Oral History of Harry Pyle

Interviewed by:
Gardner Hendrie

Recorded: May 27, 2005
Bellevue, Washington

CHM Reference number: X3200.2006

Hendrie: Harry Pyle has joined us for an oral history for the Computer History Museum's Oral History Project. Thank you very much, Harry, for doing this.

Pyle: My pleasure.

Hendrie: I think I'd like to start if you could give a little bit of your family background-- where you grew up, who was in your family; things like that?

Pyle: Okay. Sure, I grew up near Richmond, Virginia and when I think I was about nine years old we ended up moving to Western Maryland. And I had a sister who was a few years older than me, and I'd always been kind of interested in technical things. And my dad was a chemical engineer. He would take us over to the big plants where he would work and we would see all the machinery. <laughs> That was always a fascinating thing for me. When I moved to Western Maryland, there was a high school kid across the street who was a ham radio operator. And this was back about 1958, I think, and this was during a period of time when what's called the 'sunspot cycle' was very active and the ham radio operators could easily talk all over the planet. And this just really sparked my interest in communications and radio and things like that.

Hendrie: How old would you say you were?

Pyle: I was probably about 10 years old at that time. So that was, you know, the late 1950s. And then I was there in that area for maybe five years, and during that time I got really interested in the ham radio and started to get interested in electronics and started building little Heathkits and Knight-kits for those people who remember those kinds of things. And started to get a little bit interested in some digital aspects of electronics, which were just starting to happen-- this was in the early 1960s.

Hendrie: So, you got your ham license relatively early?

Pyle: Yes. It was 1963, I believe. I was 13 or 14 years old and the challenge there was to learn Morse code. Back in those days you had to learn Morse code to become a ham radio operator.

Hendrie: Right. It was the only way you talked to anybody, really.

Pyle: That's right. So, I was fortunate, that I also… My dad also knew a fellow at the company he worked at in Maryland, who was also an electrical engineer and a ham radio operator and he really helped me kind of learn some more things about electronics and kind of maintained my interest in finding out how all these electrical things worked.

Hendrie: And your dad bought you a ham-- some ham gear?

Pyle: Oh, yes. He definitely-- I had to work on him quite a bit but when I would finally figure out exactly what the right thing to get was, he would indeed pony up and get me the gear. It was always Heathkits-- I think I had one or two what was called Knight Kits-- Allied Radio built some equipment back then. But they were kits. I loved putting the kits together. In those days, they were mostly tubes and you had to wire all the individual components together and everything like that. So, that was lots of fun and that really-- that whole electronics background is kind of what really got me into the computer business. After that, I think I was 13, I moved to Wilmington, Delaware…

Hendrie: This was because your father changed jobs, yes.

Pyle: Yes. Well, he transferred. He worked for a big chemical company and they transferred…

Hendrie: What was the chemical company?

Pyle: It was Hercules Chemical Corporation, I think it was called. I guess it was originally Hercules Chemical Company or something like that and they changed the name. But, their home office was in Wilmington, along with DuPont and Atlas-- and those were all split-offs from DuPont. I think either before or after World War II, so they all had their home offices there in Wilmington, so that was a big center of activity for the large chemical companies. And once again, he found somebody that worked in the office who was a ham and was interested in electronics. It was actually after we moved there that I got my ham radio license and shortly after we moved to Wilmington. And I got into communications, the digital communications-- quasi-digital at the time. The hams had a thing called radio teletype. You could purchase a surplus teletype machine. They were truly mechanical marvels. <laughs> I mean, I wish I kind of had kept my hands on that piece of equipment, but they were kind of early mechanical computers in some sense. And they sparked my interest in communications and digital communications and kind of getting further into electronics that really had a foundation in the computer business. And it was through that type of communication thatI met up with a fellow named Vic Poor. And when I was a kid in high school, I bumped into him on the ham radio waves.

Hendrie: All right. Just tell me a little bit more about what you were-- when you were in high school. What sort of things did you do in high school, as opposed to this ham radio which was obviously an after-school, I assume.

Pyle: Right. Yes, I think our high school-- we never did actually get a club station actually running there at the high school, but I maintained my interest in computers-- I think I really got my first introduction to real computers pretty much in high school. One of the math teachers was somewhat progressive. He was a young fellow. We didn't-- in Wilmington, we didn't live too far from the University of Delaware and so it was maybe a 15 minute drive down to the University of Delaware and they had a computer facility there in the late 1960s. This was probably about 1966 or 1967. And they had an IBM 1604, I think was the machine the undergraduates got to use. And our math prof, as an after-school thing decided that he would get a computer club together. This was pretty early in the days of computers. We got a FORTRAN book and he figured out how we could get these mark-sense cards-- they were really 80 column Hollerith cards that you marked on and they had a reader that was supposed to read these things and generate the punch cards and the punch cards would then get fed into the computer. And so I remember that first session. We'd had a session or two where we learned a little bit about FORTRAN

and we learned how to write a very simple program. There must've been 20 of us. We were all filling out these cards and it just took forever to do this. And then the next week's meeting, he came back and said, "Well, the bad news is the card reader really didn't really work right so we're going to have to do this all over again!" And this turned off everybody but three of us that were kind of pals at the time and we kind of decided to take advantage of this and instead of doing the mark-sense cards, we just hopped in the car and went down to the University of Delaware. We kind of pretended like we were undergrads because we had an account at the computer. So, we went in there and we got sent out to the punch card machines and we punched up our programs and submitted them through the windows. Back in those days, most people, undergrads didn't get to actually touch a computer.

Hendrie: Yes.

Pyle: So, you would punch up your cards and you would pass them into the little window and come back the next day and your printout would be there. And if you were lucky, the program actually did something and if you weren't, you got to fix your program and submit it again and it would take another day. <laughter> It's not exactly highly interactive.

Hendrie: One day debugging cycle.

Pyle: That's right. So, that actually taught us a little bit about being careful about how we wrote our program and, you know, desk checking it and having our friends look at it too, to see if they could find any bugs before we turned it in, because it was so painful. But, that was painful enough that we finally decided we would just act like undergrads and since we had an account, we just signed up on the computer and we would sign up for an hour's worth of time, and we would go in actually run our own programs. And so, we ended up…

<camera adjustments>

Pyle: So, we ended up just kind of acting like we were college kids and would go in there once a week and we could actually fix our programs and dynamically debug them and do all these good things. 1604 was kind of a nice machine. The University of Delaware had an advanced version of it that actually had hard drives. Apparently hard drives were an option on this computer. And we were able to kind of run through our programs and all of this. And this went on for about a month until the math prof, or the teacher that we were working with in high school got the first bill. <laughs>

Hendrie: Oh, oh. <laughs>

Pyle: And all of sudden he realized that not only had we eaten up all of the account time we'd been allocated for the whole little after-school club-thing for the year, but actually kind of gone considerably far over that. So, that was kind of the end of our computer account <laughs> at the University of Delaware, but we learned a lot and it was kind of fun playing with the computers. They had an SDS machine there--Scientific Data Systems, I think was the name of the company.

Hendrie: Was it a 940?

Pyle: Yes, it was a nine-something.

Hendrie: Yes, they had a whole series of 900…

Pyle: So, whatever they had in the late 1960s, I guess-- we didn't get to use that machine, but that was one of the machines that was there. We learned a little bit about it and that was kind of interesting.

Hendrie: All right, very good. So in high school-- what's your first memory-- or maybe it was in grade school-- of what you thought you might want to do when you grew up?

Pyle: Oh, well-- I always kind of wanted to be-- very early on I had an interest in electric trains. So, you know, a lot of little kids have a lot of interest in electric trains. So, I of course thought I wanted to be a train engineer. <laughter> So for up until maybe I was eight or nine years old-- and I loved to play with the electrical part of the train, you know, wiring up the tracks and getting all the wiring right and having multiple sections so you can actually have more than one train running at a time, and making little switch circuits that could do things. And we eventually got some large pieces of plywood and would run wires underneath them and light up all the little lights and all these kinds of things. So, I think that early electric train stuff kind of got me into electricity. Another thing I remember doing when I was a kid was taking apart anything my father would let me take apart. So, if we had an old radio or an old piece of equipment of some kind, pretty soon, I was pretty much gutting the thing and pulling all the pieces out and trying to kind of figure out-- I didn't really know how any of that stuff worked, way back when I was young, but just taking the pieces apart and looking at all the bits and pieces. I think that's-- and then as I as I got more into ham radio and started building kits and things like that…

Hendrie: Then you started to understand what the parts were and how they went together and…

Pyle: That's right, that's right. And then that slowly evolved into an interest in computers because I think the communications aspect of the ham radio that I was getting into when I was in high school, kind of started me in the direction of logic and things like that. So, I always had an interest in those kinds of things when I was little and I think I always kind of knew that I wanted to be a, I think, pretty much an electrical engineer as I kind of matured. The computer business was young enough at the time that I didn't really understand that computers themselves and programming the computers were going to become an interesting career, until pretty much I got into college. Because I went into college, you know, essentially going into the electrical engineering program, but I wasn't there for long before I discovered the computer center and all the more advanced things that computers could do. And there was a group of us that really kind of started getting inside the computer center, rather than just being a student who ran a program on the computer. One of the fellows I was a roommate with-- this was at Case Western when I went to college, this guy named John Walker who actually was one of the founders of AutoDesk. And he wrote a lot of AutoCAD from what I understand and he lives in Switzerland. He's been living in Switzerland for some time and he's got a bit of history Web page he's put together some of the early UNIVAC computers we used at college and those kinds of things.

Hendrie: Well, let's go back and just talk about when it was time to go to college. What were your thoughts? Where did you apply? What did you think of doing?

Pyle: Well, I was very interested in electrical engineering and there was the University of Delaware nearby, and they had a pretty good engineering department. We went out and did a road trip, of course, and looked at Purdue. And the Case Western thing was-- it was actually Case Institute of Technology when I first went there. It kind of morphed into Case Western Reserve when Case & Western Reserve, which were right next to it, kind of decided for administrative reasons it would be better if they were a single institution. And one of my-- my sister had a friend and she had an older sister-- this was one of those things-- totally kind of random things. She had an older sister who was getting married and the best man at the wedding that summer turned out to be a prof at Case Institute and he was an electrical engineering prof and we got to chatting at the wedding reception and pretty soon he's telling me how great this place is. And so I got pretty interested in it and that was one of the places I went on the road trip. And I got really interested in going to that school, so I applied there. I think I applied at Purdue, but-- I did an early application to this school and I got accepted. So, at that point I said, "Well, this is really the place I want to go." So, I don't really remember all the other places I applied, but since that was an early deal and I really, I got accepted, and that was really the place I wanted to go.

Hendrie: It was the one you chose to do an early application. Okay, all right.

Pyle: That's right. So, that's what got me in there and I'm really glad I ended up there because they-- Case was one of the early innovators in computers. They had some of the early IBM drum machines. And they'd been doing-- as part of the electrical engineering department they'd be modifying these machines. And they had gotten a UNIVAC 1106 with-- on a $1 a year lease from UNIVAC with the agreement, I believe-- I don't know all the details, but that anything they did and improved this machine with, would become property of UNIVAC. And one of the things they did was they added protected mode multitasking in the 1106 and UNIVAC turned that into an 1107. And that was one of the early machines that could do-- run user programs in a protected environment that kind of protected the operating system from it. So, they were doing a lot of work, actually messing with the operating system and they had graduate students in there doing this work. And a number of the profs that were involved in the computer science there at Case, ended up going out to Xerox PARC and there's a fair amount of leadership in the computer science development world there at Case, in those early days.

Hendrie: It was a very good school.

Pyle: Yes. And it was kind of the leading edge, in some sense.

Hendrie: Yes. Okay. Good. Let's roll back to your first association with Vic Poor.

Pyle: Okay.

Hendrie: Okay. Now, you're still in high school?

Pyle: Yes, I was in high school, probably 10th, maybe 10th grade or 11th grade, I don't remember exactly when. And I had gotten into this ham radio communication mode that used teletype machines. And the teletype machines basically connect to a piece of gear you had to build pretty much yourself, that was effectively a modem for high frequency communications for shortwave radio. So it kind of interfaced the teletype machine into the ham radio circuits. And we started communicating on that and I bumped into a group of guys that were pretty much the leaders in developing this. One of the guys had actually written an article in a magazine called 'QST' which is a main ham radio magazine-- ARRL-- this main ham radio organization in the United States. And he'd written an article about how to build some of this equipment and so I kind of got to talking to him fairly off and on, on the airways with it. And I managed to get a few pieces of equipment, some of which were kind of mechanical computers. They had these things called selectors. I've forgotten exactly what they're called, but basically they could decode a character sequence and determine that a certain sequence of characters that occurred and therefore, flip a relay. So, these were primarily used to turn on and turn off the printout for a selected message, so you wouldn't have to waste an infinite amount of paper printing everything, if you were only interested in certain messages. So they had these interesting boxes, they were about this big with a lot of levers, and they were essentially teletype receivers that didn't print anything. They could walk through a sequence of characters and have this mechanical latching relay system work that would detect if a complete sequence had been completed and then that would flip a relay on. Or then you usually had another sequence that turned off.

Hendrie: Yes.

Pyle: And so you could make your machine run 24 hours a day, but only print out the messages that were actually directed to you. So, this kind of got it-- and this began to be a kind of quasi-computer-like setup.

Hendrie: Is this something that somebody showed you or did somebody suggest this?

Pyle: It was a piece of surplus equipment.

Hendrie: Surplus?

Pyle: So the wire services had been using this for some time.

Hendrie: Ah, okay.

Pyle: So that the teletype machines would only print out articles that were of interest to a particular wire service-- they had a common circuit. All these machines would monitor this all the time, but then they'd only print out-- they also built this mechanism into some of the later generation of teletype machines, themselves.

Hendrie: Okay, all right.

Pyle: So, anyway, I was on the air and I was communicating and Vic Poor was one of the hams who was kind of involved in this. And I got to communicating with him a fair amount. And he lived only maybe 150 miles away, over in Maryland. I was in Delaware at the time. And I got to the point where-- I can't remember the first time I actually visited over there, because I know one of the projects that happened was that his mother-- I don't know if he told you about this…

Hendrie: No, I don't think I heard this.

Pyle: His mother was involved in the Wycliffe Bible Translators who was an organization that would print up bibles and take them down to South America. It was kind of an evangelical kind of organization, from what I understand. And their primary mission was to get bibles essentially distributed out into the hinterlands. And they worked with a little outfit called the Jungle Aviation and Radio Service, which basically supplied the air transportation and the communication services for these folks. And so these folks were actually using ham radio to do some of their communication. It was an all-volunteer thing. There was no commercial aspect to this because one of the things about amateur radio is that you can't-- there can't be anything commercial involved with any of the services you provide. And they were needing to get some teletype communications setup and so we were-- Vic, I guess, was going to volunteer to build some units, some of these modem units that would allow the telecommunication gear to hook to the ham radio rigs. And he needed a prototype built, so I volunteered, I love building stuff, so I volunteered to build him one of these things. And I built one up and hauled it over to Frederick, Maryland to show him what it was. And of course I'd done a really careful job of laying out all the wires just trying all those things.

Hendrie: Yes, of course.

Pyle: So I guess that was my interview loop with Vic Poor. <laughter>

Hendrie: You didn't know what it was…

Pyle: So then after that, I think that following summer I got a job over there, working for him.

Hendrie: Yes, in his little company.

Pyle: Yes, Frederick Electronics, it was called. And that company was in the business of building this kind of stuff, essentially for the government agencies. But it was expensive, it was all solid-state, it was not something that a normal volunteer organization was really prepared to pay for at the time.

Hendrie: So this was about maybe when you were in 10th grade, something like that?

Pyle: Well, that actually happened I think when I was-- that particular sequence happened when I was in 12th grade.

Hendrie: Ah, okay.

Pyle: I had been communicating with him some year or two before that. It's a little fuzzy now.

Hendrie: This particular project came up in that period and you volunteered. All right. Good. Now, there's a story that Vic tells about when you're-- it isn't clear to me whether you were still in college or not when he was starting to work on the idea of doing a programmable terminal and putting a little computer in it, that-- to take the product that was currently being done by the company in Texas, that was just an emulation of one, making a general purpose emulator. Now is that-- when did that happen?

Pyle: I was in college.

Hendrie: You were in college, then.

Pyle: I had been working summers for him. So, after that thing with the modem, I think after my senior year, that's when I got my first summer job. And so I went down there and I was working with him. And we actually had an Interdata computer there. I don't know if you remember those things, but they were-the Interdata 4 was a pretty good size rack mounted unit.

Hendrie: Yes, I remember them.

Pyle: It was the one with the little wires to program the ROM so you could actually reprogram the ROM by running the wires through the cores, slightly differently. And we were using that as a telecommunications computer at Frederick Electronics. Then when he made this move down to the Texas company, which was during my tenure at college, he said, "Hey, why don't you come down to Texas and work for me, down there, during the summer?" And after I went back to college, after the summer was over, I actually got a GE timesharing teletype machine terminal so I could call into the GE timesharing system. And we did a lot of initial programming BASIC-- kind of all they had on the GE timesharing systems at the time. And we did some simulation work written in BASIC. We did some early assemblers that when you actually run this BASIC program, feed your assembling language into it, it would spit out a hex file which you can then load into the microprocessor to run.

Hendrie: So, it was an assembler written in BASIC? Fairly simple. <laughter>

Pyle: Exactly. And nothing terribly fancy either but-- so, I wrote that stuff. So, I tended to do that while I was at college, now, so now I was picking up-- doing some work on the side. He was paying me for it, so it was helping me get through college at the time. And that's when we really started working on this machine. The germination of the terminal which kind of ended up evolving into the 8008, or driving the 8008 for Computer Terminal, actually I'm pretty sure it was like Christmas time of 1969. We were still in-- he was still in Frederick Maryland and I went there for the Christmas time break and spent some time with him and he was starting to think at the time, about some kind of a programmable machine because Computer Terminal- I don't remember exactly when Computer Terminal started, but they had started at

least a few years before this. And they were building a glass teletype, we used to call them back then. Hardcoded logic, there was a different logic board for each one of the different protocols the mainframe manufacturers were using. And Vic was starting to get this idea of building a programmable unit and we actually had a conversation-- I remember sitting in his living room playing with his cat, or something like that. And we were talking about some kind of a programmable machine and what kind of an instruction set it might have. And it would be very simple, you know, I'd been used to working on mainframes by now. And you know, UNIVAC 1108s and big, 36-bit machines and all this stuff, so we're trying to sit there and think. I worked on a few minicomputers. There was a-- I'm sorry, the name of the company has slipped by me, now. One of the early 16-bit minicomputers.

Hendrie: Was it one of the machines from Computer Control like the 116 or the 516?

Pyle: It might've been. Yes…

Hendrie: DDP 116, DDP 516 were very early-16-bit and 316 that was a family…

Pyle: I remember the machine was about this big of a rack, so it was not a small machine.

Hendrie: And then of course there were all the Interdata ones, too.

Pyle: We had Interdatas. They were using Interdatas as communication controllers hooked to the 1108 at college, at Case, but this was yet, another machine. It might've been a DDP 116. That number seems to ring a little bit of a bell with me. But in any case, we were getting-- so I'd had some exposure to more limited machines and of course, the other thing that happened to me at college, was that I kind of got really interested in computers. So, I started taking essentially on the side, all the graduate level computer science courses because at Case at the time, computer science was not an undergraduate degree. You kind of had to get either a physics or electrical engineering degree, or maybe a math degree, then to get computer science courses, that was really a graduate…

Hendrie: They were basically graduate level. And then you can specialize in computer science…

Pyle: Right.

Hendrie: …and programming.

Pyle: So I kind of cheated the system and I went and started taking all these courses and having a great time doing it, learning a fair amount about computers. And so I'd had an exposure to some of the ideas and I'd always been kind of fascinated by the idea of designing instruction sets and machines and the logic units and all those things. It was stuff you did in electrical engineering, too, you know, the logic courses and things like that. So, I remember that time-- no, I take it back. It was Thanksgiving of 1969. And I was on Thanksgiving break and I went to Frederick for that break instead of going home. It wasn't that far-- Case was in Cleveland and Frederick, Maryland is not that far away. It's a pretty short ride. I

remember spending that break with Vic, kind of brainstorming about this programmable machine that we would be able to use as-- so we wouldn't have to keep redesigning all this hard coded logic all the time.

Hendrie: Yes, every time a protocol changed or you wanted to expand the market…

Pyle: That's right.

Hendrie: to a new machine or to a new company…

Pyle: But our mindset-- this is a fascinating thing to me about how technology evolves. We were going this way and the world ended up going this way. Our minds were fairly narrowly focused on the problem we were trying to solve, and that was communication controllers and connecting glass teletypes to mainframe computers using their protocols. We had no thought in our mind-- at least I certainly didn't-- and there wasn't a lot of thinking going on about building a general purpose computer. We were trying to build a programmable communications controller so that we could just slip a new tape or some kind of media into the thing and bingo, it would turn into a different kind of machine. But, we took a general purpose computing approach to doing that. Part of it was driven by the fact-- we had some experience with the Interdata-- using that as a communication controller. So, our minds were kind of coming out of that world…

Hendrie: And rewiring the ROM and…

Pyle: That's right. <laughter>

Hendrie: …all of those things, yes.Gets you sort of thinking about what are the possibilities?

Pyle: Right. But it was clear that our minds were kind of going down this track of a communications controller, not a general purpose computer. The general purpose computers were still the big things back up in the clouds somewhere. And so I remember doodling out some instruction set stuff and Vic had some ideas. And I tried to generalize it a little bit. And I said we ought to have a call stack so that we didn't have to do-- was it the DDP 116, I think when you did a subroutine call, it would actually store the return address at the beginning…

Hendrie: Yes.

Pyle: …the location you pointed to and jumped to the following location or something like that?

Hendrie: Exactly.

Pyle: So, Vic kind of had that idea in mind. He also had an idea of-- well, even simpler, just have a register you can set some bits in. So you jump to this location, it'll execute, then it'll actually look at this register; figure out which one of the bits was set and return to a fixed location, depending on which one those it was. So, you had some fixed number of places you could call the routine from, because the routine would return to one of these fixed number of places. And I said, "Well, that's certainly a novel idea, but how about if we put a call stack in the machine?" Just to be a little more oriented toward…

Hendrie: He was really going to hardwire it in.

Pyle: Yes, he was trying to keep it really simple.

Hendrie: Well, keep it cheap.

Pyle: I'm pretty sure the number on the 8008 was about 6000 transistors. So, there weren't a lot of transistors to be playing with here.

Hendrie: Yes, exactly.

Pyle: And the idea of a call stack meant we had to have a memory unit baked in there, which we did. Boy, this is getting a little old now. I do remember at least in some machines, we had some external register chips. I don't remember if the 8008 had an external register chip you added to it, which was like a 4x4 register chip, but…

Hendrie: I don't remember either, so…

Pyle: But clearly we had the concept of a call stack. And it was limited depth, so there was very constrained. I think the one we specified was seven deep and we ended up with 15-deep stack, or something like that. But that's kind of the genesis of the 8008. We knew we needed an 8-bit machine, because we were dealing with 8-bit characters in the communications world.

Hendrie: Okay.

Pyle: Where one of the instructions involved -- where the ALU [Arithmetic-Logic Unit] involved parity generation. Because parity checking was a very common thing you did in the communications world. So, some of that kind of specialized stuff ended up working its way into the instruction set of the computer.

Hendrie: Okay, well it was also going to have a -- wasn't it going to have a serial shift register memory. That that influenced…

Pyle: Well, it was a serial machine.

Hendrie: Yes, okay, so obviously that influenced it, too, to make it a serial machine.

Pyle: Architecturally the serial nature of the machine didn't really impact us that much.

Hendrie: Okay, didn't appear in the architecture, particularly.

Pyle: Right, the only problem with it was that it was slow. It was almost like a drum machine. In the old days with the drum machines you pretty much had to figure out where the drum was going to be for your next instruction so that you could optimize execution of your program rather than every sequential program waiting for a full revolution of the drum.

Hendrie: Well, there actually were -- just a footnote, because I thought you would be interested -- there was a military drum computer, I think it was an airborne computer, that literally -- the next instruction had to be right where the drum was when the first one… And that's how you programmed it. <laughter> If you can believe that.

Pyle: Right. So, we thought we made a great advance, because we could instantaneously stop and start the drum. We had a zero inertia drum and so we could kind of stop the drum and execute for a while and then start the drum up and get to the next one. So we didn't really have that problem.

Hendrie: Yes.

Pyle: But the next problem we had was if you called a subroutine, it was like -- the registers at the time were 512-bits long and you had a parallel set of these shift registers. Datapoint was one of the early consumers of memory from Intel. I really remember the word at the time was Intel thought the only reason to get into the microprocessor business, and I am sure Vic probably said this, too, was that it would help them sell more memory chips. And we laughed about this many times later. Computer Terminal was consuming a fair number of these serial shift register memory chips to be the buffer memory for the screen and also some more buffer memory for some of the characters coming in. At the time, Computer Terminal was one of the big consumers of these little 512-bit shift registers. So that was the kind of memory we had available to us. So when we built this machine we decided to use that kind of memory for the programmable memory of the machine and also it meant we kept the pin count down on the machine. I think it was a 16- or 18-pin chip, because basically there was just – everything was serial, the addresses were serial, everything in that machine worked serially in and out of the machine. Architecturally we didn't worry about that too much. We didn't build in any special thing like always having where the next instruction was going to execute or anything like that. But when we got into doing the software, we did end up doing a fair amount of optimization work on really tight routines that had to execute quickly. Saying, okay, I call this function here and then I know it goes to there, so when it returns, instead of returning will actually jump to another place, get some more work done and then when that returns it optimizes where the position was in the 512 shifts, the modulo 512 that this whole thing worked in.

Hendrie: Yes. Okay.

Pyle: So, we did end up doing some programming tricks where we really had to tweak the speed of the system out.

Hendrie: Okay, yes, to get it to run really fast.

Pyle: Right. Well, really fast. Remember this thing was clocking… <laughs>

Hendrie: Well, no, no, relatively speaking really fast.

Pyle: …this thing was clocking at like a microsecond per clock or something like that.

Hendrie: I understand. But really fast in the sense that you weren't spending a lot of time just waiting -- where you wanted to come all the way around again.

Pyle: Right, that's right.

Hendrie: Okay, good. So this was happening basically when you were -- this initial work occurred when you were in college. You remember, what were you? You came back on vacation on Thanksgiving. You know whether you were a freshman or a sophomore or a junior?

Pyle: Well, that was 1969, I started in 1967, so, you know.

Hendrie: Probably a junior.

Pyle: Yes, I was a junior.

Hendrie: Okay, tell me a little bit more about your, you know, about your college career. You know, what sorts of things that you got into. You mentioned that you had a timesharing terminal that was really for working. Were there particular courses that really fascinated you?

Pyle: Right. I don't really remember the names of the courses. At the time, there both a number of machine architecture courses and compiler courses and operating system. I became most attracted to the operating system parts and how the operating systems were built back then, how you would do a multi-tasking operating system and those kinds of things. So, that became kind of the core of my interests. While the machine architecture was interesting stuff, it was, you know -- and certainly I always enjoyed working on the machine architectures and I actually ended up working on machine architectures once I got to Computer Terminal after that. I think the operating system aspect of it and the networking too. We were a school that got one of the early IMP [Interface Message Processor] boxes for the DARPANET and a guy named, his last name was Glaser, was running the Computer Science department

at the time. Why I can't remember his first name? I do remember he was blind, been blind since he was fairly young…

Hendrie: Oh my goodness.

Pyle: …and he had a seeing eye dog with him all the time and he was running the Computer Science Department. And he really got us hooked into the DARPA thing and a lot of the networking stuff, so that was pretty interesting, too. And because I had been doing a lot of communications work with the ham radio and stuff, kind of communications and networking and those kinds of things had always interested me quite a bit. And that kind of led me into some of the ARCNET stuff I ended up doing at Datapoint as well.

Hendrie: Okay.

Pyle: But at school I was kind of taking as many of the courses that would let me really understand the insides of the system and how the computer's operating system really work. I got a summer job, no not a summer job, a school-time job working in the Computer Science Department at Case about, I forgot exactly when, maybe about 1969 or 1970. Case decided to set up an independent commercial entity called Chi Corporation to buy and operate the UNIVAC 1108, which is a pretty significant investment. And the idea was rather than the school having to fund the whole thing they could farm out the machine to commercial interests as well and get enough money to kind of pay for the system and also have a computer available for the Computer Science students to work on. So, I ended up kind of getting a part-time job there. We would work on the operating system and fixing bugs and things like that. A number of us really kind of got into the insides of the operating system. I vaguely remember that the listing for the 1108 operating system was a stack of paper something like… <laughs>

Hendrie: About that high, yes.<laughs>

Pyle: …several feet thick, at least. And of course it was still all punched cards. You had to kind of go in there and look at your line numbers and key in which line numbers you are changing and all this stuff. The age of really, kind of dynamic online debugging and all that kind of stuff was still kind of on the way at the time. Also this was a time when the Vietnam War was pretty active, And so there was the consideration of the draft and there was the lottery system back then. And I was like number 50 in the lottery and I was going to get drafted. <laughs>

Hendrie: Okay.

Pyle: And my student visa was going to run out after four years. And one of the things that happened was, I was spending a fair amount of time on these computer science courses. Really getting into the guts of the operating system and all these things that I wasn't going to be able to do in the electrical engineering course. I wasn't going to be able to get an undergraduate degree in this stuff I had a complete passion for. And I ended up in kind of a position after four years of having to make a change and not having gotten all I needed to get an undergraduate degree out of the school.

Hendrie: Okay, we need to change tapes.

Pyle: Okay.

Hendrie: You were-- we had left off, you were talking about the draft and your status at the school at that time based on the sort of-- your switch in interest to computer science from electrical engineering. At least a drift in that direction.

Pyle: Right. Well, like I said before, at the time, Case really didn't have an undergraduate computer science degree. And I'm one of these people that gets really involved in something and I have a huge passion for it and then I will sometimes fall down a little bit in terms of doing those other things you really have to do to do something. So I was in a position where, fortunately in some sense, by this time Vic Poor had gone working for Computer Terminal Corporation in Texas and he met some people that had some contacts with some other folks in the Texas National Guard. And through that connection, he discovered that there were actually, believe it or not, some openings in the Texas National Guard at the time when the Ohio National Guard, for example, had a six-month waiting list on it. And so if you were going to get drafted and there was a six-month waiting list, it turns out the waiting list wasn't going to do you any good. You were going to end up being drafted into the active military. So I went trotting down to San Antonio real quick and I talked to these guys and I ended up signing up for the Texas National Guard. And that was actually a very interesting experience. It's a six-year stint. I was in an infantry division. And you had to do the summertime thing and I ended up having to also do about a-- eight weeks of active duty for the training. So after my fourth year at Case and kind of the summer working in San Antonio, I ended up going off for-- well, actually, no it's four months, I guess four months of active duty. So this was 1971. I took off for active duty, went to Fort Dix, New Jersey, ended up-- interestingly enough, my, what they called individual training was at Fort Lee in Virginia which was five miles away from where my parents were living at the time. <laughs>

Hendrie: Oh, that's really nice.

Pyle: So it was kind of a weird circumstance, but I ended up being able to live off-base just fortuitously. And so, you know, got through that training and all that and got back to San Antonio and really kind of got into the whole operating system and machine development cycle we had there.

Hendrie: So, basically, you did your four years, but the mix of things, you didn't end up-- you couldn't graduate in either of those things that you had spent…

Pyle: That's right, yeah.

Hendrie: All right. So you just said, well, I don't want to go to Vietnam, I'm not going to stick around for another semester or year and finish one or the other.

Pyle: Yes. That was-- that really wasn't an option. But back at that time, after four years your student visa was up.

Hendrie: .Ah,your student deferment.

Pyle: Right. Your student deferment was up. Yes, I guess that's the right term. And you didn't really have an option. You either-- you had to get into some kind of service. Some of my friends-- one of my friends actually got into the National Health Service as an officer. So there were actually other options besides the National Guard and the reserves and things like that.

Hendrie: Okay, all right.

Pyle: Anyway, so that ended up kind of forcing me at that time into more or less full-time employment and activity in the actual area I was really passionate about and really wanted to get into anyway. So-- and Vic kept saying, hey, doing all this good work and don't worry too much about going back to school right now, you can do that later kind of thing. <laughter>

Hendrie: Vic's saying, I need this work done and Harry can do it.

Pyle: So that's kind of how that ended up.

Hendrie: Okay. All right. So you joined Datapoint full time. What were you-- what did you do when you initially got to Datapoint? Was it a continuation of something you'd been working on, you know, part time at Case or...

Pyle: Yes, pretty much. I ended up having to revamp the assembler to actually run on the machine itself. And we built a little what we called a Cassette Tape Operating System. We called it CTOS. Now in those days the machine had a cassette tape in it and we-- the engineers modified an audio deck to allow the tape to be started and stopped with a pair of solenoids. And...

Hendrie: Oh, that literally just pushed the lever? I mean…

Pyle: Well, they reengineered some of the guts of the deck so that-- but it was audio tape and it was audio heads and they basically put an audio tone modulation scheme on the tape and then decoded that for the digital data. And you had the ability to go forward, backward-- you could actually read the tape backwards, and fast-forward and rewind. So the fast-forward and rewind you were kind of on your own as to-- and there was an end-of-...

Hendrie: Where you were going to end up.

Pyle: Right, and there was optical leader sensors and you could tell when you got on the leader or at the end of the tape. And so we actually built a little operating system. At that time, the original machines had, let me get this straight, 8KB of random access memory. So we had to figure out how to squeeze...

Hendrie: So this was still-- this was now random access. They'd gone away from the serial…

Pyle: Oh, no, no, I'm sorry. The original machines were 8KB of serial memory.

Hendrie: Okay, of serial memory.

Pyle: We built a serial machine. I don't know if you got the whole story on the relationship between the 8008 and both Intel and TI [Texas Instruments] and the machines we were trying to build. But we were going through a process of trying to get either TI, I think we actually selected TI to make the first crack at it, and then Intel to build a solid state single-chip version of the machine. TI really was not able to get something running at the time. And Intel got something running, but it required a board about this big and maybe 25 or 30 discrete ICs to support the chip, to actually do all the shifting in and shifting out, support the memory chips and all that kind of thing. So our engineers-- our electrical engineers said: “Oh, boy, there's a board with all this stuff on it, maybe we can do better than that just by, you know, putting the remaining medium-scale integration chips on to do the CPU parts of this thing and build a discrete machine and not have to wait for Intel to finish the 8008 chip.”

Hendrie: Right.

Pyle: Intel was delaying things and we were going through a rough patch back at that time. There was kind of a recession and they wanted more money and we needed a product and, you know, all those kinds of things happened.

Hendrie: Yes, I understand.

Pyle: So we ended up with a machine, it was a serial machine. It was kind of ready to take the 8008 chip if we ever could get one to put in it.

Hendrie: So, it was designed in a way that there was a clear interface where the 8008 might go, but in place of that was MSI [medium scale integration]?

Pyle: Right. Right. And we built an operating system to operate in it, all 8,000 bytes of memory and we needed a little bit of memory left for the actual application to run. So, as I recall, we had to squeeze the operating system into 4,000 bytes of memory and that included a little-- the loader was a piece of hardware. So the hardware could actually read the tape and stream a set of bytes in the memory and start executing it. So there was a boot block that got things going. We had kind of a scheme for bringing in different com ports if we needed them, and then there was a little bit of space for the program to run. The reality was, it wasn't all that useful. We also had some specialized programs that didn't really use the

operating system so much, that were just communication-control programs, kind of following the initial idea of just building a communication controller.

Hendrie: All right. So you had-- yes. So that had been the sort of-- there was a thrust back.

Pyle: Right.

Hendrie: And then there was looking at could we do an operating system and make this a little bit more general purpose and more flexible?

Pyle: Yes, right. So the 8,000-byte machine clearly wasn't going to hack it. Meanwhile, we-- I believe it was Intel and TI both were developing random access memory chips. And this was going to change the whole nature of the machine because with random access things got a lot faster.

Hendrie: Yes. Could I just roll back just for a second? After you sort of worked out the architecture, then they built some prototypes so that you could, you know, do some software?

Pyle: Oh, yes, we had prototype machines.

Hendrie: You built prototypes out of MSI or whatever. That the architecture…

Pyle: What I don't know is if there were really prototypes where they were intended to be production machines. I don't remember…

Hendrie: …whether you built MSI production machines while you were waiting for the chip.

Pyle: Of that first generation.

Hendrie: Of the first generation while you were waiting for the chip.

Pyle: Right. That's right. Then after that, the company wanted to kind of get on with building machines and we kind of left Intel behind. My vague memory was we kind of had a parting of the ways with Intel where we said, Intel, we're not going to wait another year and pay you another $750,000 or whatever the number was, but you can have all the design. Essentially, you can take this design and commercialize it any way you want and we will go off and build our own machines using MSI and discrete components and logic like that. It's a vague memory. I wasn't involved so much in the business side of things back then.

Hendrie: Yes, I understand. I think that is what happened. That's what I remember Vic saying on his. Okay.

Pyle: And so we kind of got into the business of figuring out what the next generation machine would be that would be more commercially viable. As I recall, that one had, like, 16K bytes of memory. And we had the ability to put some more programming into it. Also, right around that time, we started getting the idea of hooking a hard drive system to it.

Hendrie: Instead of cassette tapes?

Pyle: Instead of cassette tapes because they were clearly going to be a problem with doing much. Although, by the way, I took one of the cassette tape only machines, because basically, the cassette tape was built into the machine and the hard drive was added on later. And I literally took one of those cassette tape machines with me to the field in the Texas National Guard because at that time I was in a unit that was doing the-- keeping track of all the material, and especially the food supplies. We were in a unit that actually supplied and delivered all the food from the warehouse to the various mess halls and things like that. And so we did all this by hand back then and I was getting very tired of doing that. And I kind of got the idea that, hey, I'll drag one of these machines out there, wrote a little program on this thing that actually kept track of all the inventory. You could fill the forms out on the machine and it would write things, but it was still all done on cassette tape. So we did actually have some applications where we could do some useful work on it. And, of course, the generals came by and thought this was cool, sitting under a tree with a generator and this machine sitting there running.

Hendrie: <laughter> A generator, I love it. Portable application.

Pyle: Yes, right. It was-- but the original 2200 was not larger than an IBM Selectric Typewriter, which is what it was styled to be like it would fit on -- if your IBM Selectric with the old ball would fit on your desk, supposedly, one of these things would fit on your desk, too. So it was a reasonably portable machine at the time.

Hendrie: All right. Good.

Pyle: And so we basically evolved there. We got very much into writing the assemblers, then Vic was very instrumental in coming up with a business language. Something where you didn't have to write machine code but could write more business-oriented code and that was called DATABUS. And so I got involved in writing the first DATABUS compilers and also working on the disk operating system so that we could actually load programs off the disk and store files on the disk. An operating system was kind of a stretch there. I mean, it was really a set of subroutines, almost more like a BIOS [Basic Input/Output System], but it did handle a file system on the disk. So-- and even, as I recall on the tapes, we had a file system on the tape as well. So a programmer could then, say, open a file and write some stuff to it, read some stuff from it.

Hendrie: And the operating system, or the drivers, I mean, whatever you called that, would then go find it?

Pyle: Right.

Hendrie: Based on the-- what you asked it to do?

Pyle: Right.

Hendrie: All right.

Pyle: So we had this disk operating system. I think we called it DOS. I don't remember exactly the name for it. And we ended up with this business language called DATABUS which really made it a lot easier for somebody to write business applications. And I think that's about the point at which the machine started to become kind of commercially viable. There was a stretch in there-- I'm having a hard time remembering the exact time. Back in the early 1970s we went through a recession. And the company was having a hard time financially supporting the development of these new machines. And we had to do something. So we actually spent some time programming on some HP minicomputers. And we built some systems. One was a-- some kind of a communication multiplexor controller and another one involved-- you know, I don't really remember exactly what applications we were doing on those. But we got involved with the HP minicomputers. And HP came up with this multi-terminal BASIC system that they could run on those computers.

Hendrie: Yes.

Pyle: And that gave us the idea-- and it was interpretative, interpretative BASIC. And this DATABUS language was an interpretative thing, too. And that gave us the idea that mixing that with the disk we could perhaps do something along the same architectural lines as the HP timesharing BASIC system could do, but do it with this business language. And that's when we came up with the product we called DATASHARE. So DATASHARE was essentially the ability to take your DATABUS programs and run them multiple-- or, you know, multiple sessions on a machine that could support dumb terminals. So we were kind of going full cycle here. We still had the dumb terminal business. And that's one of the things we used the HP minicomputers for. It's essentially a communication controller that could take a half a dozen or a dozen of these dumb terminals, wire them all in the minicomputer and have a higher speed link going up to the mainframe. And it would use the mainframe communication protocol and, basically, multiplex that out a bunch of-- amongst a bunch of dumb terminals.

Hendrie: Rather than having each terminal connected directly to the mainframe with all the protocols in it spending all its time...

Pyle: Right. So, as you can see, things evolved. You never quite know in a small business in a growing industry where you're going to go with this stuff. The concept of multiple terminals connected and the

idea of the multiple sessions of BASIC being run on a minicomputer and the fact that we were evolving our little computer to the point where it was powerful enough to actually do multiple things at one time, slowly evolved us into the DATASHARE business. And that became one of the first really successful businesses as I recall.

Hendrie: Okay.

Pyle: I remember vaguely Vic saying at one point, you know, this thing ended up being responsible for, I don't know, so many millions of dollars worth of business I've forgotten what the numbers really were. But it turned into a very significant part of the whole business we ended up evolving into at Datapoint.

Hendrie: Can I roll back to this data language?

Pyle: DATABUS, we called it originally.

Hendrie: Yes, DATABUS that you called it. Now where-- you know, how was that developed? I mean, who-- were you involved at all in specing it? You wrote the interpreter for it. But, who invented the language?

Pyle: I think Vic did.

Hendrie: You think Vic did all by himself? <laughs>

Pyle: My recollection that he had the germ of the idea. Now, what fed into that and what got him going there, we had been doing BASIC in the past. You know, at Frederick we had an IBM 1103 I think was the name of one of the minicomputers and we then used that as a business machine back at Frederick Electronics. I did some programming on it, but I don't remember what it was I did on that machine. I didn't do much on it. It might have been FORTRAN or something we were running on that machine. And so I think ideas were converging and they came together, I'm pretty sure, pretty much in his head. And I-- what I don't know is how much Jonathan Schmidt got involved because Jonathan was a very great idea kind of guy and he'd think about different ways and out-of-the-box kind of things. And he and Vic would talk about stuff all the time, too. So...

Hendrie: So he might have been involved in the...

Pyle: Right.

Hendrie: ...in the definition of the language?

Pyle: That's right. And then at some point, you know, he started giving this -- then I was kind of in some degree that circle of people that kind of talked a lot about various things we could do. And we started looking at the realities of the language and what we could really do with it and those kinds of things. And so I ended up getting involved in, how are we going to implement this thing and what kind of approach can we take and all those kinds of things.

Hendrie: Yes.

Pyle: But it's just my recollection that it was kind of a germ of an idea that formed in Vic's mind and then that blossomed into, you know, a more formal thing. I think he was largely involved in writing essentially the first specification of what he wanted that thing to do. The DATASHARE idea really came later, as I recall. It was this additional idea, hey, if a minicomputer, an HP minicomputer running multiple sessions of basic on a hard drive can do this, we're doing interpretative language, why can't we do the same kind of thing. So I think that kind of came in a little bit later once we had had some success with the DATABUS language and began to do some programming with it.

Hendrie: Okay. So now did you have the data bus language work on-- the 2200 was the first programmable glass teletype.

Pyle: Yes. I'm having a very hard time remembering whether…

Hendrie: …which ones.

Pyle: because we also had this thing called the 5500 and the 6600.

Hendrie: I know. There was a 5500 I wanted to understand what that was.

Pyle: What I don't remember, I think we still called one of the models the 22-- when we first went to the random access memory, I'm pretty sure we still called that the 2200. We had a version of the 2200. So what I don't remember was, was the pure serial version ever really a product or did we actually turn that into the random access version and that's when the first real product started to happen.

Hendrie: Ah,okay.

Pyle: I think it could have easily gone that way because the original machines were really not very capable. And if I recall correctly...

Hendrie: Well, they were perfectly good glass teletypes. I mean, they could be sold that way. But they weren't very capable of doing anything else.

Pyle: Right. And they were kind of expensive. <laughs> One of the problems was, you know, as much as it was kind of hard to keep reengineering the glass teletypes, just having some dedicated logic to do that still was kind of cheaper than this whole board full of general purpose computing and fairly expensive memory chips and all this other stuff, you know? That's how technologies evolve, right? You come out with something that's kind of expensive, maybe the early adopters will want to spend some money on it, but it doesn't become economically viable until more things happen and more evolution occurs to really kick that off. So I think that was another impetus behind the DATASHARE idea was, well, we've got this fairly expensive machine, now it's got a hard drive on it, those things are expensive. I forgot what the drives were called, but the IBM 1103 used them. It was an IBM Winchester thing, you got a plastic cartridge this big around with a single platter in it. The first ones were 2-1/2 megabytes of memory. And they were relatively expensive beasts. And then it was like, okay, how do we amortize this across enough business usage? Can't put one of these on everybody's desk, maybe every programmer's desk, but certainly not on every person trying to do some business work with it. And that really was the motivator for trying to figure out how to do this sharing system. But the early machines, I do believe we had a version of DATABUS that ran on the original-- at least on the random access version of the 2200 machines.

Hendrie: Okay. All right. So when you had this DATASHARE you were-- did you have your own computer or was this an HP machine that you sold?

Pyle: Well, we had an HP in the office. But it was running BASIC. And so we became familiar with...

Hendrie: With how it worked and how it did it.

Pyle: Architecturally, they had this-- they had a drum on that machine, a pretty expensive thing. That was essentially their caching memory for the sessions that were going on. But what we ended up doing was we ended up with a drive control architecture that had four 256-byte memory segments in it. And, as I recall at the time, the drive had 256-byte memory blocks on it. So you could read a block off the drive into one of these four cache memories essentially with random access memory on it into the controller. Then you could go in and randomly access the controller's 256-byte block. So it was really a cache and we had random access to it over the I/O channel.

Hendrie: Yes.

Pyle: So what we decided to do is three of these we would use for caching programs. This was in the initial design. It got bigger and more complicated with the later designs. But, basically, what we would do is when we wanted to execute a chunk of the DATABUS code we would read the chunk we needed to go to into the cache. 256 bytes might not seem like a lot, but since it was interpretive, it was highly compact and we highly compactified the order codes for this thing. So a fair amount of program would actually fit in 256 bytes. And so we could have three of these threads of execution essentially running in these three cache memories. The fourth one we kept for data I/O. So whenever some program would actually read or write a block of data memory-- data storage, you know, database kind of information, then we would use that fourth cache area in the controller to do that. But, basically, we could be executing the three different threads out of these three different cache blocks and if one blocked on a data operation or something, then we could go execute more code out of the other one. And that way, we kept the

swapping down a little bit, too, because then we could then swap in the next block we really had to swap in for the next execution.

Hendrie: That's a clever system. That's excellent.

Pyle: <laughs> It's kind of interesting how the hardware engineers kind of came up with the design driven largely by hardware constraints and the chips they had and things like that, and then we software techs looked at that and said, oh, how could we make use of this kind of thing. And we evolved an architecture that was kind of driven by that architecture. It wasn't like we sat down at a white board and had this grand scheme for how everything is going to work in the first place.

Hendrie: Yes, okay. Fascinating. All right. So you had-- you were working on the software for all of these different things. How many people were there working on software and how many on hardware? I mean, obviously, it was changing all the time, but roughly?

Pyle: Yes. In the early days I would say there was probably a half a dozen of us working on the software side of things and I think more than that on the hardware side. This is a long time ago.

Hendrie: I understand.

Pyle: I would have guessed there was maybe a dozen folks in the hardware engineering team. And then there was a manufacturing division, too.

Hendrie: Yes, of course.

Pyle: And there were probably more folks working on some of the legacy products and things like that, too.

Hendrie: Yes. Okay. All right, good.

Pyle: But, yes, and some of them were college kids, you know, we-- so we'd grow a little bit in the summertime because these college kids would come in and we'd have more people working during the summer and doing things. And then-- and I was one of the college kids for a while until I moved down there full time.

Hendrie: Yes, exactly.

Pyle: And then over time, as the business became a little more successful, we-- you know, we grew up. It was an interesting time because when I first joined Computer Terminal, as I recall, the entire company had 175 people, a large chunk of which were in the manufacturing business. And when I joined they

were in solid production on the glass teletypes, if I could put it that way. And I kind of-- over the years, I was there I think a total of 13 years, and watched that company grow to about 9,000 people at the peak. And so it was interesting to watch it change organizationally and kind of go through several different phases and all of that.

Hendrie: Exactly, exactly.

Pyle: And it was a very interesting experience to be able to go through that whole evolutionary thing.

Hendrie: Tell me some more about other things that, you know, as you-- you know, if you can remember some of the sequences of things you worked on at Datapoint.

Pyle: Right. Well, back on the machines again, I got involved in some of the hardware. I worked pretty closely with a fellow named Gary Asbell who was a young engineer at the time. He kind of worked his way up through the ranks, too. And we got kind of into how to design the actual execution logic of the machine and things like that. We also then evolved to the 5500 design which was a microengine-based machine. So the 2200 had just a big pile of logic that knew how to decode all the various instructions and caused the state machine to go through the various execution sequences.

Hendrie: Yes.

Pyle: In the 5500 we evolved to a micro-machine which had a simpler instruction set. The registers and things were tailored towards our instruction set for the 2200 and we extended that instruction set to add additional more complex string handling instructions and things like that because now that we had a microengine we could kind of go through and do string compares and stuff like that at higher speeds.

Hendrie: Yes, exactly.

Pyle: And I, as I recall, I'm pretty sure I wrote pretty much all the firmware for that thing originally. And that evolved into the 6600 and we evolved some of the instructions for that. And, by then, we had the DATASHARE thing in mind. So if I recall correctly, we added a few more instructions that made things more optimal for some of those kinds of executions. At some point, and I don't remember if this was the 5500 or the 6600, we got into a protected mode in the system. So we could actually run a multitasking operating system, we had a memory segmentation scheme where we had a logical address space that was still fairly small, like 64K if I recall correctly, but that could go to a larger physical memory disk space through memory map registers. And we were able to isolate the executing user mode program into just its area of memory so it couldn't go stomping all over the rest of the machine. And it had certain instructions it couldn't execute and there were interrupt instructions for getting into the kernel mode operation of the machine. So somewhere in there, it was either the 5500 or 6600, we evolved to that, even though we didn't have a great use for it initially, and that really kind of opened up the ability to get into the RMS operating system which-- the Resource Management System which was designed to run multiple...

Hendrie: Yes. What's RMS stand for?

Pyle: Resource Management System. And ARCNET was the attached resource computer and we did a-- that's what ARC stood for. So we had this idea of a fully distributed computer back then. It was-- this whole thing was a computer in some sense. And we initially did a version of ARC for DOS. But when we got around to doing the resource management system, that was a second generation completely redesigned from the ground up operating system in which the whole idea of the ARCNET aspect of it, the distributed networking, was fully baked into the design and architecture of the operating system.

Hendrie: Ah, okay.

Pyle: So, yeah, it was kind of moving through here. We kind of went through the 6600 and got the whole DATASHARE thing up and running. We even got DATASHARE to the point where it could work on top of ARCNET as I recall. DATASHARE was kind of layered on top of the DOS operating system. And then we went and...

Hendrie: Then DATASHARE's basic capability was to have a bunch of,,,

Pyle: Dumb terminals.

Hendrie: …glass teletype...

Pyle: Tied into this one machine, all executing DATABUS programs. So the programming language was DATABUS. I think there was even an effort or maybe it didn't actually happen. There was some kind of an ANSI standard done around DATABUS, the DATABUS language.

Hendrie: Oh really? Okay.

Pyle: That-- so that other people could implement DATABUS interpreters on other machines besides the Data Point machine after a while. So in any case, the-- we realized that as the machines became more capable and we wanted to build a networking aspect in and we wanted to have distributed storage, we actually had machines-- one of these RMS machines could act as a server, such as a storage server. So you could kind of access anybody's files, like you can do with PCs in your home network now. You can essentially have file shares anywhere. So we would end up with bigger machines that had lots of storage on them and then individual machines that literally had network boot ROMs in them and would boot over the network and not have to have any kind of a hard drive on them at all. And for later on at Datapoint, a lot of the programmers really didn't have hard drives on their systems. We tended to do this all over the network.

Hendrie: Wow, okay. Well, let's go back and talk a little bit about the-- you know, when-- where did ARCNET come from?

Pyle: So we had the idea kind of in the DATASHARE days, so we evolved from the single DATABUS thing to kind of multiple machines tied to one machine. But then we ran out of capacity there. It turned out more than about...

Hendrie: And these are all-- it's serial lines?

Pyle: Yes. So we had a special serial controller box that could hook to the I/O port of the 5500 and 6600 machines. But-- and they could run 9,600 bits per second or 19.2 to whatever the glass teletypes wanted to work at. But at some point, you just ran into capacity problems. And my version of the story is that we had a fairly bright field engineer who had been deploying DATASHARE systems in the field and helping his customers do this, but you get up to about a half a dozen-- even though the I/O controller box could handle eight terminals, you get past about six and you pretty much maxed out the machine, if you started doing real work on the machine. And this fellow said, you know, what my customers need is they need to be able to just have more computing power against the common database. So if there was a way-- and at the time, the idea kind of came about. He says, you know, we're sitting here streaming this 2-1/2 megabit data stream onto the heads of the drive, what if we could take that two and a half megabit data stream and just get it to some other controller on some other DATASHARE machine so that the two machines could literally share the hard drive. So they had a common database between these two machines. And now you can multiply up, all you have to do is just keep buying more DATASHARE machines. As you needed to add capacity to the system you'd just buy another machine. You don't have to rip out all the old ones or change everything you're already doing. And we started thinking about this a little bit. I started thinking about it.

Hendrie: I love it. The customer often knows best where you want to go. <laughter>

Pyle: Right, I remember exactly the conversation I had with this guy in a little hole in the wall Italian restaurant we'd had meatball sandwiches for lunch at. And we had a meatball sandwich lunch at this thing and this guy was kind of describing this idea. And somehow that triggered that idea of, well, you know, a piece of wire, certainly a piece of coax, for example, you could easily stream a couple of megabits a second down. I'd been reading some literature about how IBM did their big mainframe systems and they had these big huge cables that had multiple coaxes in them actually. And they had a way of tapping these things from one I/O unit to the next. And I said, well, gee, if IBM can come and have multiple I/O units hanging on this tapped coax bus, maybe we could do something like this. And this was before the Ethernet articles came out in the Journal of the ACM and places like that, or Communications of the ACM.

Hendrie: Yes, right.

Pyle: And so I think it was an idea that was kind of time for it to happen somehow. So...

Hendrie: A user need and some idea of a possible way to implement it…

Pyle: Right. So one of the things we were doing…

Hendrie: …came together.

Pyle: And other things. It's fascinating how these things all come together. The company at the time was struggling with building a bigger machine. Obviously, you know, one of the solutions is to keep building bigger and bigger iron, right? And hooking more and more and more terminals into it.

Hendrie: Exactly, right.

Pyle: For one reason or another, the engineering team was having a difficulty making that happen. And we had all these little machines and they were all there, they were in place and all the software was running, you know? I mean-- and there were some questions about-- and customers already had them and they wanted to add on. They didn't want to do what we often called a forklift upgrade where you pick this stuff up and bring new stuff in. And so we kind of started this on the side development effort in the R&D department because at this time-- by this time, Vic was now running the research and development department. The main line core engineering group was actually in a totally different building campus than we were at at the time.

Hendrie: Oh my goodness, okay.

Pyle: And they were off building these big machines. And we did a fair amount of work with them. But-- so they'd kind of taken over the main line development of machines. And Vic had fallen back into the position of being the VP of R&D. So we said, oh, here's a little incubation thing, let's see if we can do this kind of thing. So we started working on a little network that had a physically tapped coax line just like Ethernet had. We didn't know they had that until we'd kind of done some work. I did actually a little bit of the electrical engineering work to see how we might be able to build transceivers that could easily tap into these things.

Hendrie: <laughs>

Pyle: After a few months of that, we pretty much came to the idea that we should do point-to-point and not physically tapped networks. So rather than going down the path of what Ethernet originally did with all of its specially marked coax and the things you clamped on and screwed in and all this heavy-duty stuff, we kind of got to the idea, why don't we just have a box in the closet and every terminal runs a wire up there. We still used a piece of coax it turned out because that was a nice high bandwidth easy to drive thing. And there were some FCC codes considerations we didn't really know how to get around with unshielded twisted pair and stuff.

Hendrie: Yes, radiation.

Pyle: We ended up with the point-to-point kind of thing you end up with Ethernet now. And you ended up with a hub box that knew how to relay as signals and regenerate them and do all those kinds of things. And we ended up also designing the ARCNET controller to drive this thing. And that's when John Murphy

came into the picture. And he was an engineer that had been working at Singer Business Systems. And he-- we brought him in and said, hey-- he was an electrical engineer and we said, we need to do this thing. We had some micro-controller architectures kind of floating around that we'd been fooling with. And so he built a micro-controller engine-based machine that could do the actual protocol work for ARCMET. And there were probably during that phase about a half a dozen of us all got involved; Jonathan, Vic, John Murphy, Gary Asbell, myself and a fellow named Mike Green, I think was kind of around at the time. And we all kind of were tossing ideas around about how we could do it, what protocol we could use, the whole thing, the approach Ethernet had taken with the random back-offs and all that stuff, you know. There were clearly some issues with that. And John Murphy, I remember him saying, I will never design randomness into the system if I can get away without doing that.

Hendrie: Yes. <laughter>

Pyle: So we ended up with a very deterministic thing that took more time, had a greater initial latencies to get a packet on the network, but clearly had predictability. And one of the reasons ARCNET -- another tangent, ARCNET kind of made its way into the industrial world was that it had very predictable latencies. And so some of these people that wanted to use it in a machine control, the industrial world said, you know, rather than having speed, I'd rather have predictability. So I know by worst case, latency is going to be X and, therefore, I can design around those kinds of things. So it was kind of interesting how ARCNET then finally kind of made its way off into the industrial world, whereas Ethernet ended up going back and becoming the dominant force in the commercial world. And Ethernet wandered off in the more predictable direction with point-to-point and flow-control and all these good things that have happened with full duplex. You know, Ethernet is-- it's evolved.

Hendrie: Yes, it's really funny. All right. Can we change a tape?

Pyle: Sure.

Hendrie: Okay.

Pyle: Okay, so we were talking about kind of the evolution of ARCNET in some sense.

Hendrie: Yes, exactly.

Pyle: Because it kind of came out of all of this business. It was a fascinating sequence of how can we take little machines and just make more. We know how to make them, how can make more and hook them together, and using kind of a parallel technology. And it was kind of interesting that about the same time the Ethernet articles came out in the Communications of the ACM and we kind of went down one path with ARCNET and Ethernet went down another path. And we built, like I said before, we built this Resource Management System that was really, had the whole networking infrastructure baked into the genetics of that system. So it was fundamentally a network operating system. The networking wasn't just an add-on in the end.

Hendrie: Okay and that would run on a 5500 or a 6600?

Pyle: You know, I don't remember if we required the 6600 by that time to run the Resource Management System. I think we added some additional capabilities in the machines. We also came up with some more machines. There was the 1800, which was the-- you know, all the original machines had this short little tube in them, really special little tube. I think Ball made them for us or something. And they had eighty character columns and like twelve lines. It was the same number that was on a Hollerith card from some totally strange reason.

Hendrie: Right, okay.

Pyle: But for real desktop use, for individual networked computer use now, you'd really need something bigger. So they evolved back into more of the form factor of the glass teletype, but by now they could cram the electronics of the main CPU back into that space. So we had an 1800 and a 3800, I think, machine. Which were fairly capable machines, but they had the taller 25 line screens on them. We gave them some more capabilities in the graphics area, and those became kind of the mainstream networked machines that could actually run your program locally but have your storage stored other places, things like that.

Hendrie: All right, so can you give me some more flavor of what you were doing on this when you were developing the ARCNET, sort of how that story unfolds? You said there was this group, and you'd sort of come up with a basic idea, that had been triggered over a meatball sandwich <laughter> from a field engineer. And so is there-- I don't know, it's a long time ago, but can you remember any more sort of flavor of that whole -- what happened there?

Pyle: Yes, I mean clearly I was involved in-- Having kind of come up through an electrical engineering background, I always kind of kept one finger in the architectures of the machines, and the designs, and some of the logic, and some of the electrical aspects of it. So you know, but my core work was really in the operating system side of what we were doing. So I was involved in how we were going to actually turn this system, the ARCNET system, into something we could use in the operating system itself. And we spent a fair amount of time. I mean I did some of the thinking about the DOS, but there was another fellow named Gordon Peterson who actually did almost all of the implementation work on making the DOS system work with the ARCNET itself. I really moved more over into the kind of the whole design of the RMS system. Another fellow I worked with, a fellow named Klaus Landberg (???), had been a customer of Datapoint's in Denmark, working for one of the main minicomputer companies over there. And he eventually ended up coming over and working for us at Datapoint. And he was also one of the founding fathers of the new RMS operating system. He had had a fair amount of background in a multi-tasking system on one of the minicomputers. I've forgotten if they were using Data Generals or something like that. And so he brought some good background with him and how to do a real multi-tasking, you know, multi-purpose operating system. And he and I were two of the key folks that spent a fair amount of time working on this RMS thing. So we did a lot of the background and the basic-- I say I wrote the-- He and I wrote the first hundred thousand lines of the assembler language. We were still operating in the assembler language domain even when we did the RMS operating system. We were kind of getting into C and, or something like it. We had something called DASL. One of our engineers, a

fellow named Gene Hughes, was really into programming languages. And he had been looking at C, and he knew he could do it better and he really could do it better. <laughter>

Hendrie: Of course, every really good programmer does that.

Pyle: So he cooked up this C-like language, very C-like language, we calledDASL. The D was for Datapoint, anyway. <laughter> In any case, Klaus and I were really spending a lot of time on the operating system. But he DASL stuff really wasn't far enough along to really have us depend on that as the core guts of the operating system. So we went ahead and-- a lot of the coding was done by kind of writing C-like code in comments, and then you'd come back and kind of hand-compile that effectively into the assembler code.

Hendrie: I see, okay.

Pyle: But I think, we counted it up one time, we ended up with about a hundred thousand lines of assembler code. We had modern constructs of threading, and semaphores, and file systems. And we cooked up a complete dynamic file system for the disks that would handle a large number of files in the big hierarchy and things like that. That was kind of my core work during that period of time. Before that I'd been doing a lot of work on this DATASHARE thing. I was one of the people that really got DATASHARE to the point where we could kind of deliver it. And then after DATASHARE, I wanted to move on and do some new things. So we did this-- we knew we were having to move into a new era with the operating system and with this resource management thing.

Hendrie: And the resource management thing knew about the ARCNET?

Pyle: Yes, it had ARCNET built into it.

Hendrie: So it understood that things were out in this distributed world and took commands and then…

Pyle: Right, and there was a boot protocol, too. So we even supported a boot protocol so the machines could boot over the net. We had a ROM on the machines so they didn't have to have a hard drive to boot. And so that was all part of it, too. It was very much oriented towards-- you know, the company made its money selling hardware. So it was kind of I guess the DEC and the Data General models and stuff like that. <laughs>

Hendrie: Yes exactly, of course.

Pyle: And so we were trying to figure out how to build a system that made people really want to take a lot of our hardware and couple it all together. So things like the boot protocol were very proprietary and very oriented towards our machine and things like that. What was kind of interesting was that later on, I guess in the early 1980s, Intel was now making more progress on their machines. We kind of consulted with them a little bit on the 8080.

Hendrie: They wanted to know what enhancements you had put in.

Pyle: Yes, and what things we would like to see in the 8080 if we were going to try to use their chip. I think the fellow named Stan Mazor was still involved at that time. So Stan Mazor was an application engineer that worked with us on the 8008. And I've only touched base with Stan a few times since then, but I think he was also working on the 8080 to try to understand what would be a good thing for the 8080. The trouble was, was by that we'd advanced our machines to 5500 or something, and the 8080 was still kind of a step behind. So we were on this weird curve where we were king of like this, and the microprocessors were kind of coming in like this. And eventually they crossed over some, and of course way overshadowed anything we could do with discrete componentry.

Hendrie: Because they had such cheap transistors, right, eventually. <laughs>

Pyle: So they were on Moore's Law, we were more on kind of a linear path of some kind, <laughter> I'm not exactly sure what. And so then the company was highly challenged to come up with architectures and machines and operating systems and all these things that could be converted over to full-blown microprocessors. And that was one of the things that Datapoint struggled with later on in life, is that we really struggled with how could we put a 286 processor in this thing and really take advantage of it. So there was another effort, and I think it was actually more or less successful, to rewrite RMS again in C, and build the whole system in a more portable way that could be ported into these other chip environments more easily. But that was post, my-- I actually went off. I became an individual contributor, a vice president. The company decided to make some vice president positions that would be individual contributors.

Hendrie: Okay, that's a good idea.

Pyle: And you could kind of go off and do things, and I decided I wanted to, actually interestingly enough, get more back into machine architecture in some sense. And I had the idea of how to take-- we'd had ARCNET. It was not the fastest thing on the planet, I think it got up to ten megabits or something like that, or twenty. We had evolved it to some higher speeds. We had these individual machines that were relatively small, and I was trying to figure out how we could essentially package a lot of this into a small form factor again. So instead of having a big machine that had a huge processor that did lots of things, can we take this kind of multiple small machine approach again.

<phone rings>

Pyle: And so we could take this multiple small machine approach again, but essentially blades in a rack.

Hendrie: Yes, okay.

Pyle: And the back plane of that thing could now be a hundred times faster than anything we can do on the network, but we could still take a networked approach to the architecture of the machine. And for that

matter, we were designing it such that you could power a blade down, pull it out, and replace it. So if something went wrong you could slip in a redundant blade. And we had all these redundancy mechanisms. We had redundant power supplies, we had common48V busses in the thing, with independent power supplies of the blades. And we had redundant 48V power supplies in the rack kind of thing. And you could have battery backup easily if you wanted to. And all those good kinds of things. So we kind of went down this…

Hendrie: Sounds like a great system. I love it. <laughter>

Pyle: Yes, unfortunately we kind of only got part way into it. We actually built some LSI [Large Scale Integration] chips, and I got involved working with a spin-off, or it was part of Storage Technology I believe out in Colorado. They were building LSI chips for their storage, their mass storage products. And I think to leverage some of their expertise in doing that, they were also offering that as a business where other people could come in and build chips. I think that was the company we were working with. So we actually ended up building some chips and building some boards that had these chips on them. And we got the system to a partially functional level, but then the world kind of changed stuff. So we didn't get to pursue that much further than that. But that was kind of where my thinking was going architecturally, was building blades that were basically on a network backbone. And of course, people do that all the time now with server blades in a rack and using fairly standard networking technologies.

Hendrie: Yes and then they figured out how to do over in the storage world not have to build just bigger and bigger disks, but…

Pyle: Lots of little ones. <laughter> On effectively a network.

Hendrie: …lots of little ones, yes, have an array of little ones.

Pyle: So I've always been fascinated with the networking side of things, and the architectures you can build around networks and those kinds of things.

Hendrie: All right. And then Datapoint got acquired, bought out.

Pyle: Yes, right. It struggled before that. I think, in my view, the PC caused a paradigm shift that a company that was by now the size and structure of Datapoint couldn't keep the car on the road, if I can put it that way. The road suddenly went like this. And the big huge thing with a lot of inertia was going like this. And to some degree it kind of went off the cliff, because it just couldn't make the turn. You know, you had a 9,000 person organization. There was a lot of momentum in place throughout the whole structure of the company. The way business was structured, the business models, all of those things, that just everything really got changed a lot by the evolution of the PC. And that was really the thing that--what was fascinating was that everybody before that time was just deathly afraid of IBM, and the big iron, and IBM was going to do everything and put everybody out of business in the computer world and all these things. <laughter> But a totally different thing happened.

Hendrie: But it's what put them out.

Pyle: IBM did it , but they didn't know they were doing it either.

Hendrie: Right, exactly.

Pyle: It's like they did this thing with the PC, they had no clue what they were doing. It changed the world in a way that disrupted companies like Datapoint and Datapoint got into difficulties that therefore led to all of the turmoil, and the buyouts, and all of those things.

Hendrie: Yes, so they were in difficulty, yes.

Pyle: Right, I left in 1984 before they really got totally tangled up in that stuff, I could see that things were going downhill fast. And there were these other little groups that were kind of breaking out of Datapoint around San Antonio and doing neat little things. One of them was building modems. And I went to a little company called Image Data, it was actually Photophone at the time, where we'd build a little communication device for visual communication using a video camera and fax modem technology. Rockwell, using integrated circuits, integrated circuits were changing the world, had managed to squeeze a fax modem onto a chip basically and get it cheap enough that you didn't need a big box like this now. You could have a relatively little board with a couple chips on it and do 9600 bits per second communication over long-distance phone lines. And then from there 14.4, and that was relatively unheard of back in 1984 timeframe. So suddenly it became possible to take a compressed image, and ship that over a phone line, and the amount of time didn't take forever. And microprocessor technology with embedded chips like the 186 that Intel was doing was getting cheap enough you could build a little board with a complete microprocessor embedded system on it for relatively little money. We could take essentially DOS and reengineer DOS to put it on the thing. We actually did reengineer DOS at that company and put it on there, but later on we ran real versions of DOS on these machines.

Hendrie: <laughs> Okay.

Pyle: But you could build embedded systems cheap enough that you could start doing things. So we got off into the business of being able to say, you know, a picture someplace else is worth a thousand words. In the business community, engineering, medical, those kinds of things it might be very useful to have a picture phone. And some other guys kind of had this vision and went off and founded this little company. And I got involved in it, and we actually got Vic Poor involved in it after a little while.

Hendrie: Is that right?

Pyle: And… <nods>.

Hendrie: Okay, but you got involved in it first.

Pyle: Right, yes. And we went off and did a bunch of very interesting things. Once again, communication oriented and did a lot of operating system kind of work and things like that.

Hendrie: Okay, so they did get to the point of having a product? I mean, tell me a little bit…

Pyle: Oh yes, yes. We had this thing called a Photophone. We actually had some custom plastics, it looked like a 1950s TV. I don't know who they got to design this thing <laughs> but it actually had an integrated nine-inch screen, and a little punch button pad on it, and a little arm with a CCD camera. Actually it started off with the Vidicon cameras, but pretty soon we got into CCD cameras. And it was a turnkey package. Just plug it in, hook it to your phone line, and it would switch between voice and sending a picture. So you only needed one phone line, you didn't have to have special lines. And you would plug your speakerphone into the back of this thing. This thing would plug into the phone line. You could be having your little conference on the thing, and when you would say: "Oh, let me show you a picture of that," you could push a button maybe fifteen seconds later, the image would show up...

<alarm rings>

Hendrie: Yes.

Pyle: …and the party on the other end could actually see what you're talking about. You could move a little pointer around on the screen, and you could say, this thing here and that thing there, instead of trying to describe exactly what you were talking about.

Hendrie: Oh my goodness, okay.

Pyle: And we made a fair amount of headway initially in like the automotive field for example, where they have remote engineering teams at the plants, and they're trying to figure out problems and solve problems. Where if you can save an hour on the production line, you just paid ten times over for this stuff.

Hendrie: Yes, right.

Pyle: So our marketing guys kind of found these kinds of applications for that thing, where the payoff just swamped the cost of the equipment. And we made some headway doing that. It was a venture capital based thing. You know, we worked for a long time to try to get it completely off the ground. And as most venture capital based things go, they eventually end up getting acquired by some bigger company, because it fits into their business model. But making it on your own in a whole brand new technology area is something that usually doesn't happen.

Hendrie: Yes, that's really hard, really hard. So how long did you-- you left in when? When did you leave to go there? 1984 you said?

Pyle: Yes, I left in 1984. Excuse me.

Hendrie: Here, we can put this on hold.

<small pause>

Hendrie: All right, you were saying that you went there in 1984. Now how long did you stay there?

Pyle: I stayed there until 1996, so twelve years.

Hendrie: Oh my goodness, for quite a long time, very long.

Pyle: Right. The first I would say five or six years was kind of the entrepreneurial phase of the company, but it wasn't getting really off the ground. So then some other companies came in. I think it was E-Systems [ph?] eventually bought the company, and then Raytheon ended up buying E-Systems. Those are two-- E-Systems is pretty big and Raytheon is gigantic, so it kind of got wrapped up.

Hendrie: Right.

Pyle: The company also kind of evolved from this more industrial oriented application we were thinking we were going after towards the medical field. It turns out radiologist have these pictures that are pretty much all they really need to see, <laughs> but it's the critical factor. And if you can send a radiogram, you know, across the country to a specialist in almost zero time, that can be very, very interesting. And we ended up evolving that business into more of a medical imaging transfer business, of the remote medical diagnosis kind of thing. And that's when E-Systems, which was trying to get into that business, got interested in the company and bought it. And then Raytheon also was trying to be in that business, and they ended up buying E-Systems. So it was interesting. We went through quite a bit of changes in the way we were dealing with images and the technologies we were using. So it wasn't all one job, you know. It was, once again, an evolution of a technology that we thought we were going to go one way with, and we kind of ended up going in a different direction with it. That's very often the way these emerging technologies really go.

Hendrie: Okay, now what sort of things did you particularly, personally do? I mean what was your focus on?

Pyle: Well, once again, I was kind of quasi hardware, software. I always have had an interest in kind of the hardware side of things. I was brought in to basically get going the telecommunications side of things, which included the engineering of the modem board and the software that drove it and some of those things. You know, being a young company, we went through some evolutions. I ended up becoming the VP of Engineering and pretty much driving the engineering effort. Which there was a certain hardware element to it, but we kept simplifying the hardware. The original people kind of had great visions for the hardware, and they built a really complicated piece of hardware. <laughter> We kept

driving more towards standard embedded chips, embedded operating systems. Eventually we got onto DOS so we could get away from even--

Hendrie: Inexpensive things.

Pyle: Inexpensive things. Especially remember, during this period of time the PC was evolving. So we were kind of tracking the evolution of PC hardware and software systems and shifting more and more of our effort over towards the application side of things, which was the image transfer, the interactivity, the connectivity to various image sources. So there was hardware involved there. We got some patents on something that was fairly novel. Once we got into the medical imaging business, we actually had to hook up to video outputs on all kinds of different machines, ultrasound, CT scanners, MRIs. All these kinds of machines were pretty much custom-built machines, for some reason they all decided they had to have a custom video format. And the industry was having a hard time getting together on a digital information standard. So say, here's an image file in a certain standard, so you could get it out of the machine and do something with it. So we were in this kind of in between, no-man's land of having to actually obtain the images from the video output to these machines, but every one of them was custom and different and all these kinds of things. So we ended up doing things like designing a piece of hardware, which was an image capture card, that was flexible and able to capture videos from a wide variety of different machines. And we'd been struggling with some off-the-shelf solutions, and they were either very expensive or not very flexible. So we ended up-- I ended up having an epiphany one day while I was out jogging. I always get my best ideas when I'm jogging in the morning kind of thing. And I got this idea of using essentially what amounted to a digitizing storage scope, where you would just digitize the heck out of whatever you were looking at coming in, and then go back and analyze it later. So we had enough smarts in the board to be able to detect the sync pulses, so we knew when a video frame had been completed.

Hendrie: Right.

Pyle: And we essentially did a high-speed digital capture of all of the video, including the sync pulses and everything, and then used the computer power to go back and analyze the stream, figure out where the edges of the pixels were evenly, because we had some pixel sampling. We could actually do all the phase locking on the pixels and all these things, essentially in software. And so now we had a very flexible technique. Just run a little set-up program that let the software analyze the signal, that it knew what the signature of the signal was, it could then every time it wanted to capture an image go use that signature as the pattern for which you were going to do all the sampling, including locking on the various pixels that were in the image. And we got a few patents on that, because it was very beneficial to being able to have a flexible, low-cost essentially capture system.

Hendrie: Yes, where the modifications for particular things. Well, I mean some of it was self-learning, and it was all in software, both of which are wonderful characteristics. <laughter>

Pyle: Right. So this is kind of the where the convergence of the hardware and software came in. I did all the analog front-end design. But at that point it was pretty much, you get chips off the shelf and figure out how to wire them together. All the high-speed amplifiers and things like that were basically chip components. An engineer worked for me, he did all the logic in programmable logic arrays. So this was

one of those save-your-company kind of things you often go through in an entrepreneurial setting, and we had to do this very short. I think we had three months from inception to delivered to a customer. Because the previous solution we were delivering and had committed to deliver, the previous company had fallen down on us and we just could not get that.

Hendrie: Oh my goodness.

Pyle: So now we had customer deliveries we had to make. And so we had three months and we had to do it. So we knew that by using programmable logic arrays they're a little bit more expensive, but we had total flexibility. We even downloaded the firmware into those things so that we could dynamically fix bugs in all the digital logic. All we had to do was get the analog front-end right and the digitizing sections right. And we did actually manage to build a board, which was essentially perfect the first time. And we got the boards, and we verified they worked. <laughs> We had simulated a lot of the software, but we actually got he software to work for the first time, spent about a month or two finishing up all the software algorithms that did this analyzing of the signal, and shipped the product out the door. And so we were building boards at this time, production boards, that at the same time we were kind of verifying that the whole scheme would even actually work. And that was probably one of the real highlights of that period of my life, because it was one of those things that just worked.

Hendrie: Exactly, do not -- exactly.

Pyle: You know, many things could have gone wrong. You chose all the right paths in terms of where you chose the flexibility and the approach you took. And the ability to, quote unquote, do it all in software allowed you to kind of ship the product out the door. And if you had some kind of a software bug of course you had a release disk they could go fix things with. So that was kind of a shift from hard hardware into soft hardware that was a very interesting experience.

Hendrie: Yes, that's great. I mean probably the benefit of knowing a lot about software and still having your hand in hardware allows you to sort of make those compromises. A pure software person wouldn't have dared do that. <laughter> And a hardware person wouldn't have been able to figure out, you know, wouldn't have dared to make software do all that work.

Pyle: Right, yes. Of course, you know, the world's changed again. I mean with your cell phone-- I went over to a system-on-a-chip conference not too long ago, and now you're having to put basically everything on the chip. And now they're struggling with how can they build chips that they can do some of this stuff fairly softly. You know, architectures that are fairly flexible, you have to change your compression algorithm for your camera phone. I mean you've got the flexibilities to do these kinds of things without having it all baked in this hardware. So the world keeps evolving. It's kind of fun to watch how things change but don't change at the same time.

Hendrie: Okay. So you stayed there till 1996. What made you decide to leave?

Pyle: Well, once again, the company had been bought by a big company that had been bought by an even bigger company. And they were going in directions that weren't all that encouraging looking to us. And I had an opportunity to come up to the Seattle area. Also, both my wife and I are not from hot Texas. San Antonio was a pretty nice place to live. It wasn't as bad as Houston for example, which is down at sea level and terribly humid.

Hendrie: Yes.

Pyle: But we'd ended up by then probably spending over twenty years in that area, and since we'd grown up in more temperate climates we kind of wanted to get back to something a little more temperate. And with some things that were happening up here. Of course Microsoft had a big huge sucking sound on the world when it comes to technology I guess. Some friends of mine I'd been working with at Datapoint and even at Image Data had come up here to work on a project, kind of a private project that was some pretty interesting technology work in it, involved distributed computer systems, and lots of communication between machines. The theme here, this thing about having a lot of machines tied together with a network, kind of follows through. And so it looked like kind of a fun project, but it was a very much non-commercial project. I mean there was one customer. And so I felt like maybe it was time for a little bit of a break, to get out of the commercial world for a little while, go off and work on more or less a sandbox project. And learn some new technologies and some new things, and how …

Hendrie: Recharge your batteries.

Pyle: …recharge my batteries. So I got to do that for about three years. And then that whole team ended up kind of moving into Microsoft and doing some home oriented technology things. At the time, Microsoft and the Hardware Group had a Connected Home Group and was trying to develop some home technologies. You know, things that were leveraged off the PC, but would also be some like control automation, some of those kinds of things, so it would make the PC more relevant in the home. And that's been kind of a theme I've been on now at Microsoft. I've moved into the group that's doing the media center edition of Windows. And once again, we're very much-- we want to make the PC not only a good productivity machine, but also a good entertainment machine. And as people have started using computers more and more for things like multi-media and photos and music and videos and all those kinds of things, the Media Center has been a way to kind of focus a version of the operating system essentially on a lot of those activities. So that's been kind of the direction I've gone moving up here.

Hendrie: Okay, Got off the one-of-a-kind project.

Pyle: Right.

Hendrie: Now was that part of Microsoft or not, the one-of-a-kind?

Pyle: No, no.

Hendrie: Yes, it was a separate deal.

Pyle: Yes.

Hendrie: It wasn't the deal for Bill's home with all the artwork that changes?

Pyle: Well, as a matter of fact, it was. <laughter>

Hendrie: Okay. I just guessed that. I don't know why that came into my mind. All right. Very good. Did that ever get finished and work? Yes, I assumed it --

Pyle: Oh yes. Yes, we had a good time doing that. And you know, a number of people thought that was totally open-ended, spend all the money you want, and all that stuff. It was actually run very much like a commercial operation. We had strict budgets, we had strict timelines, it was things we did have to work like a normal product of any type.

Hendrie: I don't think it's in Bill's nature to allow things to operate any other way. <laughter>

Pyle: Yes, right.

Hendrie: Exactly. You know, that isn't what he believes in. You don't get good work unless you have a little discipline to what you're doing. Okay and necessity is the mother of invention as you've indicated in a few places on these tapes. <laughs>

Pyle: Right. Yes.

Hendrie: Very good. Looking back, you know, just rolling back a little bit, we just have a few more minutes, back to Datapoint. What are the contributions that in your mind, you think back, and you say, you know, well now, that's something I really feel pretty good about? You know, the sort of--

Pyle: Right. Well, there was the evolution of the machines themselves. And I had a fair amount to do with that. And you know, I mean I remember writing-- making significant changes to the original 8008 instruction set desires, and I actually laid out an instruction set for the 8008. Unfortunately the document I had with my original hand-written notes on it got pulled into the legal doings with TI and all of that, and I've never seen a copy of it since. <laughs> It's sitting in some archive somewhere, I don't know.

Hendrie: You're kidding.

Pyle: No, but I mean I had all the original handwritten notes of laying out where the bits went, and how we were going to do the H and L register thing, and the instructions that dealt with, and all those kinds of things. So and evolving that up through the various machine architectures with microcoded machines, I thought that was-- you know, that contributed a lot to kind of what we did.

Hendrie: Yes, right.

Pyle: Certainly the work on DATASHARE I think contributed a lot to the success at Datapoint for some number of years there. The whole bit of kind of the coming about of ARCNET was- truly took the company in a fundamentally different direction with the distributed nature of doing that. And the creation of an operating system that really did have networking and distributed machines kind of stitched into it from the beginning was I though quite a contribution to the way we ended up doing things back in that era. So I guess those are kind of the highlights. You know, doing a lot of software during all that time, and--

Hendrie: Writing a hundred thousand lines of assembly code here and fifty thousand lines there. <laughter>

Pyle: …and there. Right.

Hendrie: And all of that sort of thing, which has to do with the implementation.

Pyle: Right. So I think at Datapoint those were probably key areas that I made things happen in those projects. Very oftern -- you always work with a number of people, right? But I'd just have ideas, oh yes, we could do it like this, oh yes, we could do it like that. And it's those little things. Each individual in the team will often have a cross current of experiences that ends up they have this idea that they can then contribute to how things work. And that happened a number of times there in those areas.

Hendrie: Yes, okay. Now I'm of course always curious about anything I can find out about the ancient history of the 8008. You were mentioning about your handwritten notes on the proposal to fix-- at what period was this? Was this after you got to Datapoint or--

Pyle: No, this was like after 1969, when Vic had described this general architecture, the eight bit registers, the simple ALU, a couple of registers that referenced memory. And kind of a RISC [Reduced Instruction Set Computer] machine in some sense, because you had compute all of your addresses. And I from that did a bunch of thinking and noodling. I don't remember exactly when we ended up communicating this stuff to-- I'm pretty sure it was Stan Mazor at Intel, who was the application engineer at the time, who ended up writing a handwritten spec. I had xeroxed copies of that in that notebook, too. There was essentially a handwritten data sheet for the 8008, and it laid out these instructions. And they had morphed it and changed it somewhat, but we still had the kind of-- One of the things we did is we used octal instead of hex for our binary notation. So it kind of had two three-bit octal digits for the two registers in a lot of these instructions, and another two bits for the fundamental instruction type. And some of the instructions only had one register and used the other one to mean other things and stuff like

that. But we kind of had this thing kind of all laid out like that. So kind of the partitioning of the instructions in the instructions words formats were kind of done that way. But he also detailed the timing on the bus, a typical data sheet kind of thing, right? He folded all that instruction set stuff kind of into this data sheet thing that talked about the timing of the clocks on the bus and stuff like that.

Hendrie: Oh, all right. Well, I actually have found, I've identified where that handwritten spec is.

Pyle: Ah, okay.

Hendrie: Hal Feeney has it, who of course was the engineer that Intel put on the project to actually design the chip. Because that was his--

Pyle: His document.

Hendrie: That was his sort of spec as to what he was sort to do. And of course, Stan converted it from an architecture into a chip spec, which I mean you guys probably didn't know all the things <laughs> that the designer needed to be told to make it a chip.

Pyle: Yes.

Hendrie: But I haven't gotten it for the museum yet, but I would love to lay my hands on it. <laughter> He doesn't want to part with it yet.

Pyle: I don't know if there's any way-- I don't know if the documents had that went into this discovery process on that Intel patent dispute, or TI patent dispute, still even exist. If they got archived somewhere or they're, you know, after awhile they take all these archives and shred them or something after some number of years.

Hendrie: Yes, I hope they didn't do that.

Pyle: I don't know how we would be able to find those things, but anyway, yes. And you know, I seem to recall the process was I was kind of giving feedback to Vic, and I had these hand notes and stuff. And then I think Vic worked-- I think what happened, because I wasn't really there, Vic ended up working with Stan.

Hendrie: Yes, well you were in school.

Pyle: Yes. And they kind of ended up with the spec. So then I think I ended up reviewing the spec and looking at it and things like that. That's kind of really how the old original 8008 came to be. And I'm pretty sure we had some boards that had that chip actually running on it. Many of our machines-- the board

was about this big, and it had some card slots in it for the memory, and it was tucked in around, there's the tube, the CRT tube in here, and there's some space for the memory, and there was tape decks kind of had to go here, there was another card slots in here. <laughs> So the thing kind of-- we could lay it out on the desk. We had a big table we put this thing on and plug the memory slots into and wire up the power supplies to it, and had all the rest of the pieces kind of hanging out. And these were our basic initial machines we would use to get this stuff up and running.

Hendrie: Oh, wow.

Pyle: It's now very vague in my head when we kind of stopped doing that and started doing the discrete versions of the machine, and how much the discrete versions of the machine were trying to simulate the serial 8008. Because I know those original discrete versions were serial. So they could have very easily been totally simulating the 8008, just we didn't have chips to actually run in the board at the time.

Hendrie: Yes. That would have made a lot of sense to be simulating it, so you could do some software work and have it run. <laughter>

Pyle: Right. One of the first programs we wrote was a one-arm bandit we called it. Which I don't if some of the other guys were talking about this, but basically we would use a tight timing loop to measure when you hit a key on the keyboard, and use that as a random number generator. And each time you would hit one of these things we would choose one of the cherries and, you know, nuts and things like that. And we would put these three things on the screen and determine if you had won the jackpot or not. <laughter> So that was one of our original test programs for the machine.

Hendrie: I love it. That's great. Oh, wow.

Pyle: Very interactive program.

Hendrie: Yes. Wouldn't it be great if there was a videotape of that, that somebody had taken?

Pyle: Oh, yes. I don't even know if we had videotapes back then. <laughs>

Hendrie: Yes, probably not. Do you have any-- Was all the documents or things, you know, historical stuff that you had, was that all sucked up into that?

Pyle: Not all of it. You know, I should have thought about this because I knew you were coming, was -- I wonder if I have a document or two sitting in a box in a cabinet somewhere. I may have some documents on the DATABUS and the CTOS stuff, but I don't recall. I'll have to go look.

Hendrie: Would you mind looking?

Pyle: Oh yes, sure.

Hendrie: And I will try to follow up with you, because if you have any of them and you'd be willing to, you know, put them into the museum collection.

Pyle: Oh yes.

Hendrie: And of course they'll actually last for ever. <laughter>

Pyle: Better place than sitting in the same box. That's right.

Hendrie: Because we keep the originals, but we'll also scan them. So that, you know, building a digital archive of the history of computers. <laughter> All right, well thank you very much for your time, Harry. I really appreciate you doing this.

Pyle: This has been my pleasure.

Hendrie: All right, thank you.

END OF INTERVIEW

