability and then, I don't think he did it, but somebody else did the logic that had to do with the work at Chalk River. But essentially these are sort of two instances. The 4 and the 5 were really good instances of this. You're designing for a target customer, you know what it is you want to do, and they were good, important exercises to be able to do.

Hendrie: But then you tried to generalize it so you don't . . .

Bell: It's good to have them demanding in some regard so that you're able to cover the whole space. That gets up to the 4 and 5 and then now the 6. I'm not sure exactly when the 6 started. I've got my notebooks and I ought to go back and look at when the first . . .

Hendrie: When the first scribblings of 6 or what it looked like or . . .

Bell: Right. When or why that was done. I'm sure Andy triggered it in terms of "Gordon, go off and look at building a serious machine," and then it was a bigger machine, a machine designed for timesharing. So we'd been playing with timesharing and then the MIT CTSS system was up.

Hendrie: And people had been taking about them.

Bell: Yeah, and our early machines were in a sense way too small to be worth time . . .

Hendrie: Worth timesharing.

Bell: There wasn't much to share. They didn't have floating point and they had very little computation ability and then limited IL because of the memory structure. So the idea was to improve on all of those dimensions, so a lot more bandwidth, a lot more expandability, a lot bigger machine, a real computer that can handle numbers. That would have been probably in '63. I can probably pinpoint it to some extent because I remember working on the instruction set architecture and op codes at a conference on nuclear instrumentation at the Naval Postgraduate School in Monterey where we described the work that had been done in Chalk River. John Leng, who was ultimately the division head of PDP-10 and had started to

work for DEC, and I coauthored -- or rather he wrote it and I was a coauthor of -- a pulse-height analyzer for a PDP-1 that had been put in at Chalk River. It was one of the first papers on building pulse-height analyzer using computers. So I can find out when it was. The conference was in Monterey and at the naval postgraduate school and so I can find out what that date was and then I remember working on the architecture at that point in time and working through the op codes. In a way it's the final stages of the op codes at that point in terms of what was going to be done.

