An interview with

C. W. “BILL” GEAR

Conducted by Thomas Haigh

On

September 17-19, 2005

In Princeton, New Jersey

Interview conducted by the Society for Industrial and Applied Mathematics, as part of grant # DE-FG02-01ER25547 awarded by the US Department of Energy.

Transcript and original tapes donated to the Computer History Museum by the Society for Industrial and Applied Mathematics

ABSTRACT

Numerical analyst Bill Gear discusses his entire career to date. Born in London in 1935, Gear studied mathematics at Peterhouse, Cambridge before winning a fellowship for graduate study in America. He received a Ph.D. in mathematics from the University of Illinois, Urbana-Champaign in 1960, under the direction of Abraham Taub. During this time Gear worked in the Digital Computer Laboratory, collaborating with Don Gillies on the design of ILLIAC II. He discusses in detail the operation of ILLIAC, its applications, and the work of the lab (including his relationships with fellow graduate student Gene Golub and visitor William Kahan). On graduation, Gear returned to England for two years, where he worked at IBM's laboratory in Hurlsley on computer architecture projects, serving as a representative to the SPREAD committee charged with devising what became the System 360 architecture. Gear then returned to Urbana-Champaign, where he remained until 1990, serving as a professor of computer science and applied mathematics and, from 1985 onward, as head of the computer science department. He supervised twenty two Ph.D. students, including Linda Petzold. Gear is best known for his work on the solution of stiff ordinary differential equations, and he discusses the origins of his interests in this area, the creation of his DIFSUB routine, and the impact of his book "Numerical Initial Value Problems in Ordinary Differential Equations" (1971). He also explores his work in other areas, including seminal contributions to the analysis of differential algebraic equations and work in the 1970s an ambitious system for generalized network simulation. Gear wrote several successful textbooks and was active in ACM's SIGNUM and SIGRAPH, served as SIAM President and a member of the ACM Council as well as a number of blue ribbon panels and scholarly societies. In 1990 Gear left Illinois, to become one of the founding vice presidents of NEC's new Research Institute in Princeton, New Jersey. From 1992 until his retirement in 2000 he was president of the institute, and he discusses his work there, the institute's place within NEC, and its accomplishments and problems.

HAIGH: Thank you very much for agreeing to take part in the interview. It's always nice to come to Princeton. I wonder if I could begin by asking you, in general terms, about your family background.

GEAR: Well, I was born in 1935 in the period before the war in a suburb of London. My father was very much working-class. I think by that time he may have been a foreman of the operation he was running. He had left school I think at fourteen, which was typical of his class in those days. His grandfather left school I believe at twelve or ten (I can't remember) and gone to work as a laborer in a factory. I think there are two important things about my father. One was I think he had a tremendous moral-ethical sense about certain things, which he pushed on me, about the need to treat people properly. The other thing that he was always hammering into me was how I had to get an education—he said I had to get it because he didn't want me to work like he had to work though his life. That was odd because, when I finally became a faculty member at Illinois and my father came over and visited, he couldn't understand why I seemed to work all the time—he thought the reason to get an education was that someone then took it easy. Of course that's the great thing about working at the university: if you've got a job you really like, it's not work at all. It's the thing you probably enjoy most of all.

Now back to my early background. We lived in a suburb of London built in the early 1930s to house the growing mass of people moving into the London area. I went to the local primary school in that area. Of course, in those days England was unbelievably homogeneous: virtually all white, virtually all English. Immigration hadn't started yet. It was in the period leading up to the war. In fact, I had only been in local primary school for two weeks when war was declared. I was sent to my grandparents for the first of two trips and spent a year there.

HAIGH: And where did they live?

GEAR: About 40 miles north of London. Of course in those days, that was a long distance. It was relatively safe in the country from bombing. Then they sent me there again when the V2s were coming in, late 1944, I guess. Other than that I went to the local school and behaved like any typical working-class kid: playing most of the time, avoiding most schoolwork. In those days, I loved mathematics (of course it was called "arithmetic" in those days). I remember sitting in classes doing things like doubling a number as often as I could to see how big it would get. I can remember when I was five I decided to see how high one could count, so every night I would start counting higher and higher, and then I would ask my mother to remember the number so that I could continue from there the next night. I guess I didn't believe yet in infinity. I'm sure she just gave me an arbitrary number each night [laughs]. That's better than counting sheep to get to sleep.

So it was an uneventful childhood. My father was working class and I suspect money was somewhat limited. We weren't poor. Even though it was wartime and there was rationing, there was sufficient food for everybody. It may not have been a great diet. So I didn't see it as a period of privation, although I remember the night when ice cream became available when I was ten years old. And when candy went off ration, I had terrible teeth for a couple of weeks from eating too much candy.

HAIGH: Other than things you've already mentioned, are you aware of any particular influences that this background had on the way you approached your later career?

GEAR: Well, I think that in the primary school, we probably had one very good teacher in my last year. Let me say a little bit about the educational system in England then, because my father was pushing me very hard in getting an education. He was really worried because at that time, the career path for a normal person of my background was to go to primary school and then to secondary school at age eleven-and-a-bit, which basically prepared you for a blue-collar job. There were a few scholarship positions open to the local grammar schools, which -- although they were state-funded -- most of the positions were fee-paying and were beyond the means of people like my parents. So my father was very concerned that I should try to get a scholarship there, and I think he was pretty pessimistic that I'd do it because I was only interested in mathematics and terrible in other subjects. I remember he used to drill me on general knowledge questions in the evenings to try and get me to learn some stuff.

I was very fortunate because in 1945 after the war, the Labor government won and Churchill and the Tories were booted out . I was too young then—I was ten, so I don't know what was behind all of that. But the Labor government revolutionized many things in the educational system. They made all the positions scholarship; in other words, they got rid of the fees altogether and made them all the subject of what became the "eleven plus" exam. I was part of the first group to take that. So there were many more positions. As I recall, there were three parts to the exam that I took: one was mathematics, which was almost certainly arithmetic which I'm sure I probably aced because I really liked that; one was some sort of English; and the third one was labeled as some sort of general knowledge, my father was desperate because he drilled me on these facts of English history and all sorts of stuff every night and I would never remember it. But in fact it was an IQ test, and most of those things are pretty easy for a mathematician type. So I probably did very well on two of them, and that probably got me over the hurdle.

HAIGH: They would call those IQ tests "verbal reasoning," for some reason.

GEAR: Oh, okay, I don't know. I just remember that instead of questions of fact what I did was little puzzles. So that got me to Harrow County school, the grammar school—not the famous Harrow, but down the hill from it in the cheaper part of town. But it was the best local grammar school. From there things went on up. My ambition at that age was probably to be something like an electrician, because I loved technology, I loved taking old radios apart and trying to rebuild them in some way, I loved mechanical things. In England we had something called a Meccano set, which was like an Erector set but better. I think it still exists; I don't know. People don't seem to play with those things anymore as kids. So I used to build all sorts of devices. I remember I had this train system from an uncle who worked at some factory that made trains and had given me at some point, and I built automatic signaling systems and all sorts of things out of the Meccano and electrical wire and stuff like that. So I assumed that I was going to be an electrician. The thought of higher education—you know, I was the first in my family to go to college. I was the first to finish high school. So it was just beyond my ken. My father was pushing all the time. I think his view was that when you went to grammar school, then you were on track to become a white-collar worker, which to him, meant the life of Riley, I guess—maybe that's not the term the English would use.

When I got there, I think probably the teachers tried to push on me the idea of college, which was sort of foreign. English schools were very competitive. I don't know how they are now; you probably know that. In our school we were graded so that after each term, you got a grade that said you were say second out of 35 people in mathematics, or you were 34th out of 35 in history (which would have been a high mark for me in history). With those sorts of things I was a

competitive type, so I always wanted to be first in mathematics -- which I actually did -- or things like physics and chemistry, which I loved. I think the masters must have started putting in me the idea of college.

Then I remember at one point my father was very hot on the idea of my taking a commission in either the Air Force or the Navy, because they were offering programs that would send you to college, in fact at that time to Cambridge for three years to get a degree, and then take a commission. He thought that sounded wonderful, because then with a commission you retired at something like age 42, and you could take another job and live comfortably and not do much work. I also think that, probably, part of the thing there was the financial issue. I think he was very aware that if I tried to go to college, it was going to be a big financial burden. And it would have been, but some time in that period, the Labor government also introduced the state scholarship system. By the time I went, I got a state scholarship-- and a college scholarship that was just deducted from the state scholarship -- it didn't change the amount of money that paid *all* of my expenses. I don't mean "all of my expenses" in the American tradition, which leaves you working to try to eat. It paid my room and board and clothes and books. By working a little in the summer and Christmas break—Christmas break I'd usually go out and get a job at the post office when they wanted extra people, and in the summer I'd find some sort of job. I even paid my parents some for my lodging at home when I stayed there in the summer. So I owe a great debt of gratitude to the English government for putting me through the university and paying it all.

HAIGH: So you would have done O levels and then A levels at your grammar school?

GEAR: And then there was something called S levels in those days—scholarship levels. I don't know if that's still around.

HAIGH: Yes, I took some S levels as additional paper you could sit at the same time as the A level classes, which would cover the same factual knowledge but require deeper analytical skills.

GEAR: I don't recall what it was, but I went to Harrow. One thing interesting about that was it skipped the first year and basically we went directly into second form. In the English system, when you went from 11-plus into the next secondary school, they start at first form. So it [Harrow County School] pushed people on harder; I was a year ahead. In the English system normally you went through fifth form and then that was where you took O levels, and then that was the time where you could get out and get a white-collar job or something like a technician. If you want to stay on at a university, you went on two more years in sixth form and took A levels. In order to get into Cambridge it really was necessary to stay an extra year in the scholarship sixth. Amongst other things I had to learn Latin, which I'd never paid attention to earlier, because I had to pass a Latin exam in order to get into Cambridge. So I stayed an extra year in high school in order to take the Cambridge entrance exam, scholarship exams and get my Latin and stuff. But I was still not a year behind, because I had already jumped a year.

HAIGH: So we should say, for the benefit of American listeners, that the system would usually be to take a broad number of subjects for your O level, taken when you were 15 or 16, and then most people hoping to go on to University would take three subjects for two years for A levels--

GEAR: I believe I took four. As I recall, I took pure math, applied math, physics, and chemistry.

HAIGH: Was it a great relief to be able to drop all the other subjects?

GEAR: Yes, I think we had one required course in English literature. As I say, in the last year I had to take Latin. I also took engineering drawing; I was interested in it. In fact, I think I took other levels in it, as I recall. I was taking A levels in those other subjects. I may have also taken

biology, but I don't really remember; I remember taking some biology classes, but I wasn't that interested. In those days, biology was… I don't want to say "not a very scientific subject," but it hadn't developed the analytical structure that it has today. I think if I went through it today I would almost certainly go into microbiology as some of the most interesting thing are going on there.

HAIGH: In those days it would have been all memorizing the names of bones and parts of flowers and things?

GEAR: Right, right.

HAIGH: So, from your grammar school, was applying to Cambridge something that a large proportion of the students would have done?

GEAR: No, I think I was probably-- it was a fairly large grammar school, and it doesn't exist anymore. It was changed into…are there junior schools in England now, I think? It was changed into a junior school after some reorganization. But as I recall, it had four forms at each level: A, B, C, and D. They meant what they mean in American, basically. Each was probably 30 or 35 big, so the total class in a year was I assume in the 150 range. By the time I got to the sixth form it probably halved, I would say, and there may have been 70-some people, possibly, college-bound. I think only two or three from that school went to Cambridge. Well, I know at least one did. My friend there also went to the same college. I think that's the only one I recall.

I don't think any of the masters, any of the teachers that I had had been to Oxford or Cambridge. They were pushing me to look at it, but I could get no advice on where to go; they didn't really know. I applied to both Oxford and Cambridge, and the teacher source seemed to indicate that Cambridge was probably better for science. I think that was certainly a popular view then, and it was probably true. Then as to what I did, I was really torn between electrical engineering and mathematics. I liked both of them a lot. And the math teacher was pushing me to go into mathematics and the physics teacher was pushing me to go into electrical engineering. I have no idea why I decided. I don't remember.

I remember when I went out to interview at one of the Cambridge colleges, and I think this topic must have come up. The guy said something that for a long time I believed in, though I question it now. He said, "Does it matter how you get there? What matters is where you get." In some way I thought for a long time that that was right. That's perhaps been my attitude all along to what I do: what's important is the outcome. Well, it's important if you do things ethically or not, certainly like that, but not in terms of whether one approaches it from an engineering point of view or a mathematics point of view or any point of view. It's what you finally come up with that is important. I run into a lot of people at NEC, we'd stop and somebody would say, "Oh, that's not *physics*!" "Well so what?" was my response. I don't care what it is; I care about whether it's good. That was my viewpoint. What I realize now, when I'm sort of at the end of my career, I say, "Wait a minute, I wasn't going anywhere. It was the path that was the fun." So maybe it does matter which way you go. In the end, there is no *there* there when you get there. It's what you do along the way, so I don't know: I sometimes wonder if I had thought more about this, I would have done something different. But I enjoyed it, anyway.

HAIGH: So you would apply to a specific college, would you not, rather than to the university as a whole?

GEAR: Well, in those days, the exams were to groups of colleges, three or four or five, I forget, and you went up and took the exam in Cambridge. So I think I applied to more than one group

and got some nasty comments from one college: “Why was I doing that?” And the answer was, of course—I did the same at Oxford—I was desperate to try to get into one of them. I think their attitude was that you should pick the one you want and just go there. Maybe if one felt certain you’d get in, that would be the right thing to do. I had no idea what the various colleges represented. I don’t know why I chose Peterhouse. I am glad that I did. It was the smallest and the oldest college. Maybe I chose it because of the *oldest*; I have no idea why. There was nothing about the colleges that I recall that said, okay, if you’re interested, particularly in mathematics, you go to this one. They had reputations for sports—if you wanted to row, you should go to Lady Margaret, was it?

HAIGH: So, academically speaking, pretty much at random?

GEAR: Yes…as most of my life has been!

HAIGH: When you got there, how did Cambridge compare with your expectations?

GEAR: Well, first of all, I was pretty much overwhelmed: a kid coming in from a pretty modest background. Most of the people at Cambridge were there because their father, or their uncle, or their grandfather, or all three had been there. One of my close friends—a very nice guy, and very nice to me—his uncle, his father, his grandfather had all been Peterhouse graduates. It only had 200 students… And there was plenty of money. I was probably much more sensitive of this situation than they were aware of it as being even an issue. I remember going off for some weekends to some fancy places that were totally out of my realm of imagination in those days. I’d go in and the guys had just come in from some jolly good otter hunting. We went there for a party, and it was very nice. So I was treated very well, but I felt a little overwhelmed by it all from the social point of view. But I had a good time.

We had a lot of freedom. The educational system at Cambridge in those days (and I assume the other universities, and it may still be that way) was the math tripos, which was pretty much fixed through the whole time. I think there was one option. There were lectures you could attend or not attend, as you wished. That was several hours. I forget how many lectures I would go to a week. Maybe three hours a day typically; maybe slightly less. There was no homework. There were no exams from the lectures. There was one group of six three-hour exams at the end of each academic year, based on this material. So everybody was studying the same thing. I think in the second and third years there was one optional paper; you either took applied mathematics or natural philosophy, I think it was called, which was mechanics and electricity and stuff like that. I don’t remember now, I think I might have taken that one. But other than that, it was pretty fixed. You had weekly sessions with a supervisor that changed each term—the person changed each time. You’d go to, say, one term in geometry and another term in analysis, and he would assign you little things to do to bring back in, though they weren’t graded. You could never show up to lectures and nobody would know. You could probably blow off the meeting with the supervisor, though that probably would have been reported to the college.

HAIGH: So there was no formal registration for particular courses; you could just choose which questions on the final paper to answer?

GEAR: Right. You were told what you took. And in those days, the Cambridge tripos, meaning three paths—there were parts I, II, and III, but the standard math program was Part I and Part II. Part II occupied two years, and that was a degree program. But if you went in with sufficient background, you could skip Part I and do Part II right away. Now I said that there were a lot of people there who came from “good public schools,” but were there more from family

connections who took Part I. The preparation that I had, I was able to skip Part I right away. It was basically stuff I had done in high school. So in the second year I had finished what was needed for the degree, and then Part III was actually the collection of all the graduate classes, which you just selected something. So I took some of those, though most of the third year I rowed (crew) and blew off my course work.

We had a very successful boat club in college. I actually rowed a little bit in high school, only because I hated rugby football, and this was the only way I could get out of it. We had a head master who was very much into sports and the ROTC equivalent. It was an all boys' high school, of course. "We're gonna make men of these boys!" I hated things like that. So rowing was pretty good. It meant that we got to go into London to the Thames one day a week to do the rowing and leave school early.

HAIGH: And get your teeth knocked out less.

GEAR: Right. And so when I went up to Cambridge it turned out that the whole of the Peterhouse first boat had been seniors, and so had left the previous year. So they had nobody in the first boat—the college was very small and they only fielded three boats. The first boat now was entirely freshmen. So I got in it. Pretty much the same boat was still rowing in the third year, and because just by virtue of being together the whole time and rowing regularly, we were extremely successful. The college won "Head of the River" for the first time in 108 years. The college gave us a party in which they provided something like 30 barrels of beer.

HAIGH: Now at Cambridge, I believe, the terms are only eight weeks long.

GEAR: They're very short.

HAIGH: How did you have time to study and do anything else?

GEAR: The University's idea was that you shouldn't work during the break time. Indeed they really frowned on working, as I did, in the winters and Christmas break and in the summer.

HAIGH: Oh, I don't mean like paid work, I mean just--

GEAR: No, no, they frowned on you doing paid work, because they expected you to be educating yourself this whole time. The lectures were there to excite you to go out and learn stuff, and so on. It was less education than exposure, I guess; you were expected to do things on your own.

Cambridge published the exams from the previous years—the exams were done university-wide and they were published. The structure of the exams didn't change much from year to year—the same types of questions. They would vary, but there was always a question that relied on memorizing what was so-and-so's theorem, and then a much tougher analytical question where you had to use it to solve some problem. There were ten questions in each three-hour exam. It was well known that you could get a third-class degree, the lowest in the standard level, for completing just one question. Completing four would almost certainly guarantee you the top place. I think you were told not to try more than four or six. So you would look through and pick four, and then you worked like hell on these questions. Practice on this type of question is very important. The way I studied for the major final at the end of part two, which was going to determine the type of degree, was that I purchased the last ten years of exams and went through and did all of the ten questions on each exam as my method of study. It's partly learning the tricks of the exams.

HAIGH: Did you get a first?

GEAR: Yes.

HAIGH: So let's just talk a little bit about the state of the mathematics curriculum at that point. Did it include any of the kinds of applied and numerical things that you would later work on?

GEAR: Let me take the first two years, which was the standard degree program. There was no numerical stuff in there at all; that was very new. I would say the Cambridge math program was extremely classical at that time. There was some applied mathematics, so I don't feel it had this disdain for applied mathematics, you know the Halmos "applied mathematics is bad mathematics" sort of viewpoint. It was probably a little old-fashioned, too. It lacked some things. When I came to Illinois as a grad student, the advisors assumed, "Oh, you're from Cambridge; you must have a tremendous background." Well, I was missing a whole bunch of "modern" theory—I had no measure theory. And I was totally lost in the class on probability theory, which had assumed you had taken the basic measure theory course. So for the first half of the first semester, I was in desperate shape trying to figure out what was going on because of the lack of that type of background at Cambridge. Otherwise it was pretty standard mathematics, probably a lot of it still from the 19th century. But it was good and it was solid.

There was, of course, the computer laboratory in Cambridge at that time that had the EDSAC. It occurred to me much later that possibly I could have somehow gotten involved with that. But there were no courses, no option in courses, even in part three, the third year, where I took a numerical analysis course from a guy named J.C.P. Miller, who was a very well-known numerical analyst, long since dead. He was associated with the computer laboratory. But, again, there was no suggestion that one could get near the computer. I don't know; maybe if I had been more pushy, which I didn't feel like doing it back then, I could have had something to do with it.

HAIGH: So as far as you know, none of your professors, and none of your fellow students, had anything to do with the computer laboratory either?

GEAR: The only person that I knew that had anything to do with the computer laboratory was the cox of our first boat, who was actually what in England was called a postgraduate student—that is, what we call a masters' student here—taking the program in the computing laboratory. In fact, at one time I was half-thinking that maybe that was what I would like to do.

Now I took this course from J.C.P. Miller on numerical analysis, and he successfully concealed from me what numerical analysis was…until I took another course at Illinois a couple of years later and figured it out. He stood in front of the class with a Friden calculator—it's one of these old things with wheels and a hand crank—and cranked out numbers and wrote them on the board and muttered away. It turned out later on I determined he was doing Gaussian Elimination and so on. But he sure didn't explain what he was doing and I was totally and utterly lost.

HAIGH: So your impression was that it was just a course in calculating machine operation?

GEAR: I wasn't certain what it was a course in. I don't think I learned anything useful in that class. There were a number of faculty who didn't seem to care much about getting stuff across to students. Another part three course I signed up for, which I thought would be interesting, was genetics. It was taught by a famous statistician. He was talking about genes and chromosomes. So after class one day, I said to him, "Could you tell me…what is a chromosome?" or gene, I don't recall which. I'm the sort of person who wants to understand a little bit about the thing I'm dealing with, rather than just dealing with an abstract subject. And his answer to me was, "When

you put butter on your toast in the morning, do you ask what butter is?" I stopped going to that class.

As I said, in the last year in Part III, I basically spent all the time rowing and blew it off. I didn't do very well on the part three exam; it didn't matter.

HAIGH: So, if you didn't need the Part III to get a first, was it really just to stop people getting too bored if they'd run out of…?

GEAR: I was thinking about possibly staying on for this computer thing. I was interested in finding out about computers, but didn't know much about them then. Had I had done that, I would have had to have gotten some financial support from somewhere, because my scholarship was ended with Part III. Probably what I should have done was to concentrate hard on Part III, because that would have let me have some financial support maybe. In retrospect, maybe I'm glad I didn't therefore concentrate hard on Part III, because it might have made a vast change in my life had I stayed there, and probably not for the better. But what happened, and we're sort of jumping here to the transition of moving to America, if you want to go there now--

HAIGH: Let's not quite jump there yet. Were there any faculty members there that you did form a particular connection with, get to work closely with, get to know better?

GEAR: Not in mathematics as such. The structure of Cambridge at that time was that we went to these massive lectures that were university-wide. I mean, the university program wasn't that big in terms of American universities. But there were typically two lectures on each subject at the same time. At nine o'clock there would be lectures, say, on analysis—two different lectures in two different rooms. Each college had a director of studies for each area, so there was a guy named Burkill who was director of studies for mathematics [in Peterhouse], and maybe one or two more junior people. I think there was someone called a fellow, which was kind of a very senior advisor to students in mathematics. So that about all there was in Peterhouse (as I recall) in mathematics. Let's say there were 200 students in Peterhouse, so there couldn't have been very many mathematicians, since they were spread over a lot of subjects, and the director would assign you to your supervisor. They would come from another college. The faculty would actually get paid for each of things that they did—for the lectures they gave and for the students they supervised, and so on. And they would tell you what lectures to go to. Again, the faculty was getting paid for these lectures. So there would be two lectures, say, in analysis being given at the same time. If it was well known that one was a better lecturer than the other, one lecture would probably have all the seats taken, say all 130 seats, and the other lecture would have five students in it -- typically the five students who happened to be in the college and advised by that director of studies, so they had to go to that one!

HAIGH: So then the lectures were very traditional: the lecturer would write at the blackboard and you would copy it down and, at the end of the hour, you would go away and never really talk to them.

GEAR: Yes. I occasionally asked questions after the lectures as we were walking out, and they were communicative that way. But not very. In fact, most of the time, I seem to recall, we would sit in the back of the lecture reading the *Times* or the *Guardian* and just keeping an eye on the thing. In fact, I remember once I went to one lecture. He must have given lectures on two different subjects that year or something, and I asked him several times questions. After one question he said to me, "This is a first. Most of the time when you ask me a question it's about

the lecture *other* than the one you're in." Then he went on to say, "I'm surprised you even knew what lecture you were in because you were reading the paper." But they were very pleasant.

The main way to study for me was doing the exam questions. I learned by writing the stuff down and going through it. I'm terrible at memorizing stuff. I have to be able to re-derive it. That's probably why I like mathematics, because it's a subject you can do that in. And why I hated history, because you can't very well re-derive dates of things very easily.

HAIGH: And were there any relationships with fellow students that you continued later?

GEAR: Yes, I pretty much lost contact with all the students. I roomed in my second year with a student who was from the same high school. I maintained contact with him for a while, for some time. I saw him about five years ago at a college dinner, but I don't maintain contact there anymore. There was no great overlap.

There was a guy I met in my first year at Cambridge and then came to Illinois. He first of all went to Israel, and then he went to England, then he came to Illinois and got a Ph.D. and stayed on the faculty. He was the person I had contact with for the longest time. He died last year; he was the same age I am. He had retired from Illinois. He was at Illinois most of the time that I was there. So he was the only one from Cambridge with whom I had maintained close contact. He was a good personal friend. We played squash together; we had dinner together a number of times. He had cancer, so near the end we had dinner together about a month before he died. Other than that, I've pretty much lost contact from not being there anymore. There was nobody doing stuff similar to what I was doing, so there was no reason for professional contacts.

HAIGH: So then your final year you were rowing a lot, enjoying life, and presumably thinking about what you do after graduation?

GEAR: Thinking some. My whole life has been characterized by not planning my career. Several times I've wondered, what would have happened had I had some knowledge and some help and thought about it seriously and done something? Who knows; maybe it would have been better, maybe it would have been worse.

In those days, one didn't typically get a Ph.D. in England; it was very unusual. I was starting to think that life as a Cambridge don might not be too bad. The one Cambridge faculty member that I knew fairly well was (I think) in history; we used to play bridge with him. He seemed to have a pretty nice lifestyle. I thought that this wouldn't be too bad, and I was thinking about that as a possible career path. I didn't know how to go about it. I was probably really thinking, well, I'll have to get a *job*. I did go on a couple of job interviews, which really put me off. "Well, yes, we keep a small stable of mathematicians for when we need one," type thing. It didn't sound very interesting.

And then, halfway through the third year, my roommate at the time (who was in some other science, I forget now what) was told by his tutor (the tutor was the person in charge of your general well-being, and got reports if you were out too late too often) was giving him information about some English-speaking fellowships to the U.S. And he wasn't interested, but it looked interesting to me. They paid money for one year to come to the U.S. as some sort of "unofficial ambassador". You know all that other nonsense that was very popular at the time. So I applied but I didn't apply myself very rigorously to the process. I filled out the form. You know, where do you want to go? Harvard—is there any place else in America? I went to the interview in London. It was rather early-ish in the morning. I had been to a fantastic party the night before and was not interested in talking to anybody in the interview. I really blew that off.

But for some reason, the proverbial little-old-lady secretary who was there and running interviews and sending people in seemed to take a pity on me or something; she said, “You know, in past years, Johnson Foundation (which is S. C. Johnson, the wax company in Wisconsin) paid money to the English-speaking union for fellowships. This year they’re running their own.” They gave one fellowship in each country in which they had plants, and they had a plant in England. So she said, “There’s one for that coming up. Why don’t you apply to that?” And she also said, “You know, everybody puts down Harvard for no reason at all, other than that they’ve heard of it. You should put down a place and have a good reason for it.” That was excellent advice. I went to the mathematical laboratory and spoke to a guy named Sandy Douglas, who was there at the time. It turned out that, at that time, Cambridge and Illinois had a big cooperation; several faculty members, including Wheeler and Douglas, had visited Illinois for periods of time. Illinois was one of the few places, as I mentioned, with a computer.

HAIGH: Now had you known Wheeler at this point?

GEAR: I didn’t know Wheeler at all then. The only person I met was Douglas. And so he mentioned Illinois, so I got some more information on that. I wrote it down with my reasons, and I also got a good night’s sleep before the interview, and I also boned up on the stuff in the papers from the last few days, because it seems they like to know that you’re interested in current affairs. Whether I was interested or not I decided this was not the time to say I didn’t give a damn about this stuff (which was probably my attitude at the time). I eventually got that [fellowship]. And the English-speaking union fellowships came with a Fulbright in effect. I think in those days, Fulbright's coming this way only paid travel (I’m not positive of that).

HAIGH: They have two kinds: one is for a year’s maintenance and travel; and they also have another only for travel, when the people have funding elsewhere. I had the kind that gave the money as well.

GEAR: Oh, okay. I didn’t know that. I probably hadn’t even heard of the name and I got the Fulbright.

[Tape 1, Side B]

HAIGH: Now, I think that you had just implied that one of the attractions of Illinois was that they had a computer there.

GEAR: Yes, I was interested in computers by then, although I knew nothing about them. In fact, what I thought I knew was probably totally wrong.

HAIGH: So what did you know…had you read articles in the popular press?

GEAR: Probably. The whole issue of automation-- I remember saying that I was interested in automation on my study application, too. This whole issue was receiving a lot of press in those days and was a very hot topic. One of the things I had done in my summer break between my second and third years write an essay for a college or university prize on something cyber-related. Cybernetics and so on was a big name around then. So I was very interested in all these things and tried to find out more about them.

HAIGH: Yes. And I think in that period, late 1950s, early ’60s, there was very much thought that was going together, automation turned into cybernetics.

GEAR: Right. So this fit in with exactly the thing that I liked to do as a kid: building mechanical things, controlling them electrically. If I’d have had the sources of computer stuff that we have

today as a kid, God knows what I could have built! Actually it's probably all totally boring now because it's all software.

I always repaired my own car, as did Vel Kahan, by the way. He probably gave you a long lecture on it. He put his electronic ignition in his car very early on, I remember, and he explained [it] at great length to me.

HAIGH: So, that was how you chose the university, you won the fellowship. Now, would that Johnson Foundation fellowship would have been tenable anywhere, was that the case? Technically, you could have used the Johnson Foundation fellowship with any university?

GEAR: Well, I could have *applied* to any university. I don't know, once I put my application in and had gotten the grant where I could have said, "Okay, I'm going someplace else now." But that wasn't even a consideration in my mind. Probably not. I think they sent the money to Illinois. In fact I'm sure they did because, what I recall was, I left England and, in those days, there were severe currency restrictions. Besides, I had no money. I left England with £25 in my wallet, and that was my total finances.

I first of all went to New York on the boat Queen Mary I, I think it was because it was cheaper than flying in those days. It was a great time, too. There were all the students going over; we were all, of course, in steerage class—we partied every night. I spent a few days in New York with a friend of some friends of my parents, and then took the train on to Chicago. It turned out that it was exactly an hour late, and I didn't understand about the different time zones. Furthermore, in those days, the trains ran on Standard Time, even in the summer. So we were coming in and I had a close connection where I was supposed to get off, as I recall, at some earlier stop than downtown Chicago, jump on the Rock Island Line (and what a fancy name that sounded to me; there was a great song in those days—*The Rock Island Line*) and take the train to what I know now to be the 63rd Street Station on the Illinois Central line and go on down.

I thought something was wrong with the time, and I wasn't certain. I asked the conductor what time it was; he gave me a number and I asked him, "What *sort* of time is that?" And he looked at me and walked away quickly. So I got off at the station and it was totally deserted and I didn't know what to do. I went outside to see what I could do, and a taxi driver offered to drive me down to Champaign for $25, but that was all I had. This was Labor Day weekend, too, on Friday. So I had no money and decided that I couldn't do that. So then I rung the Institute for International Education, I think it was called. It may still exist. They sent me some information. They had an office in Chicago and I recalled this, and managed to find the phone number. I called them and told them what had happened, and that I missed my train and had to wait until evening till the next train. They told me how to get down to their office, where they gave me some information and set me on the train down to Champaign, when I had no idea what I was going to do since I had no money—the fellowship funds was given to the university and I got it later on. Actually, the guy from the local YMCA on campus who helped foreign students and had known about my coming in on the morning train and didn't show up, came down to the evening train and, by chance, I was there. It turned out that the Y had a couple of spare rooms, which were very cheap—$1.50 a night. He put me up there and let me wait on paying till I got enough money. So I survived.

HAIGH: What were your initial impressions of the university and the United States in general?

GEAR: New York was, I suppose, sort of what I expected. I had had all these warnings from a friend of a friend who had been over in England that I should be careful. When I got off the boat

and got a taxi to the place, I shouldn't say much in case they realize that I'm a foreigner and they take me for a long ride in the taxi. I had been told how unfriendly New Yorkers were and how awful it was going to be. And, in fact, it wasn't like that. New York was kind of wonderful. It was in August and for someone from England in those days, unbelievably hot. Very few places were air conditioned in 1956. All the subway lines were still independent: the IRT, the IND, the BM…whatever they were. I remember we asked some local person about how to get a subway somewhere, and they said, "Oh, you want to take the IRT; it's down there," and he pointed for us. So we started walking and we went into the wrong one; it was a different line. The guy came running after us after a block and pulled us back. We found they were very nice. And it was fascinating. On the boat we had run into a Methodist minister who was coming back, and he showed us around one day. He didn't so much take us to the standard tourist sites, like the Statue of Liberty, but took us out to Coney Island and such. It was a wonderful time.

Then I got to Illinois and it was bigger than anything I'd ever seen, less populated than I'd ever seen. In fact, when I came, I knew about the university because I looked up stuff, but I knew nothing about Illinois. I assumed it was a French name; I thought it was pronounced [the French pronunciation] "Illinwah." On the boat, the customs immigration officials come aboard before you get near the port, so they process everybody pretty much before you land. So I went through immigration and the guy was questioning me; he said, "Where are you going?" I said, "Illinwah." He didn't correct me, and he said, "Oh, that's the center of the country!" I had this vision of the hub of the wheel with all the spokes coming out; I didn't realize it was the hole in the center! There was very little there. But it turned out to be a great place to study. The computer group there was very good and very involved with people outside; there was a flow of visitors. It was very relaxed and small-town so that one could really get a lot of work done. I stayed there because I went back there later. It was a great place to bring up kids: low hassle, you could use the public schools. A lot of great colleagues.

I guess my big impression of the university from the point of view of mathematics and so on was, first of all, having come from Cambridge, where you were totally left on your own, to go into classes where in many cases they still required attendance and took attendance. The university rule at Illinois at the time was if you didn't show up for a class the day before or the day after a holiday like Thanksgiving, you flunked the course. And they had homework and stuff, which is unbelievable to me. I thought these kids were being treated like I hadn't been treated in the schools since I'd been 14 years old.

HAIGH: Did they have the GRE exam in those days?

GEAR: If they did I didn't have to take it. When I went to Illinois it was still, as most state universities were at the undergraduate level that anybody who was above the 50th percentile after high school could go to Illinois. I assume it was still partly financially limited for a lot of people because the student population was smaller. So the enrollment wasn't that big, but there were some extra-smart people there -- and some real dodos there.

I remember the first year I lived in this thing called the Cosmopolitan Fraternity, which was mostly foreign students. The guy I was rooming with was an American from a Chicago suburb. Pretty nice guy, except he was the chairman of the student Republican Party, which wasn't exactly my political cup of tea. But otherwise we enjoyed each other. He had a brother who had just come in who was a couple of years younger than him who was having a terrible time and I agreed to try to tutor him in mathematics to help him try and get through. Well, I worked with this guy and he didn't have a hope in hell (I shouldn't say this…!). I finally degenerated into just

trying to get some rote memorization of a few things into his head on things like what a log of one and the log of ten were. He didn't have any concepts. Some of the undergraduate classes were very weak, but there were some very good graduate classes and not too many weak students in them. And the mathematics was very pure, as I keep repeating.

HAIGH: How large was the graduate student population in the mathematics department? Were you in classes of ten people, or did the big classes have fifty people in them?

GEAR: Since I was initially in math, I guess, I don't have any sense because we never came together as a group. One of the big problems, and I think it's still true in a lot of U.S. universities, is if you're on a fellowship, particularly if you're on fellowship money from an outside university, you have no reason to have any direct contact with the infrastructure of the university—faculty, research, and so on—except to the extent that you would go to your advisor once a semester. So that was my contact with the math department and sitting in classes and not knowing much about anything. In a research assistantship and, possibly a teaching assistantship, you're more plugged in to what's going on. You have to do things to meet with people all the time. So I was very fortunate that I got involved with a research assistantship very quickly. For that reason I generally did not recommend people got fellowships later on. I don't think they're a good idea. I don't know how big the group was in math; like I said, I never saw them. It was a fair-sized math department, so I imagine there were quite a lot. It's probably closer to fifty than ten, but I don't know. The graduate classes I took had numbers ranging from 10 to 25.

Even then, by the way, Illinois had a big foreign student population. The biggest group at the time was Indians. Later on it became Chinese and Asian countries, but there were quite a lot of Indians in my classes back then. As usual, many of the foreign students—maybe the English excepted—were among the hardest workers. Although it seemed to me that in a lot of other countries there's a heavy emphasis on rote and repetition.

I remember that I took a class in probability theory, the one I struggled with because I didn't have measure theory, from [Joseph Leo] Doob (he's now dead). He's a very well known person, responsible for martingale theory. In fact there's a story about him I mentioned, possibly apocryphal but quite likely true, that he wrote a very important book about this stuff, and it was translated into Russian. At some point some agency of the government translated it back to English and classified it. I don't know whether it's true, but it certainly sounds possible.

About that time, the "new math" was being developed at Illinois, which propagated disaster on the U.S. mathematics curriculum for many years. So it was being taught, of course, at the nearby high school. And one day Doob was at home, possibly at dinner or something, his daughter asked him a question and he responded. She said something like, "No, Daddy, that's not a number; that's a something else." They were very heavy on using the right names. And he was absolutely furious, so the story goes, and called up the school and demanded to know what on Earth they were teaching his daughter. I don't know how true those were, but they were stories that went around.

So I took Doob's class and his final exam was on Saturday afternoon, I remember. I don't think they dare have final exams on Saturday afternoon anymore, but they did in those days. I think we even had Saturday morning classes, come to think of it. It wasn't a very big class, so there was all this very complex stuff we had done on martingale theory, and he was also interested in probability stuff in general. So he came and he wrote three or four questions on the board. And the first two were these very simple probability things that you would have to think through very

carefully: what's the probability that *x* does this-- what's the probability of the secretary going to blah-blah-blah…. The types of things that are easy to get wrong if you don't have a grasp of what probability means. There were a couple of questions like that that required some careful thinking, but nothing very complex.

Then the last question was something like, "Write everything that you think is important about martingale theory. And when you're done, just slide your exam completely under my [office] door." Now it was a three-hour exam, so I wrote until the three hours were up and left. And I was told that some people in that room were still there at midnight—they went on and on with everything they could think of. These were the people that came from those educational systems in some other countries where there's this tremendous emphasis on just being able to regurgitate this stuff that you've learned without any attempt to condense it, crystallize it into something. That was always the problem I had with grad students from some countries, to change their viewpoint. I remember one time I took on a grad student who aced a numerical analysis qualifying exam. I thought this guy is going to be great! But it turned out that he could reproduce almost any page in many textbooks, but I don't think he understood one word of it, and so it was very hard to get anything new done.

HAIGH: Returning to your experiences there: what was the mathematics curriculum like in those days? Also was fellowship was good for the whole PhD, so you knew from the beginning that you'd be staying…

GEAR: No, actually, I came over with a one-year fellowship, only intending to stay a year. I was going to a master's.

HAIGH: Okay. So tell that story.

GEAR: And so any eight courses, basically—no thesis, no nothing. So I just took courses I saw that I was interested in. I took the only two computer courses in the university; one was programming and the other was logic and design. They were both senior/graduate student courses, and they were both taught by faculty in the digital computer laboratory, which was the place that housed the computer in those days. There was a small group of faculty, maybe a half a dozen all together in that. And that was where my interest really lay. I took an applied math course, Mathematical Methods of Applied Mathematics or something like that, and a course in probability theory. That was probably in the first semester, and then another four courses in like style—I didn't take any pure stuff. My game plan at that time was to return at the end of the year. One of the reasons that I think I said that I had taken this was that it was a chance to see America, and a delay of the decision on what I did for a job for the rest of my life for another year.

Well, once I started working with computers I knew that this was what I wanted to do. There was no doubt about it. This combined all things that I liked: some amount of mathematical analysis, the mechanical stuff of programming, the invention, the design. I'm a cross between a scientist and an engineer. I love the inventiveness of algorithms. You're really creating something new. It's more engineering than it is mathematics and science, in the traditional sense. So I realized that this was really what I wanted to do, and it was clear that the place to do it in those days was the U.S., not England. And the third thing was, if you're going to do it in the U.S., you had better get a Ph.D. It wasn't a flash of inspiration; it just evolved that way.

Part way through the first semester, the guy teaching me the programming class, David Muller, asked me if I wanted a job as a research assistant programming the ILLIAC I. So I checked with

the Johnson Foundation, since they were paying the fellowship to be there, assuming that they would say, “No, no, we will take your fellowship back.” But they didn’t seem to object to me double dipping. I think actually that I only took a quarter time assistantship for the spring semester, because I still wanted to carry the four courses. I think I was probably still thinking at that time that I was going to have to go back and I needed to get the four courses. But by the end of that second semester, when I was doing programming, I knew I wanted to stay on. I got offered a full-time research assistantship in the summer and a half time during the year, and I was committed to getting a Ph.D.

HAIGH: All right. So talk, then, in more detail about that first exposure to computing. Now you’ve said that, earlier, you had a general awareness of computers, so you had an idea that it would be some kind of exciting, cybernetic, automating engine. I think you also said that you had signed up for a programming course. So what kinds of things were covered in that first course?

GEAR: Well, this was basically machine language programming for the ILLIAC I. In those days, programming was pretty high-powered; nowadays it looks so trivial. So it was just the concepts of programming, because there were very few real concepts except how to do it, and tricks to doing it, basically. The ILLIAC I was an unbelievably primitive machine: it had no floating point, no index registers, very limited memory, it was very slow. Much, much less powerful than the first IBM PC.

HAIGH: So all the students would write their own programs and run them on the machine as part of the class?

GEAR: Right.

HAIGH: And can you remember, was it a small class?

GEAR: Well, the interesting thing about Illinois computer group-- maybe we should say a little bit about how it got there. In 1949 they had set up this committee and then decided to make a copy of the von Neumann machine. They initially built the ORDVAC, which was delivered to the Aberdeen proving grounds, and then the ILLIAC I. And it was one of the few universities with a machine. So there were a large number of people coming in from other universities to use it; there was a lot of interest on campus in using it. It had a large software library for a computer in those days because of all these people coming in. There were also copies made of it, by the way. University of Iowa had a copy; the University of Sydney, Australia, had a copy that they called the SILLIAC. There were very few other universities that had them. So there were probably a lot of people in that class, because there was interest, I think, not just math students but people who wanted to use the machine for research. So I seem to recall quite a number of people—more than 20, but I don’t remember the numbers in the class. I assume it met three times a week.

In those days there was really no operating system. There were a thousand words that were 40 bits long, so there was about 5,000 bytes of primary memory, and about 60,000 bytes on a drum, which was a forerunner of the disk. There was no operating system to speak of, but a way of loading in very primitive code from tape. It was all paper-tape input, and the way you debugged the code is you went in at code check time, which I think was from noon till one, and again from five till six p.m. There was a line of seats in the computer room and you sat in your chair and you waited your turn. When your turn came up you had a maximum of five minutes. Either the operator would put the tape in for you or, if you knew what you were doing, you put it in

yourself. The machine stopped when it hit an error, and you would write down what was displayed in the registers in binary. You had the option, if you had time, of changing the registers (it was a vacuum-tube machine, of course) and there were some contacts on the front panels. If you damped your finger and touched them, the capacitance would change, so you could reset the control counter or the accumulator and start it up again to try and get something else. And then you went away and looked what you had done, changed your tapes (which meant copying them), and came back later. It took a long time to debug a code.

HAIGH: But, despite the cumbersomeness of that process, you found that programming was something that you enjoyed a lot personally.

GEAR: Yes, I've always loved programming. I mean I can get totally absorbed in it where I lose all sense of time and can't think of anything else. That's what I'm doing with my life more than anything else right now.

HAIGH: You've gone back to that then?

GEAR: Yes. And not to write a specific program, but to do some research in some numerical things, which involves producing programs that do it.

HAIGH: Now I should actually say some of the papers from the ILLIAC have made their way into the Charles Babbage Institute at the University of Minnesota. There's a Nash collection, which is CBI collection #14. I've got a couple of things that I scanned from it, including the operations manual for the ILLIAC and some copies of the program library. I've got one here as of January 6, 1956. So it seems from those papers that they had actually received a grant in order to develop a numerical analysis library very early on in the history of the machine, I think starting work even before it was finished.

GEAR: I'm not certain how much of this stuff-- I must have been paid on a grant, I assume. By the time I got to Illinois, there was AEC money there, which was mainly funding the design of the ILLIAC II. Maybe it was also paying for some of the programming, I don't know. But a lot of other libraries at Illinois had been written by various users from around the country who came to use it.

HAIGH: Actually, this is a printout of programs as of 1956, the ILLIAC active program library index. A number of matrix routines, for example…

GEAR: Yes, I remember some of those.

HAIGH: Simple things. Here's a list of courses that were offered.

GEAR: Okay. Those are the two courses that I took: 384 and '85.

HAIGH: So *Digital Computer Programming*, the description: "An introductory course in the use of automatic digital computers. This course uses the university computer ILLIAC as the basis for discussion. Machine methods for performing calculations are discussed and the problems of organizing a large-scale computation are considered in detail. Included are topics of integration, interpolation, solution of algebraic equations, and solutions of differential equations. As part of the coursework, students are expected to prepare programs for solution by the ILLIAC."

GEAR: I do not remember any of the numerical stuff they mentioned; I remember the programming. But in those days, computers were mainly used for numerical programming anyway, so they would need some examples, so those are probably what they came from.

HAIGH: Yes. So as you began to work there as a research assistant, what kinds of responsibilities did you have?

GEAR: The first semester, which was the spring semester of ’57, I was asked to program up a code for [Wallace] Givens’ eigenvalue method for symmetric matrices. So my first task was to write that code. It was an interesting challenge, because the maximum symmetric matrix you could store on that computer turned out to be 127 by 127. You could only store it if didn’t store the lower triangular part. The drum was a rotating magnetic device. Basically, you wrote code to read from the disk a word at a time. And you also had to be very sensitive as to where the disk was. So for example, depending on the number of instructions you wrote between each disk read, which determined how far the disk rotated, so you needed to use [the appropriate drum] address. The addresses had been numbered so that there was time in a simple read loop to do the necessary calculations [between successive addresses], which included modifying the instructions themselves (because you needed to use the arithmetic unit to change the address in the instruction -- there was no indexing). So you could do a tight loop on successive addresses—they were actually five locations apart on the disk. .

There was another trick you could do. There was a series of tracks. I forget how many tracks there were, but I think there were 64 words on each track. You could not only go to the fifth word down on this track and run this calculation, you could go down to the fifth word on *any* of the tracks in the same time.

HAIGH: I believe with the drum system you would have different heads for each track.

GEAR: There were fixed heads; there was a head for each track. So this meant that say you had a 64 by 64 matrix. If you stored it so a column was, say, on a track, five words apart (which was actually the next numeric address, anyway), but you stored it so the next column was displaced five words. You could then read by rows by jumping from track to track. So that was fine for a regular matrix. Now if you have a symmetric matrix, of course, you’re throwing away half the storage. If you looked at the amount of space and think about a symmetric matrix, chop it up into three pieces, cut it in half at the halfway point, and then you have a square which is in the upper triangular part and then two triangular matrices. If you take the piece at the bottom and turn it over and slide it up, you’ve now got a rectangular matrix that now fits nicely and it is 64 words long, so you can store it. And if you position that other triangle piece very carefully, you find that you can read either any row or any column. Actually, you had to throw a little bit away; so the maximum size was 127. So I guess the main feature of my program was achieving that to get maximum speed read. And there was certainly no more space.

It took three hours to compute the eigenvalues of a 127-by-127 matrix, which meant that there was no question about debugging it in the five-minute maximum time allowed. In those days they turned the computer off at 8 a.m. on Saturday—it ran three shifts during the week and then went back on at 8 a.m. on Monday. But if you checked out as an authorized use of the computer—I checked out -- so I could come in at 8 a.m. on Saturday and take it over from the operator and use it till I was done and then turn it off. So I came in and would run my code for three hours. The first time I ran it, it got halfway through. There was a speaker on the sign bit of the accumulator that made a noise which told me that I had a bug. I went away until the next week, and found it, and came back and eventually got the thing running. So that was my first task.

And sometime at that point in that spring semester, Jim Wilkinson came through to give a talk. That was the first time I met him. I was asked to tell him about this program, and he told me about finding the eigenvalues using the inverse iteration, which he had just recently come up with. So I then added eigenvectors to it, although that reduced the maximum size I could store to 64 using his inverse iteration technique. That took me thorough the end of that first year at Illinois, and then I became basically a research assistant with Don Gillies, who was working on the logical design and architecture for ILLIAC II.

HAIGH: So after that point, you weren't programming ILLIAC I anymore with the research assistantship?

GEAR: Yes, that was no longer my task. I still used it in classes and I wrote one or two subroutines for it. I don't know why, maybe out of pleasure.

HAIGH: Did you have any experience with the library of subroutines that was already there? Did you use any of the routines at all?

GEAR: Well, since there was no operating system, if you wanted to print some decimal numbers out, you took the output paper tape and then you ran it through a separate teletype printer. You went to the library, which was a series of pieces of paper tape, and copied the print subroutine. Any standard subroutine whether it was print or square root, had to be added in to your piece of tape that you put in.

HAIGH: And was there a tape-duplicating machine you could use?

GEAR: Yes; the librarian kept, of course, master copies of all these things. And then they kept copies that people had copied from. Then there were tape-duplicating machines--

HAIGH: The tape would be rolled up into a little roll.

GEAR: Right. They had built a modest -- the standard units we used in those days was five-hole teletype tape, the early teletype tape. They had a high-speed tape copier, and it went all of 120 characters a second. What we used to think of as high-speed is somewhat primitive now. When you corrected the bug, you decided what change you wanted to make—you learn to read the paper tape pretty quickly, locate a position in the tape going to the copy machine, copy the tape that far; and then put the splice in and copy that onto the new one and go back to the other one. It was pretty tedious, but it kept down the demand for the computer because you couldn't have too many people debugging—they couldn't prepare the code fast enough.

HAIGH: Was there ever an assembler for the ILLIAC I?

GEAR: Yes. After I had been there a while, they developed the symbolic address assembler. When I first went we were using something called decimal order input. The only thing it allowed you to do was to use decimal addresses instead of binaries or hex-- Actually, it was a base-sixteen system on five-hole paper tape. Base sixteen, you really couldn't go any more primitive. They didn't call it hexadecimal, which was IBM's thing; we called it sexadecimal (I guess IBM felt that wasn't appropriate!). So an instruction set with two sexadecimal digits and basically the binary forms. I still know the binary instruction set of that machine, and that's really because it was, in a way, a microprogramming machine in a sense that the DEC used the term for the PDP-8. It was bit-significant. For example, if the last bit of the second of the two digits was 1, the accumulator was cleared at the start of the operation. So for example "add" was L4 (we used K-S-N-J-F-L, not A-B-C-D-E-F), which was 1-1-0-0-0-0-0-1-0-0. If you changed that to L5, which made the last bit a 1, it cued the accumulator first; that was the "load" instruction. Because of

that is very easy to memorize what all these instructions did. The first four bits pretty much determined whether it was add, branch, test, multiply, divide, and so on. The last four bits determined what it did prior to the--

HAIGH: They also had an oscilloscope on it. What was that used for?

GEAR: What you could do is display a dot on the oscilloscope from code. You basically put out the *x* and *y* coordinates and say "display." And it had a 35mm camera that could be closed over the top of it, and a thing that would open the shutter and close the shutter and advance the film.

HAIGH: So if you moved the dot around with camera open you could make a line?

GEAR: Yes, you could make a series of dots or anything else. So it was just displaying the output. I think they had two displays that were in parallel and one had a long time-delay phosphor, so there was some persistence so you could see what was there. But for the camera it was awfully short delay, so the stuff would go away and you would change it to the next frame and take another picture. So it was generating primitive graphical output, but really graphs. I think the TX-2 at Lincoln Labs had the first graphical display, where it could do vectors and stuff. I'm not positive of that.

HAIGH: That's I guess in conjunction with what would become the SAGE Project, the air defense system that grew out of Whirlwind, because they essentially have computer-generated radar-type displays.

GEAR: Yes, that was at Lincoln Labs. Right; yes.

HAIGH: So that's the ILLIAC I, I think, dealt with. You said that after that initial project you switched over to work with Don Gillies from the ILLIAC II. So at that point, was the computer still at the planning stage?

GEAR: Oh yes, it was in the very, very early design stages. It was going to be a transistorized computer, and at that stage they were still designing the transistor circuits or various parts of it and thinking about packaging. Mainly we were working on architecture and the inner layout of the processing unit, you know, what registers it had, how many. These were still discrete component things, so everything was very, very expensive. In fact I have down in the garage a chassis from the ILLIAC II, if you want to see it.

There were three computers at the time in the design and construction with goals to try and increase the speed of computation by a hundredfold. One was the ILLIAC II, which was funded by the Atomic Energy Commission. AEC became ERDA, Energy Research and Development Authority, and then became DOE. Another was the IBM Stretch, which I believe was really designed for cryptographic work—it was heavy on processing streams of characters. That stuff was never made public, but it was pretty much clear if you looked at it. And the UNIVAC LARC. I don't think any of them were truly successful. In fact, when I worked for IBM later on, Stretch was either being referred to as "St-Retch" or "Twang." I don't know what happened to LARC, whether it made any news or not I don't remember.

The ILLIAC II was probably about two years late coming online. I think it was originally aimed for about 1960. Unfortunately at the time it was being basically hand made at Illinois, and they had to make this hard decision about packaging. Printed circuit cards, which were just coming in then, were having a lot of problems with the conductors breaking. You can imagine there's a conductor wire in this card and something moves and it snaps and you can't fix it. Whereas a company could make a decision about packaging much later in the process when they're

absolutely certain about how things are going, they [Illinois] had to settle for a packaging mechanism early on because they didn't have access to equipment, which made it worse than it would have been otherwise. So it was a massive machine. The room it was in had at least a ten-foot ceiling, maybe twelve, and it [the ILLIAC II] went most of the way to the top. We had a stepladder to work on it—a big stepladder.

HAIGH: So each transistor pin was wired to other components just with a regular wire, was it?

GEAR: They were in sockets, I think. I can go get that thing and take a look. I think they were in sockets.

There was another problem back then that was interesting. It was all made with germanium transistors. At that time, silicon transistors were just starting to catch on. They weren't fast enough. The problem with germanium transistors is, if they went into saturation, which is when you're driving to the extreme high end of the current, they took a long time to come out. I assume it took a long time to get the charge back out. So you really had to prevent voltage swings. This circuitry used all these transistors with diode clamps to stop them from going too far. Also, every time you needed to amplify a signal, make an inverter, it's also very much slower for some technical reasons, whereas you could do switching in diodes much faster. So everything was done to get speed on this, which sounds pitifully slow now—the multiply time was two microseconds, which is an eon these days. No, the multiply was about three-point-five, and the add time was about two. One designed circuits so that you went through a series of ANDs and ORs without ever going through a level restoration or an inverter, which did amplification, which did level restoration. So until the signal got so weak that it was getting in danger of getting in the noise--

[Tape 2, Side A]

HAIGH: You had just been describing the difficulties that had been involved with the ILLIAC II architecture because of its use of germanium transistors.

GEAR: Yes, and they couldn't go into saturation or that would slow it down. So lots of things were done to try and get speed. In the end it didn't achieve all the speed that was wanted. It was, of course, a machine that was designed really around numerical calculation, so it wasn't rooted in any other application. It had an inadequate memory. The guy who was my advisor stated that 8192 words would be enough for any foreseeable problem. Of course, many years before, von Neumann had said that seven computers would satisfy all the world's needs! So we should never believe numbers of those sorts anytime.

HAIGH: And that was [Abraham] Taub that said that?

GEAR: Yes, Taub. He was head of the department from '57 until '63, I think it was.

HAIGH: So your personal involvement with this… you would have started work on this in 1957?

GEAR: Yes.

HAIGH: So were you involved in making these basic kinds of architectural decisions?

GEAR: Yes with my advisor, Don Gillies. The main question at that time was basically what registers we had. In those days they were very expensive. These days you have thousands of registers in a cache somewhere. Every register took 52-bit words, and there were probably about four or five transistors per bit the way they were designed. So there were about 250 transistors,

and the transistors cost $40 each. So you had to be a little sparing about how you did this stuff. Thinking about the various instructions one wanted to be able to do, the important instructions like multiply, floating-point add, and so on, and how they were done, and what organizations or registers led to the most rapid execution. So it was a very low-level design issues at that time.

HAIGH: And who else was involved in producing design?

GEAR: Well, there was a guy named Jim Robertson, who was on the faculty at the time. He was the EE person who taught the logical design course. And he was more responsible for the design of the circuits. Another guy, [Wolfgang J.] Poppelbaum, who was from Switzerland and was on the faculty the whole time I was there. He was more of a circuits guy. Robertson was more EE/logical design/organization. They were more concerned about the implementation of issues of the system. Gillies and I were concerned about the layout of the arithmetic, really not anybody else. One guy who was there briefly at that time was Vel Kahan, who was from Toronto. He was either there for a summer or a whole year as a post-doc. It was kind of interesting because that was the first time I met him, and I have a strong respect for him, which has gotten stronger as the years go by. I remember at that time he was a very interesting and intelligent guy, but I found him pretty exasperating. Even then he was thinking about the ideas that eventually led to the IEEE floating-point standard. He was pushing some of those on us, and in those days, they were just unbelievably expensive. Whatever possibility was in his mind then, it was just out of the question. He would spend all his time on this, and I thought, "What a waste of time! We can't possibly consider any of this."

Another problem was-- there was a guy named Dave Muller, a software guy who taught programming, who was moving into what we call now *theoretical computer science*. And he had something called *speed-independent design*. The ILLIAC II (and ILLIAC I, actually) were basically asynchronous computers, which was oddball in those days; most computers were synchronous. That is, they had a clock. The idea of an asynchronous computer is that you try and have everything run as fast as it can, and when it's done, the next thing can take off. And Dave Muller's idea of speed-independent design was a very interesting one. Basically it was the following: if you have circuits that eventually change, and you just don't know how long, can you design a total system that will eventually get to the right answer, independent of how fast each element runs? And the answer is yes, if you can have certain types of properties. There was something called a "C element" (right now I'd have to go look up exactly what it did). But in some sense, each circuit had a completion signal which was guaranteed not to change until the circuit was finished. That was the easy way of thinking about it. Originally, the ILLIAC II was going to be speed-independent. It became very clear later on that it was going to be way too expensive; it almost doubled the cost of every register. It became somewhat speed-independent in that the decision that was made that the individual flip-flops in the registers wouldn't be speed independent, but it would speed-independent up to the control circuit that drove the "gate" signal. Then one would take the signal from the far end of the register when the gate drive signal had gone down, and bring it back and use it as a "completion" signal. And so the whole control part was speed-independent.

Even later on there was a desire make it faster, so we went in and experimented with adding capacitors to jump the signal to say it was done back to an earlier point to try to make it run a little bit faster. Basically you could do that sort of tuning stuff and keep shortening the delay time until it failed, to try to tune it to maximum speed. It was a very interesting computer, but it was very primitive by today's standards. You wouldn't start to design something like that now.

HAIGH: How much of the design did you come up with personally? You were working with Gillies on this.

GEAR: I no longer remember. He and I used to interact regularly. Although he was a faculty member and I was a grad student, there weren't many grad students in the department and there weren't many faculty, maybe half a dozen, I would be home in the evening thinking about it, and I'd call him up at home: "Hey, Don, how about connecting this to this? We could do the following." And we'd swap stuff all night, so it was totally a joint thing.

HAIGH: How did you approach this in terms of looking for inspiration? Did you spend lots of time looking at the designs of other machines to see how things were handled?

GEAR: Well, one knew about things like index registers from the Manchester machine, which called them B-lines because I think they were in a different B position on the CRT display, or something like that, but I don't recall exactly. So one knew about various things that were going into machines. By then those were very common. This was, as I said, AEC funded, and there was an annual meeting of all the AEC contractors who were building machines. Argonne had built a machine. So there was a sharing of ideas. But there weren't that many people total in the field then, so you couldn't interact with many people. A lot of the work was being done relatively secretly at places like IBM, so there wasn't so much interaction with those sorts of people. There wasn't a hell of a lot of literature available, certainly on the latest stuff.

So the ILLIAC II had, in some sense, some of the early parallel processors. Not parallel as we think of today. In the ILLIAC I, there was just one thing going on at one time. By the time of ILLIAC II, it had something called "advanced control" and "delayed control." Basically, the instructions were processed in one unit early on, and it did all the fixed-point arithmetic, which was mostly index register stuff. Then everything was gotten together and sent to the floating-point unit to do that process. And so the thing would be processing parts of more than one instruction at the same time, which is absolutely normal now.

HAIGH: Like pipelining.

GEAR: Yes. But all that stuff was new, and one was feeling one's way and trying things.

HAIGH: Was there anything that you think was novel or interesting about the floating point or the instruction set on the ILLIAC II as compared with available machines such as the IBM 704 or 709?

GEAR: Compared to the 704/709, it had a bunch of registers called the "flow gating unit." I think this was a development of Ted Poppelbaum. As I said, I think the typical register in the accumulator took four or five or six transistors—a lot of stuff, and really expensive. The ILLIAC had maybe eight or nine flow gating registers. They were kind of interesting in that they were running at, I think, two transistors per bit. The way you gated information into them was to lower the voltage on the whole register and the information would flow down a bus from the way in, because it suddenly became lower than some critical value. The way you got information out, as I recall, was you raised it and it exceeded some diode level and the information would flow out. So that was the switching mechanism, and it used diodes, which were much, much cheaper.

It enabled us to have quite a lot of registers, I think nine registers—F1 through F7, or F1 through F9, I can no longer remember. F1 was the input register from the memory, which of course, already you were already trying to pre-fetch from memory. Oh, F0 was the output register *to* memory. It was dumped there and then, later on, written in memory. I think F2 and F3 were

general-purpose registers; you just use as scratch-working registers. Then I think it was F4 through F7 could either be used as general registers, or they each contained four thirteen-bit index registers. As I recall, you could use them in multiple ways. And furthermore, because this stuff was all going on in pipeline processing, you had to be aware of that in programming. Oh, F8 and F9 were two words of instructions because they were prefetched instructions. For example, if you wrote back into the word that was already in the instruction registers for the next word, it was too late, it had gone. So you had to keep in mind in your program what was in those registers. That was one of the challenges of the compiler and the assembler, to remove that from being a problem for the programmer.

HAIGH: And the floating point design. So by that time, was that something that all machine designers were approaching in pretty much in the same way, or were there still difficult challenges or decisions that had to be made in terms of how to implement floating point?

GEAR: We hadn't yet gotten to incorporate the ideas of Vel Kahan, which have been incredibly important. The big problem in those days, and why addition was slow, was the carry propagation. When you add two numbers, if I add one to the number 011111, it becomes 100000. And if you think about how it happens, I add one to the bottom number and it causes a carry, which goes to the next number, and this is a serial process. These days, where the number of transistors is irrelevant and it's the space you use on the chip that counts, you simply have things called "carry look-ahead logic," where you just try and determine very quickly if a carry is coming from this group of bits, so it gets up there very rapidly. In those days that was out of the question; you needed to worry about it.

So a very clever idea, which I recall was Robertson's idea, called "carry-save logic." In the CPU (in the registers and in the accumulator) one had a redundant representation. It was really a base-four machine, two bits, and then an additional bit that was the carry out of those bits under bad circumstances. So what you did was when you added two numbers, if you knew right away a carry was going on, so you could send it out right away and it got incorporated. But if you didn't find that out till later in the process, you saved it, and then absorbed it in the next process. And you could show by very standard theory that this was enough to always do the arithmetic in one carry-propagation time, so you didn't have to wait for it and it could run much faster. And you never got into states you couldn't represent.

So this made things like multiplication much, much faster if you didn't have to wait for the carry to propagate. It was a big challenge for division, as it turned out, because you had this weird representation, which made division a real challenge. And division was a little expensive, because basically you had to look at about eight bits of the divisor and the dividend and make a guess as to what the most likely answer [next base four digit of the dividend] was in order to get to guarantee that you had it within sufficient accuracy so that you could still use the carry-save bit to correct it. So it turned out to be expensive, but it was a clever idea. In those days we were working very much down at the bit level.

HAIGH: I don't mean to jump totally ahead yet to when the machine was operational, so I'll ask you about that later on. But as you were later fortunate to have the experience of programming the machine and writing system software for it, I was wondering, were there any elements of your design that you came to regret later?

GEAR: No, it wasn't my design. I did not like the index register organization, but it was -- as all these things we were just talking -- about cost. Anything that I might have done differently

probably would have only really affected the speed. Positive or negative, one wouldn't know. There were not design decisions that were being made that had a major impact that I can think of.

HAIGH: Okay. Did you continue to work as a research assistant through the rest of your Ph.D.?

GEAR: Yes, until I left in 1960.

HAIGH: All right. I guess before we talk specifically about your thesis, if you have anything else to say about relationships with other professors or with fellow graduate students…

GEAR: Well, I was in the digital computer lab, as it was then, and had very little to do with the math department. As a research assistantship in the digital computer lab, I was very close to the grad students—Gene Golub being one of them, a very close friend. He was a year ahead of me, a student of the same advisor. And there were a few others—I remember a few of the names, but I don't know what happened to the vast majority of them anymore. One who was there—and it may have been in the early '60s after I came back, I don't remember—was Bill Wulf, who is now the chairman of the National Academy of Engineering. He did his master's at Illinois before going to Carnegie-Mellon. So I knew him from those days, some time then, but I don't know exactly when.

It was a very small group. After Taub took over as head when Nash left in 1957, he would have an annual Christmas party (we weren't calling them "holiday parties" in those days). He would have it in his house (which was not a very big house in Urbana) the whole department—all the faculty, all the grad students, and all the staff. And it wasn't terribly crowded. So that's an indication of the size: I say there were maybe about six faculty; possibly 20 grad students, maybe less; and a few staff, machine operators for ILLIAC I and technicians for the construction of ILLIAC II. So it was a very collegial group. We were almost pretty much on a first-name basis with the entire faculty very early. I used to call Don because I was up in the evening and discussing stuff, or go over to his house. I used to play bridge with Gillies and Robertson.

HAIGH: So how would you describe Gillies as a person and a researcher?

GEAR: He was a very smart guy. He did his undergrad at Toronto, went to Cambridge for a year as a post-doc in possibly 1955, I'm not positive. I didn't know him then. He was totally disorganized, and I think therefore, not very productive because he never sort got around to finishing anything. But very smart.

An early story I heard about him, when he was at Cambridge and he was looking at a job; he got a job offer from Illinois, which he thought was pretty interesting and he decided to take it. So he went there, came in, and walked in the lab, and Robertson was there and seemed a little surprised to see him. No problem, though, so they found a desk for him and so on. And then some many months later, I guess when it got cold he put his coat on and put his hand in his pocket, and he found the letter he had written to accept the job in the coat pocket! In any case, it's also an indication of how few people were available then that although he didn't accept it, it didn't mean he had lost the job. There were just not many people to recruit in those days.

I enjoyed him a lot. He died (unfortunately rather young) in his sleep. I was on sabbatical in summer of 1975, I think it was. I was 40, so he was probably about 45 when he died.

HAIGH: So you also mentioned that you were in contact with Gene Golub, who obviously has become very prominent in the field. What were your first impressions of him in his younger days?

GEAR: I met him when I first took the research assistantship and started programming; in fact, I shared an office with Gene (along with many other grad students at various points in my career). We quickly became friends; we were quite close friends. I've always liked Gene and have great respect for him, and I think the feeling is pretty mutual. In fact, after I came back to Illinois in '62 I was working on ILLIAC II, and Gene had by then gone to Stanford. He called me up and wanted to know if I would consider a position at Stanford. I said, "No, I'm having too much fun working on the ILLIAC II." Maybe it was a bad decision; I don't know. I made it, anyway. In the early days he helped me learn to program the ILLIAC I, and some of the tricks and showed me around a lot. Like I said, we became close friends.

Abe Taub was an interesting character. I liked him a lot. A lot of people, including me, found him a difficult person. He loved to argue. My sense was that, the more uncertain he was of himself, the louder and stronger he argued because he didn't like to lose an argument. This never particularly bothered me, because I like to argue too, I guess. I had shouting matches with him and it didn't seem to upset him at all. I remember one time I was in his office and we were arguing about some stuff on my thesis. He was determined to win a point, and he made some mathematical statement. Then I saw my opportunity, because it was clear from that statement, if it was true, I could prove something that made no sense. So I banged on his desk and said, "Well, in that case, every goddamn function is continuous!" I proved it to him, so he backed off.

Well, Gene didn't like that sort of interaction at all, and had a terrible time. He felt browbeaten or whatever. He would come over to our apartment and we'd have to calm him down. And I found out much, much later from Abe's wife, Cece, who was a very nice person, that Abe didn't like it, either. He would come home from his meetings with Gene and be unable to eat lunch, she told me. It's so odd, because I guess I thought he thrived on this.

HAIGH: Taub was miserable because he wasn't being argued back to?

GEAR: Maybe, I don't know. Well, maybe he didn't like the fact that he had obviously upset Gene. He wasn't able to handle it any other way. I don't know. I'm just guessing.

I'm jumping forward to when we were working later on; doing some new stuff to the ILLIAC II, and Abe was still department head. There was a young professor also from Toronto, Ken Smith in the department. He told me a story about being in Abe's office once, and Abe was getting really mad over something. Ken was arguing with him, and Abe banged his fist on the desk and said something. And Ken said, "Did you hurt your hand, Abe?"

But, you know, you're going to learn to interact with all sorts of different people and not get too upset with it yourself.

HAIGH: So how did you come to do your thesis work with Taub?

GEAR: Well, I was a student in mathematics formally, so I had to have a mathematics-type thesis. The faculty in the digital computer lab then included a couple of electrical engineers, maybe three; a physicist; and three mathematicians—Gillies, Muller, and Taub. Gillies was so totally into architecture and stuff. The person who was, in '56 at least, was still running it, I think that must have been Ralph Meager, had this very rigid view, which I don't find too unusual, that if someone was paying you to do some work, then they were paying you to buy that work from you so you couldn't make it part of your thesis. It wasn't that you couldn't be in the same topic, but it had to be different material.

HAIGH: Presumably the same logic would prevent you from getting academic credit for papers you published from a grant.

GEAR: Yes [laughs]. Well, it seems like a crazy idea these days, and I think when one looks at it from another perspective, and that is, the real purpose, I think, of U.S. support of research in universities is not so much for the research, but for the education that's done and the talented workforce they've gotten from it. I believe that enlightened view is responsible for the success of the U.S. through the last part of the 20th century. And it's not unreasonable to do it, but one could see where you might have that kind of rigidity at the time.

So I needed a different topic. And besides, I don't think the architecture would have sold in the math department as a thesis anyway; it had to be something in math. I had taken a couple of classes from Taub on numerical analysis and one to do with partial differential equation computing. The other person was Muller, but he was getting very much more interested in things we now call theoretical computer science, which was not of so much interest to me. Taub was the obvious choice, and he essentially assigned me my thesis topic, or suggested a thesis topic. My view, as it has been for the longest time, was that I wasn't choosing a thesis topic because this was going to be my life career; this was an exercise in learning how to do research and be a researcher and it probably didn't matter what I did, as long as it didn't bore the hell out of me. So he gave me a topic which was really hydrodynamics. I think it was a problem that he had probably worked on in the war, and was still an open problem because at that time, computers were not powerful enough to do what we can do now. It had to do with apparent nonexistence of solutions in some regimes of weak shock reflection (although in experiments in shock tubes you could see the results) and how to explain them mathematically. So I did some stuff both in trying to provide theoretical explanation by means of, actually, a singular transformation, which showed the existence of some more solutions. Then I did an unbelievably crude numerical approach to it, based on some work that had been done by somebody else.

HAIGH: And did that involve some programming on the ILLIAC?

GEAR: I ran some code on the ILLIAC I to try and solve what were essentially some algebraic equations. I don't think that part was very successful; it was not a very good model. I don't think that it was a very important piece of work. In a sense, I had nothing to do with it ever after, because I left Illinois to work at IBM in architecture. In fact, after I came back to Illinois, Abe asked me to supervise a student in a similar area, so I had one graduate student who worked in that, and that's the only stuff I did in that again.

HAIGH: If you had been in an electrical engineering graduate program, do you think you would have been able to get credit for the architectural work, or would they not have recognized it as appropriate, either?

GEAR: Oh, yes. I don't know what they were like then; probably they would have done it in those days. I believe that engineers were less rigid in their views. I'm not certain that the math department would have said no in those days. It wasn't clear to me what there was to do from it… I mean, this was design. There wasn't a lot of disciplined analytical technique to call on to do it yet. I couldn't have started designing other architectures; there was no way to implement it in those days. It took a million dollars, so to build something was out of the question. So it isn't clear to me that I could have structured a thesis in that topic in those days.

HAIGH: As you approached your Ph.D. graduation, how did you think about what to do next?

GEAR: Well, I had to go back to England. I had been on a Fulbright fellowship for travel, which involved a J-1 visa, which has a two-year home return rule, although, actually, the department was offering to try to get it waived for me, because they had this AEC contract. By then, though, I was married; I had a kid, and my family in England had not seen the new grandchild or my wife. I felt that I should go back, so I don't think I gave much thought to any option other than returning. I'm pretty certain that I had no intention of staying, though. Since I had never planned my life carefully, I would hesitate to say that I had absolutely no intention of staying in England. But I did not look for jobs in the U.S.

HAIGH: And how did you go about your job search for England?

GEAR: Well, not very carefully. I'm not absolutely certain how I did it. I was interested in going to Harwell—there was a lot of interesting computational work going on there. Of course, Wilkinson was there among other people. There was Aldermaston, too—there was a lot of work going on, and still is, I guess, in plasmas. They've been trying to solve the fusion problem since the early days of computers, and they're not certainly much closer yet. I actually was interviewed in New York; I think at Courant by somebody was visiting from one of those places and I considered going there. I think I got an offer from there, I don't absolutely recall that. I really wanted to go to a university, and I got an offer from one university, and I can't remember, one of the newer Midlands universities. Sandy Douglas, who suggested Illinois for the fellowship, was there and made me an offer.

The big problem I had was that I was married and had a year-and-a-half old daughter by that time. The university and, I think, Harwell or Aldermaston, wherever I got the offer from, made me an offer of about £500 a year. That was considerably less than my part-time research assistantship at Illinois—even in those days when, I think, it was $2.80 or so to the pound. And they expected me to pay my own way over for my family. I had a Fulbright to pay my way back, and I had a few more belongings by then, more stuff for a family.

IBM offered actually about what I was getting for the research assistantship; maybe a little bit more, I think it was £1,250 a year and paid travel. And it was an interesting research place. I think I spent one summer in '58, at IBM research; this was pre-Watson Research Labs days when they had some various labs in Ossining then, so I had contacts there.

It was in Hursley at that time; it was an old estate in downtown in Hursley village. It was an old manor house where my office was in a big room there; I shared it with my boss. There was also an old building that had been sort of an aircraft hangar-type place, and I think during the war it had been used for assembly of some aircraft parts then later on they built a big research lab there on the grounds. When I went there it wasn't very big. In fact, when I first went there, only the house was used. It was kind of a pleasant place. We would wander down to the village pub for lunch and walk around the grounds on our lunch break. We played cricket on the lawn behind the house. It was interesting, and I'm glad I went there, because in fact it was more active from a research point of view than it would have been in an English university in those days. They were in the process of designing IBM's first micro-programmed machine. I became involved in that. That was about the only project; it was a very small lab when I went there. There were some interesting people. I worked on this machine, which was a lot of fun. I was, first of all, considering some software issues; again, it was a pretty small machine.

Then I got involved in a simulation. For a while I was going to Paris. We didn't have any computer equipment at Hursley, except for some card machines, card calculators. I was going to

Paris pretty often to use the 704 in the Paris office. That was a pretty nice trip. Then it was clear that the machine we were designing was becoming too expensive and, by then I had gotten pretty involved in it and, for about three or four months I led a team that they put together by bringing in people from various places—Germany, France, and various other places—to try to reduce the cost and reorganize the machine. Not change the design from an architecture point of view, but repackage it to cut the cost. That was pretty interesting. The guy from France, I remember, he had a place in the country and made wine; he'd come in with two or three bottles of wine each time he came back. Usually there were a few ants and stuff floating in it, but it was pretty nice and it was free. In those days, wine was very expensive in England.

HAIGH: A follow-up question. You had mentioned your wife. Was she a fellow graduate student?

GEAR: This is not the same as my current wife. She was an undergraduate at Illinois who I met when I went to Illinois. We got married in '58, had a child in early '59. She took an undergraduate degree in painting, though I don't think she had done anything with it. She lasted until 1970; I don't follow what she does now.

HAIGH: So did having an American wife make you feel that it was more likely that you would be returning to America?

GEAR: Probably, but it was pretty clear to me that I liked America. I found England unbelievably frustrating when I went back. The first thing was housing; there was very little on the rental market and there wasn't a lot on the sale market. I found a new house that was available—a small house, but okay for our purposes—about five miles from Hursley, so I decided I'd buy it. At that time my wife and child were living in London with my parents and I was driving down there and staying at a residential hotel during the week, so it wasn't very pleasant. So I wanted to buy this house, and the lawyer said, "Well, it will take about two months to close." And the house was finished! "It always takes this long to do the paperwork." I knew if I had asked, "How much extra to speed it up?" he would have just thrown me out of the office. All sorts of things like that, just frustrating beyond belief.

The company, which was very English, was pretty rigid. I was treated pretty well. That was IBM—very English. The personnel director was an old RAF type, the sort of person that you would not particularly like, I think—kind of heavy on the white shirt. I was left pretty much alone. I was one of the two Ph.D.s in the company, so I think I was viewed as an oddball already. It was hard to get anything done even in the research arena; everything took its proper length of time. After I'd been there only a few months and I expressed frustration, at one point my father said to me, "If you like America so much, why don't you go back there?" I had obviously annoyed him.

So it became pretty clear that I wanted to go back. Though I enjoyed my time at IBM thoroughly and was exposed to a lot of stuff that I wouldn't have been exposed to otherwise, like microprogramming. I helped design that machine. I spent some very pleasant evenings with an oscilloscope, checking out the first version on the floor.

HAIGH: And was Hursley a freestanding research installation, or was it associated with any manufacturing plants?

GEAR: No, this was just a freestanding design place. IBM, in those days, would have several machines in design and they would give them time, and then at the end of the process they would choose the best one. It was a good way if you had plenty of money and not much competition.

We lost out to what became the 7044 (because it was compatible with the 704 and 7040). The machine we had was really radical for that time—a fascinating machine. It had independently programmed channels. The test-and-set instruction, which was in the IBM 360 I think for the first time, was created in Hursley. Most reads are destructive. In a single read-write cycle, to get something out of memory you have to strobe it in some way (at least in that we were talking about core memories, but also with most other memories). You have to destroy the information to read it out, and then you write it back in. So that's a full cycle—either to read or write is a full cycle. On the test-and-set instruction, you read a word from memory, you flip the top bit to 1 and write it back, but you deliver the original version to the reader. What's important is, you use that 1 to indicate a block in the entry to a queue, for example. It's a semaphore, basically. So that if you have several different channels in this case, or CPUs, and wanted to do something, you say, "If this bit is a 1, someone is using it." And so, when you go in you do a test and set you look at it: if it's already a 1, it means that something is there; if not, you go on through, you set it to 1. When you go out, you write it as a 0 and that lets the next person in. I think that was the first place that that instruction had ever been used.

HAIGH: Was this machine intended for data processing rather than scientific work?

GEAR: Probably data processing, given where IBM was. But it was a two-address machine, which is very oddball for IBM. (Another one they produced was the 650, which was two-address for a totally different reason, to do with use of drums as primary storage). It had independently programmed channels; a fascinating, very complex interrupt system; memory queues for the jobs for all of these things that were done automatically. The reason that you could do all of this stuff was, because it was micro-programmed, you didn't have to go to control by discrete component hardware. That was the power of microprogramming in those days.

HAIGH: Did the product have a name?

GEAR: SCAMP. It lost out because it wasn't too compatible and it was too expensive. By then, IBM decided it needed to unify its series for the whole thing, which became the 360 series, to satisfy everybody from the high-end user to the entry-level user, obviously to avoid the software hassle.

HAIGH: After which there would have been a lot less architectural designing to be done?

GEAR: Well, no, the machines were architecturally different. The instruction set was the same, and the software was the same. But in fact, the architectures were very, very different to achieve different speed classes. So after SCAMP was canceled, IBM started this project to design what became the 360 series (that wasn't called that, then; I don't remember what the name of it was). Gene Amdahl was in research—by then I guess Watson Research Labs existed—and had designed an interesting machine: the 8000, I forget the number, which was a stack machine. It was an internal stack machine and IBM designed it. Amdahl was responsible for it in research. Brooks was in Poughkeepsie, which was more of a product development division. He ran a fairly large group. So a committee was set up, and Amdahl was brought in to chair it.

[Tape 2, Side B]

GEAR: So we'll finish up the IBM stuff. There were representatives from various IBM places that designed machines. One other person and I were the main representatives from Hursley, and there were representatives from Kingston, Poughkeepsie, and I don't remember where else. It

wasn't a big committee. Oh, and I think there were some representatives from the sales and marketing end; I don't remember who they were. There were people who were talking about the kinds of problems that they needed to solve. So we starting talking about the design of the machine, and it was pretty hopeless. Everybody had their own ideas.

The 1401 was out by then, I believe. The 1401 had an unbelievably small amount of hardware in the CPU. It's amazing how little there was, and how cheaply they could make it.

HAIGH: And then the 1620 was…?

GEAR: The 1620 was modeled on the 1401; it was more for scientific calculation, but the same idea. Well, the 1401, in dollar volume, accounted for about half the sales of computers for IBM. The rough rule of thumb, if you looked at the 360 series sales projections, was each machine as you went up the line accounted for about half of the remaining sales volume. So the bottom end, initially the 30, was going to take in half of the total dollar volume—an enormous number of units. Basically we were told [by the representatives of the 1401] that the machine has to be competitive with the 1401, price and performance wise, or we won't be able to sell it, and we won't therefore go along with it and so there was the problem of trying to match the high-end machines and get a high-performance machine. And it was going nowhere.

So at one point (I think somebody must have just given up), we had a design competition. Each group suggested designs. I was responsible for the Hursley design, and I put in what I thought was a really neat machine; it didn't get anywhere. It had sort of a multiple personality, and at one end it was able to act very much like a 1401. In the 1401, basically the CPU consisted of a very few registers. It was a decimal machine. There was one digit plus some stuff, accumulators; everything else came out of memory. There were two address registers, the *A* and *B* addresses; it was a two-address machine: *A* + *B* goes to *B*-type thing. And since it was a digit serial machine, if you set the *A* and *B* registers, say, to 200 and 300 and said "add," it would read out the numbers from 200 and 300, add them together, and bump those locations, 201 or 301, put the thing back, keep the carry, and then do add again. So it would add up the line. It was a word mark machine; there was flag in memory that said, "This is the end of the number," so it would stop there. And when it had done that, the address registers had been incremented by however long that word was. So they sat on the next one. If you wanted to add up two columns of numbers, you only had to give the first address of the column, and then you'd say, "Add 200 to 300." You'd say "add, add, add, add" and each "add" was one digit. So it had some very interesting features to make it fast and cheap.

The stack machine was a fascinating architecture. I really wanted it to be a stack machine, so I basically came up with an architecture that would do essentially what the 1401 would do, but it had some other instructions that would leave the registers set in a certain way. You see the same thing running a stacking memory, where you go up and down doing exactly the same thing, so it would get a stack that way. Furthermore, it was designed so that when you had more money, you could bring many of these registers into the CPU and keep the stuff current there—you'd call it a cache today. So it could run much faster. You'd get all the advantages of running on a much faster machine, but it would have the same instructions.

HAIGH: So just to get the context on this clear, what you're describing isn't the SCAMP machine; it's a separate design you came up with as an entry to a kind of bake-off contest within IBM.

GEAR: Yes.

HAIGH: Was this a contest to design what would become the 360 architecture, or was it separate?

GEAR: No, the context of it was to propose designs for what was to be the 360 series.

HAIGH: So this kind of committee of people getting together—I think it's called "Spread." I've seen that as a reference to the design that became the 360.

GEAR: I don't remember. That sounds vaguely familiar, now that you mention it. Oh, I think that's right, because there was a "Stretch" going on, and this was spreading it out. I think you're right.

HAIGH: So involved with that were also the General Products Division and the Data Systems division in Poughkeepsie, who both have their own ideas about how things should be done. So there were three groups involved with that project.

GEAR: At least three groups, yes.

HAIGH: There was an article that describes all that. It's been described in a number of places, but a nice one is from Bob L. Evans.

GEAR: Okay. I think he was in charge of that whole division at that time.

HAIGH: Published in *Annals of History of Computing* in 1986.

GEAR: One of the interesting things I remember about that, by the way, was the big debate over six-bit or eight-bit. If you remember at that time, IBM was all six-bit characters. There was a lot of desire to go to eight-bit. There are not enough combinations to six-bit for the characters that are wanted, etc. It was a big, big battle. I cannot remember anymore which side the various people were on, but what I remember was that it alternated all the way up the management chain. I think the committee was probably split, because there were probably those concerned with marketing who were more concerned about six-bit compatibility. Those people who were more abstract-bent, such as me, were more interested in what the future was. I don't remember which side Amdahl was on anymore. What I do recall was, whatever side Amdahl was on, his boss Brooks was on the other side. Brooks' boss, who might have been Evans or there may have been someone else in between, was on the other side, so it oscillated all the way up. So it was very difficult to reach a decision for quite some time.

HAIGH: And were you flying to America for these meetings?

GEAR: Yes, several times…which was pretty exciting for me, in those days?

HAIGH: Are you aware of any of the ideas that you or the other people on the British team might have come up with that actually made their way into the 360 design, or at least influenced it?

GEAR: Well, the implementation of the Model 30 was almost pure Hursley. It had the transformer read-only memory. Well, I think higher levels, some of those might be micro-programmed, but the Model 30 had the transformer read-only memory, which came out of Hursley. That was a physical implementation. The transformer read-only memory—it sounds unbelievable these days—basically, think of having a row of transformers, and a wire for each word, and then you wind the wire either inside or outside of the transformer. So when you drive a current down the wire, the transformer either sees it or doesn't. In the Hursley machine, these transformers were arranged around the outside of a wheel type of assembly. Then we built a

machine that spun this wheel, and it wove the wires in and out. By the time it was put into the 360, they were aligned and the wires were printed on sheets of Mylar and laid in. It was the same basic idea.

HAIGH: All right. So after the overall 360 architecture, in terms of the instruction set and byte length and so on had been set, some of these ideas resurfaced in implementing it for the lowest-end machine, was it the 30? Was that the smallest one in the range?

GEAR: [Model] 30 I think was initially the smallest machine. There may have been a smaller one later, I don't know. I lost interest in commercial machines after I left. I left before much of the architecture had been designed.

HAIGH: In terms of the architectural description for the range as a whole, you're not aware of any of the ideas you might have presented to the committee that they would actually approve?

GEAR: I did not see them in the final design.

HAIGH: Well, that might be a good point at which to break, then.

Session 2 begins on the afternoon of September 17, 2005.

HAIGH: Now you've just discussed your work at IBM Hursley and your involvement with what became the Spread project to design what became the 360 architecture. So at this point you were still working very much with computer architecture. Now, was Amdahl already very famous at this point?

GEAR: No, he was probably not very well known outside of IBM at that point. There was a tendency for people in companies like IBM not to be too well known outside of them. I think he probably became famous after he left IBM.

HAIGH: Did you have a sense that, at this relatively early stage in your career, you were working with the top people in the field?

GEAR: No, I guess I never thought about that. It was fun work.

HAIGH: You had also mentioned earlier your general sense of frustration with England upon your return. How long was it before you were beginning to plan your escape?

GEAR: Originally I was planning my escape shortly after I got there; I guess I decided I probably wanted to go back. It was clear that the activity in computers was in the U.S., and I saw nothing in England to change that view. I think I was there because I felt an obligation to go back and see my parents, etc., etc. So my original plan was to spend the two years there and go back. I had moved over to England, I think, in August -- towards the end of the summer, anyway. So I would have normally gone back in August of 1962. In terms of going back, I found out that the time I spent in the U.S. on those trips for the 360 design committee counted against the two years, so it had to be a cumulative two years out. Later on this presented a problem; I actually returned in the early summer of '62. I hadn't decided to go back to Illinois. They wanted me to go there and teach a summer class. It turned out that even to go back for the regular semester, I still had the problem with the two years out. So they agreed to seek a waiver, because I would be working under an AEC grant. They said that they could get it. So I was going to get a waiver for the end of the summer. If I was going to get a waiver for the end of the summer, I might as well get a waiver for the beginning of the summer.

HAIGH: So then you returned at the beginning of the summer in 1962?

GEAR: Yes.

HAIGH: Now, did you consider other possible destinations within the U.S.?

GEAR: Well, Illinois had offered me a job to stay there before I left, so that was one obvious choice. I can't say that pursued other universities particularly. I've never been very active in doing this sort of thing. When I started to indicate that I was probably going to go back to Illinois, IBM promptly offered me a position in Yorktown at the research center, which would have been a much better salary…or even in Poughkeepsie, I think, to work on the 360 design. Somehow it didn't seem quite as interesting to me. I think I wanted to go back to the university, basically.

HAIGH: So what was it that you found more attractive about the university environment as opposed to a corporate research lab?

GEAR: We used to joke about IBM, and that it meant "I've Been Moved." At one point when I was working on a part of the SCAMP design (the machine that never appeared), and I went off to Paris. In that case it wasn't to use the computer; I went off to talk to different customer-type people within the company about what might be needed. When I got back I found out that I had a totally new position that made that last trip completely irrelevant. I was now in charge of the cost reduction redesign. I like to follow things through more to the end, and I didn't have the satisfaction of really finishing stuff and really seeing it come into realization at IBM. I felt there was a greater opportunity for that at a university. And at Illinois, ILLIAC II was close to coming online and I had a lot of my soul was in ILLIAC II. So it was a natural place to want to go back to. Plus I knew people there. Plus I had left a small amount of furniture there with friends!

HAIGH: So at that point, was it considered unusual for departments to hire their own graduates into tenure-track positions?

GEAR: Well, I don't recall that at that time. But remember, there were so few computer scientists in those days that there weren't many choices for hiring within that field. Furthermore, I think that the attitude is that, if someone goes away for a few years, it's not such a problem.

HAIGH: Right, you've been laundered.

GEAR: Yes, put it that way. Been through the wringer, at least.

HAIGH: All right. When you returned, was it to the mathematics department in terms of your appointment?

GEAR: Well, the appointment had to be in some discipline, so my appointment was as Assistant Professor of Mathematics (it might have said Applied Mathematics) in the Digital Computer Laboratory. And the appointment stayed that way. Then when it became the department of computer science sometime in the middle 1960s, it changed to Assistant Professor (or maybe it was Associate Professor by then) of Computer Science and Applied Mathematics.

HAIGH: So it would now have been around 1964?

GEAR: I don't recall; I would have said that it was '66, but I don't remember.

HAIGH: Had anything important changed in the laboratory or at the university more generally, in your absence?

GEAR: A new building had been built for the digital computing laboratory shortly before I left. It wasn't a very big building because it wasn't a very big department. My final exam was in the

one room that was a combination library/conference room/meeting room. My office, which was a bullpen for about six grad students, was next door to it. I took my prelims there earlier. I just froze up on a number of things that I should have been able to answer easily. At one point, I remember Doob said, "Why don't you go outside and get a drink of water and come back in?" And then after I left the room I went to my office, which was right next door, and I could hear a little scribbling on the blackboard—you could hear this little chalk noise through the wall—and then laughter. I thought, oh my God, what are they saying? But apparently they were talking about one of the questions that they had asked.

HAIGH: That was the oral examination?

GEAR: Yes, the oral exams. That would have been in the prelims, I think. Anyhow, the new building was finished. So when I got back the ILLIAC II was semi-operational, so that was the thing I immediately got involved in. The university continued to grow the whole time I was there; I don't remember any other big changes. Of course the big change from when I went there in 1956 the first time was when they took the top 50% of any high school graduating class to when I left in 1990, where if you were in the top 5% of your graduating class, you might have a chance of getting in, provided you had outstanding ACT scores. Just an unbelievable change. I remember the engineering college where I was in the advisory committee for a while, once the dean decided to find out whether we were we getting the top ACT scorers. The ACT score max is 36 on math; it's a different thing than the SAT. When we looked and found out, indeed, we were, in fact, getting all the 36 ACTs, and most of the ACTs were up there in math. There were outstanding grads coming in by then.

HAIGH: So once the ILLIAC II was finished, was the hardware reliable in operation?

GEAR: After the first year, maybe, it got pretty reliable. It seemed to me that we went through a period where there were some failures. You often have a burn-in period. I think it's less of a problem today, although it may still be. Certainly in those days, if you made something electronic—if you look at the failure rate, there's a big flock of failures in the very early days, then stuff lasts pretty well, and then some stuff lasts forever. It's extremely bimodal. So when you build something new, you tend to be losing stuff early on. Then you get rid of the bad components; basically they're near failure all the time. Then you get extreme reliability, and we found that with ILLIAC I, for example.

HAIGH: So once you got past that initial burn-in period, things were working nicely at the hardware level. I think you had mentioned that an obvious problem was that there was no software for the machine whatsoever. So when you returned, did you have the idea that this would be something that you would be working on?

GEAR: I think I was assigned to do it. Maybe in those days, you assigned faculty to do things a little more that you do these days. The research support was kind of made by AEC, and they were now interested in software for the machine. So it was clearly where the money was. Initially I was not in charge of the financial issue; I just was one of the people on the ground, so I didn't really have to worry about those issues. I was interested in it, so I started writing first of all an assembler so we could use the machine more effectively, then a compiler, and then an operating system, and then eventually a time-sharing system.

HAIGH: How long did those things take to do?

GEAR: I did this simultaneous with doing work in ODEs and stuff, which was an interest that was developing them. I would have to go back and look at records. I think the assembler was less

than a year. I basically wrote a really crummy initial version. It was in binary, which was the only loader we had. And by basically writing in a whole bunch of little bits that I put in different parts of the memory I had room to stick stuff in when I wanted to patch, then used it to assemble itself. I'd get an initial version and then kept using it to keep bootstrapping to get better and better versions. Once that was done, we needed a minimal operating system, obviously. IBM had given us a 1401, which we used offline to load tapes. So it would read that stuff and start the jobs and stop them and so on. Then we added a FORTRAN compiler to it, then we added some additional channels. We added a PDP-11 with graphics facility on it for some research, and we started to write a modest time-sharing system for it. It didn't support many terminals, but it just let people get at it that way. It was initially extremely low level, but at that time most other ones were too.

HAIGH: What did people mean in those days when they said "operating system?" I got the impression that in the early days, people thought of it as a bit more of a system that would do some of the chores that the operator had to do before, in terms of going from one job to another, and maybe doing some of the basic things to configure the machine and software instead of hardware.

GEAR: Essentially that's correct. Basically in the early days, we're now working with card-oriented systems so it was a little easier. You had a deck with a FORTRAN program and you had a deck maybe with an assembly language subroutine, and you needed a few library programs, and by then there was some secondary storage or tape in library code and so on. You needed various things like that, but it was still one job at a time on the computer. There was no time-sharing yet in terms of having multiple jobs on it. I don't want to say the "culmination" of this approach was JCL, let's say the "abomination" was JCL—IBM's operating system for the 360.

So in those days an operating system for a typical job would be, first of all, there was your I.D. card that served as a verification that it would check as an authorization to use it and how much time you had left (or whatever was needed to be accounted). Then it might be a card that might say something like, "What's coming next is a FORTRAN program I want you to compile." Maybe whose name it is, or maybe that just came out of the FORTRAN compiler. And then it would say, "Here's an Assembly language program," and then, "Oh, and I need these library programs," square root or whatever it was. And then maybe something that was probably the instruction to the loader, "Now load that program." We didn't really have disks back then, but later on you'd say, "I've got this one on disk that I want to load with it; I want to load these subroutines. Now execute it. Start in such-and-such a location or start in such-and-such a program. And here's my data." So it was just a serial input, what the operator would have done in the old days. Then there was "the end" card, or the end of file for the whole deck so that when the program failed, the computer could say, "Okay, that one's gone; let me start working on the next job."

HAIGH: In those days would people consider generalized input/output routines as the core parts of the operating system? Or was it that once a job was loaded, it just had the whole machine until it was finished?

GEAR: Yes, you had the whole machine until it was finished. I don't remember when that started to change. The idea of time-sharing, the idea of remote terminals, was really not separate early on; they were just one and the same thing. But it started to come in, the idea of running multiple jobs and overlapping the jobs, because one was using one resource. So at that time it was then that the operating system had to start worrying about allocating resources to different

jobs. By that time you would start to say, “Okay, I need to mount such-and-such tape unit,” or you couldn’t get it unless it was available, etc. So they kept expanding it with more and more stuff; the initial operating systems were unbelievably simple.

HAIGH: Okay. So this virtualization of input and output devices really only came along with time-sharing.

GEAR: Yes, because it all started about the same time we started getting more input/output devices. There were tape units, but one was still addressing those by the physical numbers: you say, “Now, I want to write on tape unit ten. Please rewind the tape. Now I want to write this physical record.” It was not a high-level symbolic thing. If you used disk, when we eventually did get a disk drive, you would specify the disk track, platter, and sector number you wanted to write on. So I’m saying that later on, operating systems started to remove all that low-level stuff from the user.

HAIGH: Obviously the loader would be part of the operating system; would you have thought of the assembler as being part of the operating system?

GEAR: No, I’m not certain what we called it then. I think in those days what we thought of as the operating system was just that thing that took over the old operator. We had the assemblers while we still had computer operators. You have to remember that, in those days, the memory was still extremely small. Even the largest computer, like the 7090, had 32,000 words of 36-bit memory. You could not afford to keep much that wasn’t part of your job in the memory.

HAIGH: Did you have a resident monitor?

GEAR: Yes, something like 2000 bytes for it. We had, what, 8000 words of 52-bit words, about 64,000 bytes of main memory, which wasn’t much. You didn’t want to use up much. So all the monitor did was to grab -- maybe it took up a little bit more but the memory was blocked into 256 words, and I think they used just 256 words -- which would grab an interrupt, and if there was something wrong, write out a block from the user program and bring in a block of my stuff to try and deal with it from the drum. We didn’t have paging memory on that machine, although paging had already been invented by the Manchester machine before that.

HAIGH: The Atlas.

GEAR: Right.

HAIGH: So that’s the system software in the machine. Now the money to build it had come from AEC. Did they take time on the machine to run any jobs, or was it completely at the disposal of the university to do anything they wanted on it?

GEAR: It was completely at the disposal of the university. I think, first of all, that the AEC was interested in promoting new computer designs, and who knows what main reasons they had in mind at that time—they wanted ones that they could run at the University of Illinois. Secondly, I think, as I said earlier, the great strength of the U.S. system was that people were really supporting universities because the biggest benefit they got was not that machines were designed or theorems were proved, but the graduate students that we produced that provided for the needs of the DOEs and the DARPAs and the companies in the country.

HAIGH: So no one working on the project would ever need to get security clearance or anything like that?

GEAR: No. I don't think we would have accepted any security work in the department. Although the ILLIAC I was doing security-cleared work in the early days, because it was shared by something that was called Control Systems Labs in the old days (later changed its name to Coordinated Science Labs). It was doing radar signal processing in the early days, which was military. Every afternoon at about noon they would come down and seal off the place with guards on the doors, take over the machine, and run it for a few hours.

HAIGH: But by this time they had their own computers?

GEAR: Yes.

HAIGH: So what kinds of applications was the machine used for?

GEAR: For a while we let public people use it. In fact, we tried to let other people use it most of the time. A number of people in the department did their theses on it; I don't know what all those were. I don't really have any details of the actual applications used other than what my thesis students were doing on it. It was probably being used for experimental software development and method development more than it was for actual production code.

HAIGH: So was there another computer on campus?

GEAR: Oh, yes, by then there was a 7094. A 7094 when it first came on; then the 360 came in 1965 or so, I forget exactly when. And then the other departments started getting computers. When the minicomputers arrived in the later '60s, they started to go all over the place.

HAIGH: So the ILLIAC II was never in the main computer center?

GEAR: No.

HAIGH: Now, you did have some early publications and presentations on the compilation and systems work that you did.

GEAR: Actually that was the first paper I ever published. [Optimization of the Address Field Compilation in the ILLIAC II Assembler, Comp. Journ., 6, #4, pp332-335, 1964].

HAIGH: And that was in 1964, so at that point, you had finished your PhD back in 1960; you had come back and you were two years into your faculty appointment. So was there less pressure to publish in those days if you were doing this very useful, practical work?

GEAR: Well, whatever you were doing—much less pressure. These days you wouldn't hire someone with a publication list that I had had when I was hired, which was zero. Graduate students didn't publish; there was no great pressure to publish. Really, there were not many outlets for this stuff. I think that one was in *The Computer Journal*, which was a British publication. There wasn't much available then. So that was essentially the first publishable piece of work that I did at Illinois, and it represented a vast amount of work. Nowadays that's years of hard work.

HAIGH: Nowadays that would have been 15 papers or something. And your first publication in a conference proceedings wasn't until two years later; that was in 1966.

GEAR: Well, in those days, graduate students didn't go to conferences. It would have been unheard of to take funds from a grant to send a graduate student to a conference; even faculty, hardly. Now that was probably the first conference I went to, four years after I had been on the faculty. About once a year we had a joint meeting of the various AEC contractors doing machine development; I'd go to those. But I hardly ever went to any conferences until I had been there

several years. There just weren't that many of them at that time, anyway. About that time I guess I decided to start publishing, but there was no great pressure. I was promoted to Associate Professor in 1965 or '66, with almost no publications.

HAIGH: That's true, actually with two journal papers…

GEAR: Well, I would not have gotten promoted these days!

HAIGH: That's true. They might have wondered why you still hadn't presented at a conference.

GEAR: [Chuckles] Well, why I put in that piece of paper at a conference was I thought it was an interesting idea. That was 1966. I can't remember exactly what was in there, but I had gotten very interested in ordinary differential equations by then, and started to look at how to integrate them pretty automatically, and that was the start of that work. I was also interested in compilers, so here was the situation where I was typing in the equation symbolically—like typing it $y' = \sin(3x \times y)$ blah-blah-blah, saying I want to know what the answer is between 0 and 1 with these initial values, or something like that. It involved compilation, numerical analysis. So it really combined a lot of my interests.

HAIGH: I think I have that paper here. The system is called *MAP*—is that the one? Here we are [Numerical Integration of Ordinary Differential Equations at a Remote Terminal, Proceedings 21st National ACM Conference, pp-49, 1966].

GEAR: Yes, so it was essentially a symbolic one. I remember a little bit about it. One of the other things that I had done was trying to develop some minimal time sharing on ILLIAC I. I had been to another meeting, the National Computer Conferences (Fall Joint or Spring Joint Computer Conferences) which used to run in those days. At some point around then, General Electric had shown off its early Basic system, which was developed at Dartmouth, which they were peddling. That must have been around that time period, and that fascinated me. I came back and sort of over a weekend wrote a very primitive interpretive system that did some of that for the ILLIAC. Initially it was single-user, and then we made it into a multi-user. It was very much modeled after what I saw in that Basic system.

HAIGH: So I see in here that it looks like it was interactive—it had a lot of prompting, then it would draw a little graph with axes. So presumably it was coming out of a teletype.

GEAR: Right, that was all there was in the way of a plotter then. So I had written this small interpretive system based on what I had seen in that Dartmouth Basic system to allow very simple computations to be done. It was a very simple top-down compiler. It was very fast. It didn't require a lot of work; like I say, it was a weekend job to do that. Since I was interested in differential equations, I decided that I could take in differential equations this way and treat them in pretty much the same way with a different structure, linking to the codes that I was working on at the time with differential equations.

HAIGH: So prior to creating this interactive system, you had already working on differential equation codes?

GEAR: Yes. Mainly, then, I guess I had two parallel efforts. My research support at Illinois, almost the whole time I was there—once I took over running the grant, which happened in the middle '60s—had two streams to it: computer systems-type work and numerical analysis-type work. These blended together in a number of areas, like this. On the systems side I developed that small interpretive compiler and stuff, and in the summer of 1965 I had gone to Argonne.

Well, let me back up a bit. In 1962 or maybe ’63, Leslie Fox visited, and I got more interested in ordinary differential equations from talking with him. When I first went back to Illinois in 1962; Abe Taub, who was still there (he left shortly afterward) threw a problem at me that one of his students who had not finished (most of his students never finished) and he wanted me to look at. It was a problem in general relativity, but it was one-dimensional—it was an ODE. I started looking at that, so I got very interested in some ODE stuff. I think I wrote a paper at that point, somewhere in 1964 or something. Then I went to Argonne in the summer of ’65 and essentially worked on ODEs there. Argonne was a great place at that time. You went up, you had a desk, you interacted with people, and they really didn’t care what you did. They were interested with you interacting, helping them with their problems, whatever they were.

HAIGH: Were there many people from University of Illinois Urbana-Champaign going to Argonne?

GEAR: There was one guy named Lloyd Fosdick, who later became head at Colorado and is now retired, but was at Illinois in those days. He would spend time there. But there were people from Northwestern, from Chicago, Purdue. Walter Gautschi was there some of the time. Lots of people. I met Beresford Parlett there for the first time, from Berkeley. Lots and lots of people there; it was a great place, a lot of interaction. The first year I worked on differential equations, and that became a paper, integrating differential equations of various orders. I had been working on the Nordsieck scheme, which was a way of representing differential equations using scaled derivatives. He had been at Illinois in the Control Systems Lab, and he had written a program, a variable-step Adams code. The idea was, instead of using various past points, he computed the scaled derivatives of the polynomial through those points. Because if you change the step size—scaling included the step size—all you had to do was multiply them by the appropriate power of the step size. So what he stored was y, hy', h^2y'', and so on. Pretty much the terms of the Taylor series. I used that quite a bit in my codes in the future. DIFSUB used that.

[Tape 3, Side A]

GEAR: Okay, so I was talking about the Nordsieck vector, and it created this way for making a variable-step implementation of a multi-step method a lot easier, at least in the days of small-memory computers and slow computers. I exploited that by adding equations of various orders. Because one was storing the higher derivatives—like when you handle a fourth-order equation, you had the fourth derivative there, directly relevant to the system. So that’s what I worked on in Argonne for the summer. I wrote a paper that appeared in *Mathematics of Computation*, I think. That was probably about the time when I started writing things.

HAIGH: Now is that “Numerical Integration of Ordinary Differential Equations”? [Numerical Integration of Ordinary Differential Equations, Math Comp, 21, #98, pp146-156, 1967]

GEAR: Yes. So that was why that was creeping into what I was doing at that time. Sometime around then, I think it may have been a year later, was the one paper I had that was rejected for publication and I never chose to pursue. It probably happened after the stiff equations, but I wrote a much bigger program called ODYSSEY—Ordinary Differential Equations Solver System. It also incorporated stiff methods, which came about a year later. This was a batch processing program that took in much more complicated systems and processed them and ran the results automatically. It was rejected by CACM [Communications of the ACM]. Publication was not so important in those days that I thought, I’m not going to spend my time rewriting it—I’ve done my duty.

HAIGH: So how long did you continue to spend your summers at Argonne?

GEAR: Just two summers. Also, for about a couple of years there, 1965 through'66 and maybe on through '67, I was also a "consultant" at Argonne. And this meant that you went up one day a week and sat in an office there and talked to anybody that wanted to talk to you, or you worked on your own thing and interacted with people. It was pretty nice. Lloyd Fosdick and I used to go up together. It was kind of a hassle. We had to catch the 6 a.m. train out of Champaign. In fact, I remember one of the years we also agreed to teach an off-campus course in the evening at General Telephone and Electronics [GTE], which was outside of Chicago in those days. So we spent the day at Argonne and then took a car over to the western suburbs near the airport, teach a three-hour course in the evening, and then catch the plane back to Champaign that got in at midnight. By then I was pretty shattered.

So in the summer of 1966 I also spent some time in Argonne. That's where I got into stiff equations; I didn't know what the name was then. Actually what I was doing at that time was struggling with what turned out to be differential-algebraic equations, though I didn't realize it at the time. What was called then the MCD, Math and Computation Division, which provided both computer services and mathematical services, was next door to a building that I think was chemistry and it was joined to it. And there was an analog computer in the link between them. And the guy who ran that stuck his head in my door one day and said, "I've got the sort of problem you digital guys can't solve."

As I said, I'm a very competitive sort, and that was the sort of challenge I couldn't pass up. He was solving analog problems for people in the reactor division. I believe the particular problem I looked at was for the failure of the iron containment vessels. The problem involved iron and oxygen and some water—there were very few, seven, radicals maybe, and some energy input, and [finding] what was happening. And they were stiff equations, which I later found out when I found out what the word was. They couldn't solve them on the [digital] computer. So I got the equations and started looking at them and what the problem was. One of my failings, maybe, over the years is I'm always much more interested in digging into the stuff myself than I am in going and reading all the stuff about it. I think I stray too much on one side; I know plenty of people who stray too much on the other side and never get anything done because they're always reading. I don't think I strike a happy medium. But I looked at Dahlquist's paper [G. Dahlquist. "A special stability problem for linear multistep methods," BIT (1963), Vol. 3, p. 27], and that's where noticed all about stiffness and so on. I got a little bit of an idea. But I was very much interested in modern stiff methods at the time.

HAIGH: Now this phrase, "stiff differential equations," did Dahlquist come up with that, or is that in the earlier paper by Curtiss and Hirschfelder, do you know? [C. F. Curtiss and J. O. Hirschfelder, "Integration of stiff equations," *Proceedings U. S. National Academy of Science*, (1952), Vol. 38, pp. 235-243 – on reflection the title appears to answer this question].

GEAR: I'd have to go back and look; I don't remember. I think it was pre-Dahlquist. The name "stiff" just gets used for all sorts of different things now, including oscillatory problems, because it came to be just a vast separation of the eigenvalues, which could lead to oscillations as well. At that time, it was very much the fast decaying components fostered new interest in it. The example I always gave to people was it's the effect of having a very stiff dashpot—that is, a shock absorber. The shock absorber absorbs energy and stops your car from bouncing up and down. If you take a spring dashpot system and make the dashpot stronger and stronger, it becomes stiffer and stiffer.

HAIGH: You mentioned the analog differential analyzers that had been used, that you worked them by having a trace for the input and the output. Was the idea of "stiffness" coming because the input tracer was moving more stiffly as the system struggled to integrate an equation?

GEAR: Well, who knows? The word may have had earlier origins, but by the late '60s it very much used in the context of rapidly decaying components. The differential analyzer, which was a way of solving differential equations, is an analog computer, but it was pretty limited in what you could do. I'm talking about an analog computer that was electronic. It has adders and subtractors and amplifiers and integrators and so on. So you simply plug together -- plus in the old days you were still using patch cords to plug together equations -- and then run this simulator of differential equation, the analog, in other words. Interestingly, the analog computer comes from the fact that it is an "analog" of the system. Now, we think of analog versus digital as being the question of whether you have a continuous variable or a discrete variable. Analog computers could handle stiffer equations; these things didn't bother them for some technical reasons.

They very much bothered digital computers at the time—they didn't bother the computer, they bothered the method. It was just way, way too expensive to solve a stiff equation. To do anything, you had to run a step size that was akin to the shortest component present, which could be 10^6 or 10^{12} shorter than the components you're interested in. So there were still serious proposals, even into the early 1970s, to spend multimillion dollars for analog computers to solve special problems, mainly in the defense arena, where people were prepared to throw money. People didn't yet know that it was no longer a real difficulty.

So I got interested in that problem, and I was interested in using multi-step methods. I had done a lot of study of the stability issues of multi-step methods, and this, of course, is what the whole issue is in stiffness. Dahlquist had this very nice theory about A-stable methods, what we call the "second Dahlquist stability limit," which is that the maximum order of an A-stable multi-step method is two. One clearly wanted higher order methods than that, so the question was what to do? So I started looking at what happened as you did various things with the coefficients of multi-step methods. In a multi-step method, you have past derivatives and past function values all to use as information about what you might do. Because the first Dahlquist stability limit criteria, or barrier—I guess *Dahlquist barrier*, we used to call it—which says that the maximum stable order of a multi-step method, whether it's stiff or not, is basically the number of steps plus one, and you've got twice the number of free parameters. You have a number of free parameters.

In fact, one of the books that I read very early in my career was by Hamming. I would say in some ways it was a terrible book—it's got lots of errors and misunderstandings—but it gave me outstanding insight into the whole thing. I forget the exact title, but it was about numerical integration. He spent a lot of time looking at how, as you varied these free parameters, you could change the characteristics of the method. The problem with the book was that some of his conclusions were totally wrong because he used the wrong scaling for the error coefficient and stuff like that. Although the book in detail was terrible, in terms of the understanding it gave, it was outstanding. That was a very valuable book to me in those days, much more than Henrici's book, which was full of mathematical detail. I think he wanted to show he was really a mathematician.

HAIGH: Is that book *Discrete Variable Methods in Ordinary Differential Equations*, Wiley, 1962?

GEAR: Right. A very good book—very mathematical and not much insight, if you understand what I mean there. Hamming explained it but he didn't get the details right.

HAIGH: So just a big picture question in terms of the numerical solution of differential equations on digital computers. Clearly at this point, these stiff equations were giving real problems. Would you say that in general people were fairly satisfied with the kinds of results and performance that they were getting? So you have the idea that there's a whole bunch of equations we can solve and we are generally happy with that and we know how to write the software for those, and then there's this much smaller class of stiff problems with which unfortunately these methods don't for work.

GEAR: Yes, I think that's true. By today's standards, the methods were pretty primitive, but they satisfied the needs at the time. It was just stiff equations—but a lot of the important problems were stiff: electrical networks were getting bigger and bigger; you had small capacitors, they become stiff. Chemical kinetics was the biggest area, probably. A lot of mechanical systems are stiff. So there was getting to be an increasing interest in those, which is why I was lucky to be in the right place at the right time.

HAIGH: You had mentioned there that as a digital computer person, you were being taunted by this analog computer fan who thought that digital computers couldn't solve his problem, and presumably he thought that they wouldn't be able to in the future either. Was this a common view, that no one was sure that digital computers would ever be able to handle these? Or did most people think that it was just a matter of time before people came up with methods that would be satisfactory?

GEAR: I think in the computer world, we're always optimistic that we're going to find a way to do it. I don't think there are many problems where we'll say, "We'll never be able to do that." In fact, I think the opposite is usually true—I've sat in too many idiot conferences where people said, "Well, computers get a little bit faster…" I sat in a conference once in a simulation where the guy was dismissing the need to worry about stiff equations, because in another five years computers will be this much faster and there will be no problem. He doesn't have any comprehension that the speed issues in stiff equations go up exponentially if you don't deal with them directly. So that's nuts.

HAIGH: Although, since maybe the late 1960s, people have viewed the complexity of algorithms as the core idea in computer science. I suppose in the early '60s, the idea just wasn't as widely understood and as clear conceptually.

GEAR: Yes, when I said exponentially there, it's not in the sense of computational complexity (in terms of [which] if you double the first eigenvalue to make it twice as big, you don't just need double the time, you need enormously more). When the method is unstable (dues to stiffness or anything else) the error increases exponentially with the step number. There was also the issue of if you took small steps, you lost accuracy. I think he also said that we'll get the word length longer, and that will take care of it. That's also equally nonsense, when you're thinking about looking at a thousand times more bits to take care of some of these issues.

But I don't think anybody didn't think we would be able to solve it. And there were a lot of papers in those days on stiff equations, suggesting various methods. They were localized methods: here's a way that you might be able to solve this little problem; you could tailor it very carefully and you could solve it. The challenge always, of course, is to provide a piece of code

that will tackle most stiff problems. That's a challenge for any piece of numerical software: it will go in and not balk on most problems that it's given.

HAIGH: And was it generally understood that the way to tackle it was going to be to come up with some satisfactory method of changing the step size appropriately?

GEAR: The step size selection, and later the order selection (which is also stuff I wrote on a bit later) was just an issue as we moved into the mathematical software era from the algorithms era. You no longer gave the user a prescription for a code to write and say, "Choose the step size." The user was too removed from that. So you had to find ways of figuring out these parameters automatically, where possible. So this was just evolving, and it needed to evolve also for stiff equations. In fact, the Nordsieck code, very early on, was already choosing step size automatically. Lots of programs in the 1960s for simple cases chose step sizes automatically by various tricks. They were just getting more and more sophisticated. At that time, multi-step methods seemed to be more efficient. I still think they generally are, though Runge-Kutta methods have a lot of advantages for other reasons, particularly if there are discontinuities.

So I was very interested in trying to use multi-step methods, because by bringing in more information from the past in each step, you have a chance of increasing the accuracy. I was studying the stability curves for all sorts of possibilities. Now, I don't want to get into too much technical detail. When you take a multi-step method, you basically have two stability extremes. One is when you're just solving $y' = 0$, so there's no differential equation at all. You have one stability polynomial, which has to do with how you use the function values. You're interested in the roots of that. In Adams' method, you have to have one root of one in order for it to be consistent. It has to be consistent to be accurate. Then all the other roots are ones that could possibly cause you problems as you wander around. Where you have Adams' method, all those other roots are zero for that, so it's pretty nice. It's very good for things that are not stiff. Then you have the other polynomial, which is how you use the derivatives and where its roots are. What happens when you take a problem like $y' = \lambda y$, as λ varies from 0 to∞, is that the stability polynomial you're looking at moves from the stability polynomial for the function values to the one for the derivatives. And the extreme is the one for the derivatives. One is interested in how long those roots stay inside the unit circle so that it remains stable. That's how far you can go before you lose stability. Dahlquist's important theorem, the second Dahlquist barrier, as we called it, was if the order is greater than two, they have got to go outside at some point. So clearly, if you're going to a have a higher order than second, they're going to go outside sometime.

I thought about a couple of things. One was, I really want very large roots to be stable. This is what the character is of stiff problems. When I'm looking mostly at the polynomial due to the derivatives, I'd like those roots to be as small as possible. Well, the natural place to put them, since you've got the choice of all of them, is zero. That's about as far away from the stability boundary of the unit circle as you can get. What I found is, of course, it's not A-stable, which means that it would be stable in the whole left half plane, which is ideally what you would like. What I found was, by just plotting them, that for all orders up to six, they retain stability for the vast majority of the left half plane, and for a piece near the origin.

HAIGH: And you found this, basically, experimentally, then?

GEAR: Well, by computing the stability regions; not by running--

HAIGH: So that's what you mean. You did some mathematical things to figure out an equation that would give you the stability regions.

GEAR: Yes. It was not very complex. Right. Then I defined a term called *stiff stability*: this is making the definition fit the needs. The ideal would be to be stable in the whole left half plane. If I take the problem $y' = \lambda y$, with λ in the left half plane—it decays. If the λ is up near the imaginary axis and big, it oscillates for a long time till it decays. So I'm never going to solve problems high up near the imaginary axis with a large step size because in one step it would be making many oscillations. If I'm interested in that behavior, I'm going to track it slowly. I'm not really interested in this region up here. There's the solution I should be following, but if it's far out in the negative half plane and very strongly damped, I just want it to be stable. If it's around in here, near the origin, I would like it to be fairly accurate. So I termed this "stiff stability," rather than A-stability, and showed that the BDF methods—these turned out to be BDF methods later on—had that property. In fact, had I taken the time to look back at the Curtiss and Hirschfelder paper, I might have gotten there a lot faster, because they mentioned them as being useful for this.

So that's how I got there. I remember one of the first times I met Dahlquist; we were both talking about this. He claims I made this statement first; I claim he made it first in a meeting which I thought was in IBM at Yorktown, but I don't remember. I said, "All one wants is fidelity in this region, and lack of adultery in this region." Now I claim that remark was due to him; in fact, I think it was at his seventieth birthday party, I repeated it and he said, "No, that remark was due to you." I don't remember.

HAIGH: All right. Is that what becomes DIFFSUB? The implementation of this?

GEAR: Well, DIFFSUB involved the implementation of both the Adams methods and the BDF methods—you could choose—for stiff or non-stiff equations, using the Nordsieck vector scheme for easy change of both order and step size. At some point at that time, I had also seen a paper by Krogh about variable order methods. He was using a much more complicated, but actually better representation, of divided differences to do it. Clearly variable order was valuable, but there's no reason to ask the user what order to use. If you are a sociologist who wants to solve this differential equation why would you have any idea of that, any more than what step size to use? It's clear that the method needs to work those parameters out. Nowadays you would also say that the method needs to figure out whether it's stiff or non-stiff, and everything else, but one step at a time. This [DIFSUB] had variable order, variable step size, stiff and non-stiff. The user had to specify whether it was stiff or non-stiff. You could stop and switch, and then continue integration.

HAIGH: I'm looking at your account of this from the 1987 History of Scientific Computing conference. [C.W. Gear & R.D. Skeel, "The Development of ODE Methods: A Symbiosis between Hardware and Numerical Methods," *Proceedings of the ACM conference on History of scientific and numeric computation*, 1987, archived in ACM Digital Library]. You have the kind of same story here with Nordsieck.

So you say that the first implementation of that was at Argonne in the summer of 1966. Did that program even have a name?

GEAR: No, it was just a very early test program.

HAIGH: So nobody else ever really saw this program; nobody ever used it for anything?

GEAR: It was used to run equations for that reactor problem I mentioned. In fact, when I went back to Illinois after, I continued to run their equations for them for a while. A very interesting thing happened there. In the days of analog computers, one of the things they used to check their calculations was the mass balance relationships. In chemical kinetics, you're modeling, in effect, the number of OH ions present, oxygen atoms present, and so on, in a mixture. It's fairly obvious that if you're really doing it exactly, the number of oxygen atoms shouldn't change because you're only simulating chemical reactions; you're not stripping electrons off the outer shell and changing them into something else. Although the equations, of course, are simulating the percentage concentrations rather than the integer numbers, nonetheless… I mean, if you say I've got this much water, this much hydrogen peroxide, this much oxygen, you can say that, okay, there's one oxygen there, there's two oxygen atoms there—this sum is the number of oxygen atoms present, and it should be constant. That's called a "mass balance equation." On an analog computer, they would use those relationships, look at the results, and say, "See how far they failed to satisfy the mass-balance equation." And when they failed they'd say, "Well, then I've got too much numerical error."

They tended to do that when we decided to give them digital computer solutions; it seemed the natural thing to do. It turns out (and a number of people had noted this in some papers, and I may have even written one on it, too; I don't remember) that virtually all numerical methods you use automatically satisfy the mass-balance equations, except for round-off error. And virtually always these days you work in double precision. The round-off error is much, much smaller than the truncation error, so it's inconsequential in any ordinary-size calculation.

On the IBM 360 in double precision, the mass-balance equations were typically satisfied to fourteen digits, and they were delighted. They must be very accurate. They sent me one thing [problem], and I sent them back the solution. A couple of days later I noted that I had a big error, and had solved the wrong thing [problem]—I had messed up. I hadn't changed what the mass-balance equations were in the system; I changed something else. So these satisfied the mass-balance equations perfectly, and I called them up and said, "The solutions I sent you are wrong; I have to redo them." They said, "Oh, no, no: they're perfectly right. They're absolutely correct. Fourteen digits exactly in the mass-balance." I said, "No, no, you don't understand. I solved the wrong equations, but they happen to satisfy the mass-balance equations." It was very hard to convince them that these answers were wrong.

In fact, after that for a while, there was a big debate in that community. See, the mass-balance equations basically say that the number of this radical present plus the number of these should be equal to this constant. I could throw an equation away and solve directly for this or that thing, and have one less equation to solve. The big issue is, should I use the mass-balance equation to reduce the size of the system? The old hands said, "Oh, no, no, we have to use those to check the accuracy of the answers," which is nonsense. The small chemical kinetic system people said, "Yes, you should remove it; it will make it faster." In small [chemical kinetic] systems, by then, computers were so fast that it didn't matter. For big systems, which was where the problem really arises, it was questionable, because when you do that, you reduce the scarcity of the underlying matrix that you had to deal with, and it might make it slower even though it is a smaller matrix. It's denser; it could cause a lot more fill-in. So later on in the electrical engineering community, where this issue kind of also arises in some other functions, the attitude became don't try and reduce the system at all; just throw everything into the matrix and let the sparse matrix solver try and find the optimal treatment later.

HAIGH: The method that you created at Argonne—does it have a name?

GEAR: Well I didn't real create it, it was already in existence; it was the BDF method.

HAIGH: So it turned out to be a special application of the BDF method?

GEAR: My contribution was that I could recognize the extent to which it could be used. Curtiss and Hirschfelder recognized it to some extent, but I don't think it had been done very much. Secondly, I put it into a code that provided automatic step-size control and order control, which was a hunk of work.

HAIGH: So mathematicians wouldn't consider that doing that made it a new method; it was just a different application?

GEAR: I wouldn't consider it a new method. It tends to be called the "Gear method." I guess I shouldn't complain!

HAIGH: So other people would call it a method, but you wouldn't?

GEAR: I think it would be presumptuous of me to call it a new method.

HAIGH: All right. So, if it has any name, it's called the Gear method. You had this initial implementation of it that was basically just used as a proof of principle on a problem that was coming from Argonne. Now, this ODYSSEY system that you mentioned was another implementation of it?

GEAR: Yes, it was an attempt to make it more available.

HAIGH: You've mentioned that *Communications of the ACM* turned it down for publication. Despite that, did the package make its way into the outside world?

GEAR: I don't think so, because what I found very quickly was in those days, everybody had special requirements that required reprogramming. People in Sandia started to use it, and basically I had to rewrite some stuff for them. So I gave up on the automatic package.

HAIGH: What kinds of special requirements?

GEAR: Oh, this thing, for example, maybe to put in cards saying what the reaction rates were, or they needed to write another differential equation that computed the temperature and to modify the reaction rates for the temperature. Someone might have radiation. You never knew what--and I don't think, in those days, you could have conceived of writing a general program to handle all these special requirements. Computers were still limited as compared to what we have now. In a sense, that's why eventually DIFFSUB became what it was, which was much more minimal than Alan Hindmarsh provided. It just did one step at a time. I kept trying to look at how to do more, and I'd try to do a lot more in some of these things. Every time you ran into a problem, somebody wanted to, between each step, recalculate something. I finally decided the only way to handle this is to let them do a step at a time and tell them how to use it on the outside.

HAIGH: So you would have to write all your own code to initialize things and have a loop, and then it would just keep calling DIFFSUB. Basically DIFFSUB would be called from the inner loop?

GEAR: Yes. But it wouldn't be much. I mean, for a very simple code, it would be almost nothing—like just a *do* loop, or what we call now a *while* loop until you pass the final thing.

HAIGH: But then if they want to do some other things between steps--

GEAR: Right. For example, if they want to force it to print every so often, they can insist that they go to certain points only. Anytime I try to look at that and put it into a code, I get unbelievably complex sequences of instructions to the code. If you look at some of the software for simulation—I've got a bunch in my study now—I've got one there for doing some simulation of mechanical-type systems. The manual is this thick [indicates a 3-inch thick manual]!

HAIGH: So that's what you mean in the 1987 paper when you write, of the ODYSSEY package, "The package was of little value because it was too restrictive. It was impossible to append programs to compute the coefficients if the system would use tabular data, for example. For this reason, users at Kirkland AFB were unable to make effective use of the program."

GEAR: Right. So I had provided them a separate code for it.

HAIGH: Yes. "And I removed the integrator from the package and made it into a subroutine for their use."

GEAR: Oh, okay, so maybe that's where that package that Hindmarsh got came from. Maybe it was a preliminary…. I had forgotten about that. I don't read my own papers!

HAIGH: So that, I guess, was '67, you think? I'm just trying to get the date from this; '66 was the summer where you were at the preliminary version at Argonne. Do you think this ODYSSEY would have been written in 1967?

GEAR: I would guess so.

HAIGH: And that was never published. Do you have a copy of the paper that you wrote? That would be a very interesting thing to put online with this. The one that was rejected.

GEAR: I possibly do. It was in my NEC office, which I still have. I don't have room at home for everything. When I left Illinois, I tried to take one copy of everything I'd been involved in. I'll have to go look.

HAIGH: Anyway, except for the possibility that the code for the core single-step processing piece of code for that, might have been what Alan Hindmarsh got, that code pretty much vanished from the world.

GEAR: Right.

HAIGH: And then in 1969, you returned to it. It says here that this is in the context of your work at the suggestion of Forsythe on producing a book.

GEAR: Well, yes, I was doing the book, and he suggested that I should put this code in it. So my guess is that a code must have existed then, because I doubt if I had started from scratch. So maybe I had that code already, I simply don't remember.

HAIGH: Well, let's jump slightly out of software and talk about the book, so we'll do this, basically, chronologically. So meanwhile, then, this finally appears: *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice-Hall, Englewood Cliffs, 1971. And that's part of this series in computer science that Forsythe was editing.

GEAR: Right.

HAIGH: So how did that book come to be?

GEAR: The year before I went on sabbatical to Stanford, I was teaching a graduate course on numerical solution of ordinary differential equations at Illinois; at that time there were two courses at the graduate level. One was ODEs and one was PDEs. So I was teaching the first one. And the overhead projector was relatively new in those days, as I recall. I took the course by taking notes on three-by-five cards about what I wanted to say that day, and then writing my notes on new transparencies on the overhead projector rather than board, telling the students that I'd get them copies later. I always left adequate room so I could go back to my office and add in some of the verbal stuff that you don't have time to write down (and to correct the many errors on the transparencies). I copied them for the students, and I also thereby had a good Xerox draft of a book. Then my sabbatical plan was to turn that into a monograph on ordinary differential equations, which I did. Most of the time while I was at Stanford I spent rewriting that, and I spent time cleaning up drafts of it. Forsythe had suggested that a book would be much better if I included a code for ODEs in it, based on the stiff work. So I cleaned that up and put it in the book, and also then decided to publish it in the CACM Algorithms section, along with the paper.

HAIGH: So the idea for this book was that it would be useful as a graduate textbook?

GEAR: Yes, a graduate textbook, research monograph. Virtually all books I've written-- I think if you don't feel like you really like what you're doing and you want to do it, it doesn't work. I was really interested in this stuff, and when I teach I like to write the stuff down, so I wrote it down.

HAIGH: Then that was where the impetus came from to clean up and publish the code. How much work did it take beyond what you had already had?

GEAR: I really don't remember that, because I had forgotten before that conversation earlier that I probably already had something that I started with. I must have started with a piece rather than just the ideas from that earlier work. I remember spending quite a lot of time on the computer at SLAC, trying to adjust the various parameters that controlled the automated piece of it, like when it changed step size, how often, trying to improve the efficiency. An unbelievable amount of tuning goes into that. I ended up spending many hours poring over printouts, trying to see what was working and what wasn't, and trying to see the influences of various things.

HAIGH: Was that tuning you would do to a large extent experimentally—you couldn't just deduce from thinking about the mathematics of it, exactly how it had to be handled?

GEAR: My experience had been, every time I sat down and came up with a nice mathematical result, it bombed terribly because there's something I hadn't quite thought about. I spent the whole of last week here on one single problem, and I keep running into that issue. I have a great idea that's going to solve my whole problem, and then I find another little glitch in it. So in the end you have to look at the numbers. A large part of this is still art and not, unfortunately, science.

HAIGH: Would that kind of tuning for speed be platform-dependent?

GEAR: The name of the game was to try and make it platform-independent. Of course, today it's very much easier because if you can assume, for example, IEEE floating-point standard, that's the platform you have to worry about. In those days, there was not only the machine dependency, which we used to do a little bit by having machine-dependent parameters defined that you would look at. But also there was the issue of being compiler-independent, which was a big problem, still. All the FORTRAN compilers on all different machines had slight variations. We were still

incredibly worried about efficiency of the resulting code (in FORTRAN, in those days). Machines were still relatively slow.

If you look at DIFFSUB in today's light, it's a terrible piece of code. There's actually three small subroutines in it, but you won't see a subroutine statement for them, because subroutine use was on most machines, a high-overhead thing. So they were actually in the lines of computed GOTOs, which was just considered abominable code, but it was much faster on most machines. I also was very, very worried about storage allocation; I didn't want an enormous parameter list, but I needed a whole bunch of storage for various things. So I asked the user to pass in a couple of big arrays, and I would tell them the size to use for it. Then I would chop them up internally, and the choice of whether to chop them up in row orientation or column orientation is impacted somewhat by the efficiency of a lot of compiled code and how stuff does. That came back to haunt me. I had one user who called me up and said the code was running terribly. It was producing the right answers, but it was taking forever. So I questioned him a bit, and he said that he had noticed that I asked for 12n storage locations in this variable, but he noticed inside the program it was declared as 8n, so he changed it. The reason I used 12n was that I used the other 4n as rows across the other direction on the end of the space for something else. So he had those lying now on top of something else. It turned out that they were all working variables that were used in the automatic control, and the automatic control was [such that] as you drove the step size down to zero, none of this stuff played any part. But with the smaller step size, the automatic code gave the right answer, despite all this trash.

Then I visited Russia (that was in 1979), and they told me they couldn't get my code to run properly on their machine. I questioned them a little bit, and it turned out that their FORTRAN compiler violated the FORTRAN standard that said that matrices were stored by row (or whatever it was, I forget now). They stored them the other way, and my code was dependent on that for the storage allocation.

HAIGH: Given your background on how much you've been involved with this kind of architecture at a very low level for ILLIAC 2, and your other design projects, do you think that gave you a different mind-set from someone who hadn't had that experience and was just coming from mathematics and trying to write a program? That you enjoyed or felt more comfortable really going down in that kind of machine- and compiler-dependent space to try and squeeze out the performance.

GEAR: Well, I think it does. But most people working with computers in the early days had to deal with those issues. Things were slower. When FORTRAN first came out, we pooh-poohed it and said, "It won't get used. It can't possibly be efficient enough." For very small pieces of code, it can't. But you can't write 10,000 lines of code (that would be 10,000 lines of code in FORTRAN) in assembly language and ever hope to get it correct. So you can't do it in FORTRAN, either. So you have to go to higher-level languages for other forms of efficiency. Now you trade a lot of computer cycles and memory for reliability, for ease of change, and for all sorts of other issues that are important. The ability to write the code and check/debug it—if you can't do that, it doesn't matter how efficient it is.

[Tape 3, Side B]

HAIGH: So then you don't think there was anything in particular in your programming style or approach to mathematical software that has been influenced by your work on architecture and systems software?

GEAR: Yes, I guess I still think a little bit about efficiency in ways I think some people don't. You don't lose it. All the software people now are only interested in the beauty of the expression; they're almost like pure mathematicians—they don't seem to give a damn about the efficiency of the underlying code. But frequently, they're not writing the sorts of programs that run for enormous amounts of time. If you're writing a large PDE code on 10,000 processors, you do worry about what's going on at the lower level. So actually most of the people I've been around in the last ten years of my active career were not in scientific computing, and they tend to have a very different mind set than people concerned with large-scale numerical calculations.

HAIGH: Looking at the 1971 paper in *Communications*, "The Automatic Integration of Automatic Differential Equations", you say, "This paper describes the techniques used in the accompanying program; the program chooses the step size and order for the method. The method may be either a form of the Adams method or a method for stiff equations." Was the inclusion of the Adams method there to make it more general, so it could be used on a broader range of problems?

GEAR: Yes, I was interested in getting a piece of code out that could be used by a lot of people, not just for the stiff equations. I had done this stuff for variable order, and the issues were very similar in both. There weren't such codes at the time, or there weren't any variable-order codes widely available; Krogh certainly had some stuff for non-stiff equations. I guess he was working on stiff equations, too, by then. I don't know how well his was distributed; I never saw the code. So that was the outcome of what Forsythe urged me to do to put in the book; it was exactly the same code in the book. I owe Forsythe an enormous debt of gratitude, because in terms of my career, that was the most valuable publication I made. It had the biggest impact I'm sure. In fact, I still tell students, I say, "If you really want to have an impact, make sure you write a piece of code that people can use." In theoretical computer science—algorithms—you can write some theoretical methods, and that's all that is needed; people can implement them. But in scientific software, it's almost impossible to just implement something from just a written description. There's so much there in the way you tune little things, the way you decide to handle stuff. There's not a whole lot of theory behind some of that. You just have to get a sense of how to do it.

HAIGH: How would you describe Forsythe?

GEAR: I didn't know him well. He was a very pleasant man. He died rather early of cancer, which was sad. He built a fantastic department there; he decided to go to Stanford to build that department, and he did it. Interestingly, he probably was not an outstanding researcher himself. He had some good work, but he was outstanding in way he had insight into what was important and getting things organized.

HAIGH: You mentioned that the code had a big impact through its publication in *Communications of the ACM*. How did that play out in practice? A few weeks later, did you start getting telephone calls from people? How did you become aware that it was actually being used?

GEAR: Well, various ways. In those days, my choices for giving people a copy were either a box of cards about this big or a magnetic tape about that big that cost quite a bit of money. I think I would have had to ask people to send me some money if they wanted it. I guess in those days we had plenty of contract money, so I probably didn't have to ask. I sent out quite a lot of copies, mostly, I imagine, on tape, and I suppose most of those people copied them and sent them on.

HAIGH: So people would read the article and then contact you to get you to make them a copy?

GEAR: The code was printed in CACM in the Algorithms section. I'm not certain that the ACM provided a mechanism for getting--

HAIGH: They certainly did after TOMS was created.

GEAR: Yes, I think that's what they did before that. A number of people had somebody keypunch it in either from my book or the paper, and of course, they invariably got errors, so I usually finished helping them out debug it. I wish I would have been charging them a consulting fee—I would have been a rich man. I don't know if they still do, but *ISI Weekly* used to print a little book of the table of contents of all the various scientific magazines, and they also each week would have their "citation classic, which was the paper that had gotten the most number of citations in the most recent period.

HAIGH: So were the people citing it, say, physicists or engineers, who would use this routine and then were citing it in the paper publishing the results?

GEAR: I don't know; I have no idea. You know what it is: if you have relatively the first paper in something, you get cited a lot anyway, whether anyone has ever opened the thing or has any idea of what it's about.

HAIGH: That would reinforce itself, because then…

GEAR: Right, because the next people just cite the same stuff over and over again [chuckles].

HAIGH: So you were not aware of whether these citations were coming from people who were using the software, or from numerical analysis researchers who were building on it?

GEAR: Well, I think it was probably more people who were using it, because I find, even more now, the computer science community doesn't even know what a stiff equation is now. Right now I have a courtesy appointment in chemical engineering at Princeton so I can have access to the library; I'm doing collaborative research there. I talk to people like chemical engineers; they know my name. But people in the computer science community say, "Uh, who?"

HAIGH: That's interesting. You were giving out copies of it; you were getting telephone calls from people who typed it in and were trying to make it work. Were there any other kinds of interactions that you had with users, or ways in which you became aware of how it was spreading?

GEAR: Well, of course, Alan Hindmarsh did me a wonderful service by writing this program and calling it *DGEAR*, or something like that. It had my name on it in some way. I wouldn't have dreamed of putting out a program calling it my name (unlike some of my colleagues, calling it their name). I should be very thankful to him for providing me with a lot of free publicity. IMSL also put it in their library and gave it a name and included Gear; I assume they probably started with Hindmarsh's code. I don't know, I didn't have the IMSL library, they never sent me anything of the source. That was kind of a pain, because I quite frequently get phone calls from IMSL users saying they're having a problem with this code. I had to say, "I haven't even seen the program—I don't know what the code sequence is, so I can't tell you anything."

HAIGH: Do you know if it ever made its way into the NAG library?

GEAR: I should know that. I don't know. I knew Brian Ford pretty well in the days when he was running NAG. We were both on WG 2.5 together.

HAIGH: So as the code spread out into the world and got adopted and repackaged in different ways, did you try and keep track of what was happening with it and see how people had adopted it? Or, as far as you were concerned, was it something you had finished with it and you had move on to the next thing?

GEAR: Another of my failings (this is confession time, right?) is that I sometimes don't stick with stuff long enough. Once I feel it's sort of done and from now on it's clean-up time, I get bored, I want to move on to the next thing. By then I was interested in differential algebraic equations, which I think also came out in 1971. Interestingly, in some ways I made a much better contribution there and had a far greater impact, but I'm probably not known from it at all.

HAIGH: So what year did you really start working on those?

GEAR: Even back in the first summer in Argonne, 1965, I was thinking about equations that changed their order; I was working on multiple order equations. Supposedly you got this equation that looks like some coefficient times y'' + another coefficient $\times y'$ + another coefficient $\times y$. Well, that's a second-order equation. But suppose that first coefficient goes through zero and it becomes a first-order equation. So if the order of the equation is varying: what can I do? And I didn't come up with a good solution for it. I know much more about it now.

So I was turning that over in the back of my mind, and then at Illinois I started another project that created a lot of student theses. That was considered to be a major part of my job, anyway, to train students. What I was interested in doing, because of my interest in systems and my interest in differential equations, was to create a very general-purpose simulation system. There's hardly any publications on it except student theses.

HAIGH: So this would be something you would have been working on in the early- to mid-'70s?

GEAR: Yes, in fact, it continued for quite a while. I don't see many direct theses on it, now that I look at it.

HAIGH: Would you like to point to a couple of those, then? We have your list of students you supervised. Wilkins? Runge?

GEAR: Yes, it must have been so. Well, Speelpenning was related to it as well. There must have been some others there. Chung's work was related to it, too. In the 1960s, there had been a lot of simulation packages, in electrical engineering in particular, and in mechanical networks.

Interestingly, those are almost identical problems, except that the mechanical networks' dimensionality is just three times higher. Electricity basically has current and voltage. In physical systems, you have position and velocity and they're three-dimensional. But if you look at the networks, a building structure with columns, the columns are elements like resistors and so on, and the way they all join together are like nodes in the network. In these equations the whole things are very similar. There was a whole lot of work going on in the '60s about how to take general descriptions of networks—electrical and structural—and reduce them to a set of ordinary differential equations to solve. And that's where some of the stiff equations are coming from.

I got very interested in this. I'm interested in differential equations, and I'd like to put together a system where somebody sketches in the network on a display terminal (I was interested in graphics, too), and we describe the connections, but rather than say this is an electrical network and this is a resistor with ten-ohm resistors on it, they are able to define arbitrary elements—put a box there and put a little design on it, say this is element type Y3—and here are the equations

that describe how this element behaves. You could represent a water pipe, anything you like. And you could have multiple types of terminals: you could have water terminals, electrical terminals, structural terminals, control terminals, all things. Then you connect this whole thing together, and then you get a bunch of equations and you try to simulate them. In fact, there are programs like that right now, and I'll mention the story about that in a minute.

HAIGH: So some kind of generalized network simulation package?

GEAR: Right. So we started work on this, and this was the main thing. It brought together my various interests in programming and graphics. I had the group working mainly on different aspects of this problem, so we were doing compiler work. One of the things we didn't have was good higher-level language systems, and I wanted this to be portable. On any given computer, you could have certain languages that it wouldn't have on another computer. We wanted a higher-level language; what could it be? So we decided to write our own higher-level language, called PLW (where the *W* stood for *Won*). It had a compiler that compiled into FORTRAN, so you could then run it on any machine. We viewed FORTRAN as a sort of intermediate machine language. So we had all these subsidiary projects that were various theses, and some of them I don't seem to have down there.

We started work on this, and one of the early things that I found as I looked at all this literature on electrical networks and so on, there were very clever ways for taking the matrix representation of circuits, manipulating them until you got an explicit system of differential equations—this was $y' = f(y)$, basically. You could do it because of certain fundamental facts that were true about network connection matrices, they have certain properties (we don't have to go into them). And so there was a lot of work in the '60s on that, some at Illinois, Steve Fenves was at Illinois then -- civil engineering department. People like Steve Director —a whole bunch of people. I knew some of them. And there was a lot of interesting work by people at MIT, Carnegie-Mellon, etc. I was going to try and use that to get the differential equations. When there was stiffness—*that* I knew how to solve.

I quickly ran into the problem that as soon as I introduced this generality, there was no way in general of reducing this to a system of differential equations. What you had was this mixture of non-linear equations and differential equations in very strange forms. My first problem was, what do I call them? *Differential and non-linear equations* doesn't seem to make much sense; most differential equations are non-linear anyway. With great reluctance I eventually settled on the name *differential algebraic equations* but the math community assumed it was that versus, say, differential transcendental equations, which it is not. I wrote a paper on that, about how to solve them, in '71. That was the first time that term was used, I believe. IEEE-something; I think it was '71.

HAIGH: That's it—"The Simultaneous Numerical Solution of Differential Algebraic Equations." Then it says, "IEEE Transactions on Circuit Theory. TC18, No. 1, pgs. 89–95, 1971."

GEAR: Okay, so that was the first time the term ever appeared in print, and now it's almost its own division of numerical analysis. Differential algebraic equations got very interesting. And we used those methods in that system I was building.

HAIGH: Did that project ever produce any kind of distributable software?

GEAR: No. So from that point of view, it was a failure. And in spite of my advice to students--well, I think, probably that individual students distributed some of things they did.

HAIGH: Was the problem that it was just too ambitious? Were there missing pieces mathematically in terms of the methods needed?

GEAR: I think it was probably too ambitious as a total system. It was too early—we were using a PDP-8 with storage tube technology monitors added to it for the graphics. You couldn't get away from hardware-specific stuff in those days.

As I alluded, there is a story about that that came much later that didn't actually surface until I was here. Sometime in the early- to mid- '90s, I got a call from a company which was being sued by another company for patent infringement. It turned out to be over issues that were very much like these. This other company had a system that looked remarkably like what I had proposed and was available in many reports. In fact it turned out that one of the principals in that other company had been at Illinois in another department at that time. So the first company decided to fight this patent infringement case, and I was called in to testify for it. I made a long deposition. But they were successful.

HAIGH: So there were some technical reports from this group that describe the system you were attempting to build?

GEAR: There were a number of technical reports, yes. I probably have those in my office, if you want to check anything.

HAIGH: I know in your publications list you have a--

GEAR: I have a short book of reports; I don't publish nearly all the reports.

HAIGH: You just said that you thought that this was, perhaps, your most important area of work…

GEAR: Differential algebraic equations, I think I said. Not that vast system… But trying to do that system led to the understanding that you had to deal with differential equations directly, and I think that was pretty valuable!

All the money spent on that system there, a) produced several theses; and b) I think it led to an important understanding of what we needed to do to handle differential algebraic equations directly.

HAIGH: So in terms of that 1971 publication, then there's a lot of things on ordinary differential equations. And then jumping slightly out of chronology, in 1984 you have a publication with a Linda Petzold. [ODE Methods for the Solution of Differential/Algebraic Systems, (with L. R. Petzold), SINUM, **21**, #4, pp716-728, 1984].

GEAR: Well, what happened with differential equations was the following. We pursued, for a while, that system. It probably waned out when most of the students doing those things as their theses graduated. One thing I was very reluctant to do, which some of my faculty colleagues did, was just have people go on doing more of the same: one student finishes; another student starts doing the next, the $n+1^{st}$ piece. I usually start them off on something different. So the time span of many of my projects really had to be on the order of three to four years, and the student moved on.

We had the system sort of running initially, and one of the early examples we used was the pendulum problem. Well, you can describe a pendulum in various ways; if you do it carefully, you describe it by the angle and it's a very simple differential equation—second order, no trouble at all. But if the idea was, you should be able to describe things in arbitrary ways—what's a natural way to tell somebody who doesn't think about mathematics to describe a pendulum? Well, I have this mass. I'll call it a point mass here. It's got *x* and *y* coordinates, and it's satisfying the laws of gravity. My connection system has to do with forces and so on. And then I have this bar between them, and the bar satisfies, namely it's absolutely rigid, so that the distance between the ends is this, and blah-blah-blah, and it's fixed in this position and can pivot freely. And you write down these equations, so you basically get Newton's equations for the acceleration of the mass on the end, and then you get this constraint: $x^2 + y^2 = l^2$. That is what we would call now an *index-3 differential algebraic equation.*

My system just died on it. I didn't understand why. Well, I started to understand why. I managed to produce various sorts of systems, so I finally understood what the issue was and why I thought I couldn't handle it at the time. Since the system was dying, I somewhat forgot about it, although I went out one summer—maybe 1975—to Stanford. Two summers I spent some time at Stanford; I think it was '75. Gene Golub used to organize a weekly get-together there where people could talk informally about things they were thinking about, and that was on my mind at the time. So at one of these weekly lunches I talked about this problem and what the issues were; Cleve Moler was there at the time. He shortly thereafter had a student work on it, whose name begins with "S", and I would have to look it up. He wrote a thesis on the issue. I'm not certain of the exact sequence of events here: I think it was after that that Linda Petzold saw it. She didn't leave me until '78, I think, [when] she graduated. She was at Sandia by then.

HAIGH: All right. Would that be Norman Schryer? That's an early one…James Sanderson? John Starner?

GEAR: Starner, I think. Was he at New Mexico? [John William Starner Jr., PhD, University of New Mexico, 1976].

HAIGH: Yes, 1976 he graduated.

GEAR: Yes, okay, that was it. Linda was at Sandia and I ran into this thing, and then she wrote a reports entitled, "DAEs are not ODEs," which was very interesting. That piqued my interest. Then I started working with her, and we wrote a paper. We wrote several papers. I had her as an adjunct professor at Illinois for a while, and we jointly supervised a student, Ben Leimkuhler on some stuff. I got back into differential algebraic equations, which I worked in until I moved to NEC, when I ran out of time [for research]. My last paper was with Steve Campbell, which appeared sometime in the early nineties. [The Index of General Nonlinear DAEs (with S. L. Campbell), Num. Math., **72**, pp173-196, 1995].

HAIGH: So was Linda Petzold your student?

GEAR: She was my student, but not in DAEs. We did oscillatory equations.

HAIGH: She worked on one thing with you, and then she graduated, and then independently came to work on DAEs?

GEAR: Right. And that whole field has taken off enormously. In fact, I've been invited to a meeting on it next spring, which I said I'll go to, but I think I may not have time enough to try and catch up on it since I've been so much away from it.

HAIGH: Do you think there are good ideas or important work that you did on this that you just never published and people really didn't become aware of?

GEAR: Well, there's probably some stuff there, but there were lots of department reports, so probably most of it got picked up. There's something I found out much, much later, because somebody asked me about it. There's something called the *Gear Anode*, and I said, "What's that?" It turned out, I was interested in graphics, and one of the things we needed was very low-cost graphical input devices. They were very expensive at the time. There was also a lot of interest in being able to put the graphical input pad right over the monitor—people were at the monitor sitting down and then you wanted to be able to put a pen on it. (People later on decided that wasn't very important.) One of the technologies that was of some interest was -- you put a resistive coating on a sheet of glass, so then you can touch the conducting point to it. You add some electrodes on the side, and an electrode there, and by sensing the voltages you could sense where the pen was. Well, this was fine—unbelievably non-linear.

Today, that's not a problem: if I wanted to use a technology like that, they would say, "No problem! I'll just put a little microprocessor there." They cost 50¢ in bulk. Do the non-linear conversions to figure out where it is, and even adjust for variations in the manufacturer of tablet by doing a few samples. In those days, that was out of the question. It was still very expensive—mostly discrete components.

So I asked myself, what can you do to get linearity in this? And I said, okay, let me have a sheet of uniform resistance. But now let me put some line resistances around the side and see what I can do. It turns out the following, very interesting result was sort of trivial. This is the advantage of having a Cambridge education. I learned a lot on how to solve Laplace's equation in applied mathematics. Here's a basic thing: if you take a sheet of uniform resistance of anything that imparts a uniform current, say coming from north to south, and you cut a circular hole in it and you put a line resistance around that hole whose resistance is the ratio of the surface resistance divided by the radius (or the other way around—I forget exactly what the scaling is; you have to get it the right way), then, away from that hole you cannot see it [the hole], the current continues to flow uniformly elsewhere. Basically what happens is if it [the current] hits the side at a quadratic boundary, it goes around [the boundary] and then that current comes back out straight. Which means, to do it the other way, if I take a square and put circular edges the other way, so they're concave line resistances, and say make that corner plus, that one plus [indicating two adjacent corners], and the other two sides minus, I have a uniform current through there. So I get a uniform voltage gradient.

Okay, so we tried it; the trouble was, the surface resistances were too liable to get scratched; they didn't work, so we gave up. But it appeared in a very minor publication at one of those AEC annual meetings/discussion things. I don't think it's in my publication list. It's probably not there because it was just some report to some AEC meeting.

Some physicists picked it up and realized the same thing would apply for a detector that's being hit by radiation. So they make the anode this shape, and they get linearity. And it was called the Gear Anode. I found out about this many, many years later. In fact, I found out about it when, I believe, I mentioned that lawsuit where I was asked to testify and they had done a search on me on all the stuff I had done, probably in preparation. They asked me about this and showed me a paper that referenced it. And that's also, by the way, when I found out that I had a patent from my IBM days I didn't know about.

HAIGH: Ah! I meant to ask you about that. I noticed on your résumé you have a patent, and it's got this really impressive title: "read-only memory." [US Patent 3,246,315, ``Read Only Memory," Apr 1966]. So did you really invent ROM?

GEAR: Oh, no, no. This was from my summer as a grad student at the IBM Research Lab in Ossining. A wonderful place: it sat right above Sing-Sing prison, so at lunchtime we'd go outside and peer down over the bars into the inmates of Sing-Sing prison who were, presumably, peering back at us, envying us. I was looking at various things. We were still very much in the discrete-component phase. I don't know if the word *read-only memory* was in use very much. I remember hearing that word for the first time when I was at Illinois, and suggesting, "What about write-only memory?" which would be really valuable for the results of some of the people I know. Anyway, so we were interested in read-only memory; they were very important in things like micro-programmed machines, though that was not the context. They were important in providing tables or stuff for various applications.

In a sense, this is probably really a data compression scheme. I was considering a very simple diode-switched matrix scheme to put in read-only memory. Basically the idea was, if you already had something stored in the memory, you've got this array of resistors and diodes—kind of a matrix. On the addresses you're interested in, it provides the right data. And now I want to store something else in it, in another address; the issue is what's the minimum cost of adding stuff to this, so I don't change what's already in it and I can add this new word? Another thing is, if I'm prepared to change the resistors that are already in it, or whatever, but so it doesn't change the data, because I haven't actually made it yet, how do I finish up having something that provides the output I want for the input I have at a minimum cost? It was a suggested solution—a rather empirical solution—to that problem. And I remember how frustrating it was to describe it to the lawyer at IBM. Basically, what you said, let's suppose you've got the stuff you want in it now, and then you feed in the address of the thing you want to store, and you look at what comes out, and then this is what you do. And every time I said that he said, "How did the stuff that was already there get in there?" And I would say, "I'm about to tell you that," because, you know, I was still working recursively. They could not ever get that concept! I said, "Well, it's simple: there's nothing in there, so the first time you try it, there's nothing."

HAIGH: You just keep adding things until it's empty – that's how you should have explained it.

GEAR: So I didn't think the patent was going anywhere, because the lawyer didn't seem to understand it. I guess there was enough push to get patents, but I wasn't sure that they would know the value. They did, but I didn't know about it. Of course, at the time, I signed the usual release for the nominal dollar payment, which I never got.

HAIGH: You didn't get a plaque or anything? [Laughter]

So moving on to something else you alluded to. We talked already about your work with systems and microcoding, but you have alluded to an ongoing interest in graphics, and I know earlier in your career you were involved with the graphics community. So was that interest coming out of the oscilloscope on the original ILLIAC?

GEAR: No, we bought a PDP-7. It was a compound system. It had a 338 display on it. It was a DEC. It had a whole name, but I can't remember what it was, which we'd attached to ILLIAC II. There were a couple of theses going on on that.

Probably my most successful, from an academic point of view, has been Linda Petzold; my most successful student from other points of view has been Fontaine Richardson. He was also the first

or second Ph.D. in CS at Illinois. We were exploring novel ways of programming graphically using that system. So I got interested in graphics. I was interested in a number of things in graphics in those days. Graphics was still extremely primitive. There were only vector displays; there weren't many—they were expensive. They had been used a little bit in air traffic control. You mentioned things like SAGE. There were a few around, and people were starting to look at various ways of using them, and so I got involved in it. I still think it's a fascinating area, and from time to time, I've done a little bit in it again.

HAIGH: So here's a list of students supervised.

GEAR: Okay, which order. Fontaine, Richardson, "Graphical specifications of computation," 1967. Well, I guess we did that earlier than I had remembered. I guess that was about at the time I was involved with the ACM Special Interest Group on Graphics. I think that was when it first started. Would you believe it contained about 15 members?

HAIGH: Now it's this enormous…

GEAR: Yes, bigger than ACM, practically.

HAIGH: So you were editing the SIGGRAPH newsletter.

GEAR: Yes. At that time, it was printed in the Department of Computer Science at Illinois at our cost; it was just printed on paper like this [indicating standard 8 ½ by 11" bond paper]. Unfortunately, there don't seem to be any copies of that stuff; I couldn't find one when I left. I think I put out two issues— it didn't have much in it.

HAIGH: So that wouldn't be in the ACM digital library? I know they have made an effort to include things like that.

GEAR: We printed it in the department; I mailed it out from there. I doubt how many other people were involved at that time who would have gotten a copy, and there weren't many. So it's probably lost. I couldn't find one.

HAIGH: Do you have any general impressions of the computer graphics community in those days?

GEAR: It was a diverse group. One or two people there were really salesmen trying to sell us stuff; some people were interested in trying to get standards; other ones we saw just found it a fascinating topic. There were no fancy displays; we were interested in, really, the nuts-and-bolts. I can't remember when that first textbook came out, the Newman-Sproull [Bob Sproull & William F. Newmann, *Principles of Interactive Computer Graphics*, 1973]. It was probably the first textbook in the area. I taught a course in it, using that book. I don't remember when that would be, probably around 1970, '71. There are still a lot of interesting scientific computational problems in graphics and image understanding, and a tremendous amount of work done in the scientific community now. People like Tony Chan at UCLA working in it some. Fascinating stuff.

HAIGH: Your involvement in it was part of this bigger project, the network simulation system. Was that your main ongoing involvement with graphics?

GEAR: Yes. I've always been interested in some of these "gee, whiz!" things, so it was probably that. It was a fascinating thing, and I don't think that people yet knew what it was going to get used for.

HAIGH: Anything else about graphics?

GEAR: While I was working at NEC, I wrote one paper on graphics for some work I did at NEC.

HAIGH: Okay, I'll ask you about that later when we get to NEC. So, we've talked about your work through the seventies on differential equations, also differential algebraic equations and the origins of this generalized network simulation and computer graphics system. Unless there are any other big things to do with your research in the sixties and seventies, I suggest that we return and talk more about the department and your career within it.

[Tape 4, Side A]

HAIGH: So I think we've certainly discussed your return to Illinois as a tenure-track faculty person. You had mentioned that you got tenure about 1966. Was that a traumatic process?

GEAR: It was not a traumatic process in those days. I don't remember ever thinking about the issue of tenure, unlike people now. In fact, my recollection was, I walked in to the head of the department one spring and said, "I think I should get promoted." He said, "Hmmm…" and then he said, "Okay."

HAIGH: Now how was the computing program developing there? I think I found from the history of the department that the Department of Computer Science apparently was created in 1964, around the former digital computer laboratory that you had been involved with.

GEAR: Right.

HAIGH: Then in 1966, a computer science graduate degree program was established in the graduate college. How much practical impact did that have on life there? Did it make a big difference? Was there an expansion? Were there more resources? Or was it mostly an administrative thing?

GEAR: I don't recall of it having much of an impact. I think we had pretty good relations with the other departments in those days. I don't ever remember any problems at that time with grad students. I certainly hadn't had any with the departments. Up to that point, of course, we didn't have to run our own qualifying exams, and suddenly we had to start doing all that, so it increased the workload considerably, and, probably revealed some of the dissentions in the department -- what should be in the exams -- which I'm sure is a perennial department problem there. In fact in later years, I remember very vividly, by the time I was department head in 1985 plus, and we were growing rapidly. While I was department head we grew from just over 30 to almost 45 faculty. So we had this rush of new people, and the new people came in and wanted to change the qualifying exam and the curriculum because they didn't like what they saw. And I realized that what they wanted to change it to was what, when I had been a young Turk a long time ago, we had agreed to change away from because we didn't like it. I came to the conclusion that it really didn't matter what it was; what was important was that every five years you had to change it because people got so lackadaisical about falling into some form, it had to be changed, and that changing was the only important issue in the graduate program.

HAIGH: Do you mean what was going in the course syllabi, or what the students themselves were preparing?

GEAR: Well, it was getting people to write questions for the qualifying exam, et cetera, et cetera. It was like drawing teeth to get them to turn in questions or grade them. But if they just changed it to something they wanted, they were all eager to do it for a year or two.

HAIGH: The department was growing at the time. Do you remember any periods of particularly rapid expansion or particularly important hires that took place during the ‘60s or ‘70s?

GEAR: Well, in the middle ‘60s we hired Dave Kuck, probably about ’65, and Dave Liu a year or two later. His real name was C. L. Liu—Chang L. Liu. He’s retired. Kuck’s retired. We’re all retired here; half of them are dead. Other names… We hired a number of people who have not done such great things and we hired some people who have done some great things. What would be their names…. I forget when [Herbert] Edelsbrunner was hired, maybe quite a bit later. I think he’s gone now, but he was a big name.

HAIGH: In the department history page, it says in 1991 he becomes the first computer scientist to win the NSF Alan T. Waterman Award for his fundamental contributions to computational geometry.

GEAR: Yes, he was a great acquisition. There were a number of people who came in; some were big names, some were harder to deal with than others. I noticed on your list you had the question about Wolfram, which comes up a bit later.

HAIGH: Well, that will some time to think about a diplomatic response then. As the department grew, were there any areas in which it concentrated and became particularly known for? Or would you say that hires were spread fairly evenly over what would now be considered the different main areas of computer science?

GEAR: Well, it had this big background in computer architecture from the early days when we built computers. And, indeed, other computers, the ILLIAC III, which never made it (it was finally destroyed in the fire, which might have been fortunate) and the ILLIAC IV, which caused a lot of controversy and was taken away. Then there was [David J.] Kuck project, NCSD…no, that’s not quite the right name. NCRA? No. I think you had the name in this list. Well, Kuck was involved with Slotnick on the ILLIAC IV. When that folded, Kuck formed his own group and got a lot of support for doing a higher-speed computer. He also formed something, -- KAI, Kuck and Associates, which is now owned by Intel. He’s a manager of that for Intel. So he had this separate research group, separate from the department. There was a little bit of a falling out of the department because Kuck would have liked to run the department ancillary to his research group.

HAIGH: There is something called CSRD.

GEAR: CSRD—that’s right. Yes.

HAIGH: All right. That’s Center for Computing Research and Development.

GEAR: Yes. They were working on a computer that ran into a number of problems, including financial, in the end.

HAIGH: So they were trying to build their own machine, were they?

GEAR: Well, they were going to have it built outside, but they were doing the design. It reached the point in time when you couldn’t really build hardware inside; it was too much equipment to do the building.

HAIGH: So you mentioned that there was an ILLIAC III, which burned down.

GEAR: Yes, that was a pattern-recognition computer, whose major work things would have been processing data for things like bubble chambers and high-energy physics experiments. It was

AEC money. It was suffering from, mainly, I think the director was having new, good ideas and was always saying, “Oh, let’s stop doing that and let’s go and do this.” So nothing ever got done. They got a lot of equipment and never quite finished, and then someone left a power supply on that was sitting on a wooden workbench one weekend and it overheated, set fire to the bench, and caused an enormous amount of damage. They essentially terminated the project.

HAIGH: I think the ILLIAC IV project was widely known as unsuccessful. Was that something that was the big thing that was going on in the department for while?

GEAR: It was the big thing that was going on in the department for a while. Slotnick had been at, I think it was General Electric before that, where he had this computer paper design called *Solomon* for a large SIMD, Single-Instruction-Multiple–Data [stream] machine. He was hired to the department in about 1968, ’67.

HAIGH: Actually it says 1965 on this history page.

GEAR: Oh, really. Okay, well, maybe it was ’65. Anyway, he got some government money to pursue this idea. He was getting, I think, DARPA money for it. It got more and more expensive. It was going to be manufactured by Burroughs. I guess it finally was. Originally it was going to be 256 processors, I believe, so four groups of 64; I think in the end it finished up with 64 processors because of the cost. One of the things that happened, and it was 1969 when it came to a head, it was getting more and more expensive and he didn’t have enough money, and he basically started, as I understand it, offered to provide time to the military or the CIA or somebody on the computer for defense work in order to try and get more budget. As you can imagine in ’69, which was kind of a heady time with all the students and the military, this did not sit well.

HAIGH: That’s true, people were bombing campus computer centers and so on.

GEAR: The reason I can fix that date very clearly—it was in May or June of ’69 that it came to a head—and the department was taking a big stand against secret work. I still feel very strongly: you can’t have students do theses on stuff that they cannot publish freely. To me, that is a non-negotiable item. So basically, the college was not prepared to have it in [the department] anymore. The reason I can place the date very clearly was at the time I was in India giving lectures at a summer institute, the Indian Institute of Technology in Delhi for seven weeks on NSF sponsorship. Communication was pretty primitive in those days. I came back to the hotel where I was staying—it was a pretty minimal hotel—from having lectured one day. There was a phone message that just said, “Call from Urbana.” No other information whatsoever. So I panicked—oh, there’s something wrong with my family! I tried to call but radio links; they didn’t operate at that time of the day. It took me over 24 hours to get through to find out that it was one of the faculty in the department [who] wanted to get my vote on an issue. So that’s how I can place the timing very carefully.

Slotnick formed his own group, called the Institute for Advanced Computation, and got funding for a building down the road. It had copper lining so they could stop the radiation from the computer from being sensed outside, and so on. He used to refer to the department of computer science out there as the “department of retarded computation.” Slotnick had a way of annoying people deliberately. And that was what eventually became-- because Kuck was involved in that process, and then when that kind of folded up, Kuck formed CSRD and went into other stuff.

So the ILLIAC IV wasn’t very successful. When it finally installed at NASA Ames Research Laboratory, it was because it wasn’t safe to put it on campus. They didn’t want it bombed.

HAIGH: So then you would say that the kind of thread that goes through there is these sort of large-scale, kind of supercomputing architecture work as being the distinctive thing about the department. Now, how about this general process that I think was reproduced everywhere, where numerical analysis and scientific computation went from being the center of computing groups at universities to eventually being quite marginal in most. How did that play out?

GEAR: Well, it was possibly happening less at Illinois than in most universities; a few of them maintained strong areas there, Purdue may, but it really depended on the people there. If there were some senior people, typically, it maintained its presence. So there was Rice at Purdue, Golub at Stanford—you can go around to places where it remained a presence, and the places where it folded were the ones that didn't have anybody and didn't want it there. I don't think any computer science department really viewed it as, "Oh, we've really got to have somebody there!" Well, we could have somebody there. We don't have to kick them out of the door…

HAIGH: If they're already in place….

GEAR: So a lot of it now gets done in other departments, which is interesting. It's like computing was in the very early days, the middle 1950s, early '60s. Computer departments were in all sorts of strange venues—there were some in chemistry departments, some in engineering departments, some in math departments, some in business departments. It was simply that they had to have a faculty member there who was really interested in getting into computing and got something going, and it naturally gravitated there. And then one or two universities, where they had three different computer departments because they had three different faculty in three different places.

HAIGH: So through your time at Illinois, did numerical analysis remain a compulsory part of the graduate and undergraduate curricula?

GEAR: I think towards the end it became optional. As did computer hardware, finally.

HAIGH: You had mentioned that the main major impact of having a graduate degree program was that people had to come up with questions for the comprehensive exams. Presumably, also, that you started in having Ph.D. students that you need to supervise and can put to work on different projects. How about the undergraduate program? Did that have a big influence on the character of the place?

GEAR: The influence there came much later. I don't exactly remember when the peaks were, but sometime in the late-1970s, early-'80s, undergraduate enrollment in computer science started going through the ceiling. There was a point in which I believe-- you know, when we still had something like 30 faculty, I think we had, I forget the numbers, between the two degree programs—one in LAS [Letters and Sciences] and one in engineering (I think by now that the "Math CS" had pretty much fallen into just a CS in LAS), I think we may have been pushing 2,500 undergraduates with 30 faculty.

HAIGH: Can you talk a little bit about how you came to have two different computer science programs?

GEAR: At Illinois (and maybe many universities), there are some core requirements in any given college. Engineering has core requirements; physics, chemistry, and a variety of fundamental subjects like that. And there were different sorts of students. There were students who were closer to other LAS disciplines who wanted an LAS-type degree and didn't want all the engineering stuff. In fact, our LAS undergraduate degree had a requirement for something like

four courses in some other discipline. It could have been art, it could have been history, it could have been in English—something that made some sense that ultimately might be an application area. Our engineering degree had to satisfy the core requirements for engineering; it was designed for people who were going to work in the engineering arena. So the requirements were quite different. Plus, the engineering college at Illinois was getting to almost impossible to get into because there was such a demand to get there. We had a lot of people apply to LAS because it was a little easier to get into, with the idea of switching.

HAIGH: So you had one set of courses and one set of faculty and two degree programs?

GEAR: No, no, there were common courses. The requirements for the degree programs were different, but they were all expected to take the common courses. We were offering, by then, about three introductory programming classes: one was for engineering majors—FORTRAN; one was for business majors, which I think initially was also in FORTRAN—it was a very quantitative business school; one was for CS majors, which had a lot more theory in it and used a different language. I haven't checked, but this whole thing we did in introductory computers in those days wasn't so much programming. It was no longer needed because you assumed that in computer science the kids coming in already knew how to program. So the courses were pretty much the same, although if they were an engineer, they were expected to take a lot of engineering classes, and they might have been required to take a numerical analysis course. But in LAS there was less emphasis on those courses. And there was quite a selection by the time you got to the senior level.

HAIGH: Now, your own appointment—did that switch out of the mathematics department and into the new department?

GEAR: I think in fact I was said to be in the mathematics department. I don't know what it meant financially since I was not involved in university issues in those days. My suspicion is that the budget allotment was actually in computer science in the digital computer lab.

HAIGH: All along?

GEAR: All along, but they were required to be approved by the math department as a faculty appointment. Because that's the way we used to operate with some of these other labs—they would want to appoint somebody on their budget, but they wanted to give them the title of professor, so they had to have an appointment. And it meant that we were stuck with them if that lab ever disappeared. That was always a bone of contention and a problem in many universities. These laboratories are set up. They don't have tenured faculty slots; they want be able to appoint people and, frequently, they want to be able to appoint people because, "Boy, this guy's a real *gee-whiz* on doing this design." But they would be absolutely hopeless as a professor to teach classes do anything else and get money. You don't want to give him a tenured position as professor. So it often caused arguments over problems like that.

The other issue you mentioned earlier on about hiring your own. This was a big problem with these labs; they would go to a grad student who had just finished his Ph.D. and had been doing all this valuable stuff for the contracts they have with this lab, and they want to continue him on. They want to make him a professor and keep him -- and no way!

HAIGH: Were there people who left in the mathematics department who had an interest in applied mathematics and computational mathematics and so on who would work with the people in computer department? Or was there basically nobody with those interests remaining in the math department?

GEAR: There were one or two old people, like the guy I took methods of applied mathematics as a grad student; I think he had retired some time later. They had a young guy who was more interested, maybe, in education than computation—Francis I think was his name—who was doing some stuff, but it was not really applied mathematics, setting up labs for doing mathematics. From time to time, I had tough relations with the math department. I'll give you a couple of stories.

Periodically, somebody from the math department -- the current head -- would call me up and say, "We really should get more cooperation." "That's a great idea. Why don't we get lunch and talk about it, blah-blah-blah." On one of those occasions, he said, "We've got this student who would really like to do computation; maybe you could supervise him?" I said, "Fine." He was a TA in math at the time, so I agreed to start supervising him to do his Ph.D.. Math promptly canceled his TA—he said, "Oh, Gear's got money—he can support him." That's great cooperation, right! I had to ask around and find some money.

The most extreme case was I had a colleague who was part of my research group, another faculty member, somewhat junior, and I've got a couple of papers with him, and he was in the same area I was in. He didn't have an appointment in math; he had come after we had had our own appointments. His appointment was in CS. We had one student from math who wanted to work in our area. He was a bright guy. He wanted to do a mathematics thesis. That's fine. So my colleague was supervising him, which was a courtesy-type thing, because he wasn't formally approved and there was a mechanism of being a *de facto* advisor and you had to have a *de jure* advisor on top of that. Anyway, I guess I wasn't even that because I wasn't at the final. The guy prepared his thesis and went into a final, and it was clearly a set-up by the math department. The committee, which was mostly mathematicians plus my colleague, looked at the first page of the thesis with the abstract, and that talked about a stiff equation having a large separation of eigenvalues. "*Large*…that's not a mathematical definition. We won't accept this as a thesis."

Well. The next day, I got the department to agree to admit him as a computer science degree student, agreed to accept his math prelims and his math qualifying as CS things, and we gave him a CS Ph.D. final that afternoon. He got a much better job with the CS Ph.D. than a math Ph.D.!

HAIGH: That's true, in terms of job markets of the two disciplines.

GEAR: Yes, right. Math at that time was having some problems, I think. Departments at Illinois opted to be organized with either a *head* or a *chair*, and they had very different responsibilities. A chair served at the pleasure of the dean, was appointed by the dean, and reappointed by the dean or fired by the dean; a head was elected by the department.

HAIGH: So would you have one or the other of those in the department, or might you have both at the same time?

GEAR: No, one or the other. A head had a lot of discretionary authority. Installed at the pleasure of the dean, you could run things. Nobody could tell faculty what to do; at least, you had a lot of control. A chair was elected. Everything had to go to the department executive committee. The chair could do nothing. Math switched from having a head to a chair, and you know what problems that leads to! So there was a lot of division for a while.

HAIGH: It seems that might be a convenient place at which to stop for today. We still have all your work at NEC to discuss; involvement with the broader community; authoring of textbooks; ODE work in '80s; and your time as department chair. But we can deal with it tomorrow.

[Tape 4, Side A]

Session 3 begins in Dr. Gear's home in Princeton, New Jersey.

HAIGH: This might be a good time to discuss your experiences during your sabbatical and visiting appointments in the 1970s. I think you've already alluded to your time at Stanford in 1970 in the context of your interaction with George Forsythe and the book that you wrote for his series. There may be other aspects of that visit that you would want to talk about.

GEAR: I went to Stanford partly because my good friend Gene Golub was there, so that made it a natural place to go. Of course, Stanford was an outstanding place. California was very nice -- as compared to Illinois weather -- was another motivation. I found it to be a very stimulating time. I spent half my time up at SLAC (Stanford Linear Accelerator Center), which had a very active computer group in those days. It gave me access to their high-end IBM—I think they had a 360 Model 90—on which I did a lot of the testing of the DIFFSUB code that I was developing. I taught a course at Stanford that brought me in contact with some very bright grad students, some of whom I had later interactions with at various times. Mainly my purpose in going was to get away and finish the book. That probably could have been done anywhere, but it was a lot nicer place to do it.

I taught a course on that topic at Stanford one semester. The Illinois sabbatical system pays you half a salary for a year and allows you to take a leave of absence for half a year and teach in that other time to cover your salary, so they paid half my salary. It was a very productive period for me. It was where I resolved the ideas about differential algebraic equations I had been struggling with, having time to think about all these issues. And wrap up the code for stiff equations and get it ready for publication. So it was a very productive time.

HAIGH: Were you ever tempted to leave Illinois for somewhere with better weather?

GEAR: I don't think I would make weather a main priority in choosing a job; maybe now I would, but I'm retired. At that time, what one can do in one's work was, perhaps, the most important thing. A secondary thing was schools and things like that, because I had young children at that time. Champaign/Urbana in those days was a wonderful environment for bringing up kids, maybe still is, because it was a relatively small town and not all those big-town problems. One didn't have to, as you would have to in New York or San Francisco, say, send your kids to private school because the public schools were impossible.

I was recruited at that time, that while I was on sabbatical at Stanford, I flew out to Cornell. They were trying to fill a spot on numerical analysis. But I decided that I really didn't want to move. I was recruited sometime around then, I think maybe just before that, by a combination of Berkeley and Lawrence Berkeley Labs, and I wasn't interested in pursuing it. From time to time I was recruited by a number of other places. I looked at it. My wife is also a professional, and often it was a two-body problem, in a way, so that was also a factor. I didn't take an offer until I moved to NEC, and we'll cover that later, I guess.

The time I spent visiting Yale was not a sabbatical. What had happened then was my wife was taking a sabbatical; at that time she was teaching art history in Champaign. She wanted to spend the time at Yale because of the Beinecke Library there—she was particularly interested in pursuing some materials that were only there. So she was going to be up there for a year, and in order to spend a little more time with her than would otherwise have been possible, I took a leave from Illinois and ran into some part-time jobs. I taught a course at Yale, an introductory course; I taught a graduate course at Yale another time. I was doing some consulting work for ICASE at

NASA Langley. I was actually shuttling around between New Haven, Hampton (Virginia), and Champaign/Urbana to work with my students, flying around those areas every week for a little while. I learned some new things.

At ICASE, which is on the NASA Langley base, there was a lot of interest (and still is, I imagine) in simulation of, like, air flight for training, and simulation of things like helicopter flight. They were also doing Space Shuttle simulation for re-entry. In all these cases, when you have a man in the loop, you're talking about real-time. That introduced a whole class of problems that I hadn't thought about before and were of considerable interest to me. I can't say that I ever reached answers that I felt were satisfactory. They had some unbelievably crude methods that worked very well. That was the nature of the problem. They were taking time steps that were equal to 40 degrees of the rotation of the fifth frequency—the fifth harmonic of the rotation frequency is very important in a five-bladed helicopter, for obvious reasons. No, sorry. The time step was 50 degrees in the first harmonic, and the fifth harmonic was very important, so they were taking a time step of 200 degrees, and the component was important. And it was just simply sinusoidally matched to get the accuracy.

HAIGH: So did anything from that feed into any of your later publications, or did you never get it to a stage that was publishable?

GEAR: I've got one or two minor publications; I'd have trouble identified them right now where I looked at this real-time issue. There are several concepts that were sort of what we thought were the things to use for differential equations at the time, ordinary differential equations, andthe human in the loop problem kind of blows all those out the window. One has the concept of stability and implicitness. If I'm solving the information from the human in the loop, I can't feed back the information to that person immediately, so there's no way I can be implicit. And, in fact, there's a time delay, the computational time delay to compute that information, generate the necessary graphics or whatever you're feeding back, and get it to the person. So you have to really start thinking about delay equations, and the stability of that comes in.

There are a whole bunch of other issues. The other issue was, one isn't interested in accuracy—you're not interested in getting ten-digit accuracy when you're simulating a fighter plane, especially with a human practicing it. Probably you don't even need one digit accuracy, in many senses. What's important is, in this case, is that the pilot has the *feel* of a plane that's like a real plane. Because, you know, you get in two different cars and they have very different characteristics, and you have no real trouble adjusting to them because they present the same *type* of characteristic, even though the values may be different. So the issue of errors becomes more a question of what do you mean by error? What is it you're trying to calculate? One has to really emulate the physical world very accurately, and what doesn't matter? I think what matters critically are things that do with the stability; it has to have similar stability characteristics. If I give you a simulator for a plane that's incredibly stable, and then you go and get in one and it's incredibly unstable, you're going to probably crash. So those things have to be done very well.

The paper I had that was called, "What Else Is Left to Do?" is interesting. [Numerical Solution of Ordinary Differential Equations: Is there Anything Left to Do, SIAM Review, 23, #1, pp10-24, 1981]. I was asked to give one of the "invited papers", or whatever they call them in SIAM, the major papers in a SIAM meeting up in Toronto. For me I always find that problematic. I don't like to repeat stuff; I hate to do a paper that's got anything that's in a paper I've already done. In a general meeting for a general audience, you can't give too technical details, so I couldn't give latest research results. I struggled for some time on what I was going to talk about.

What can I present to a general audience? So I thought, well, maybe I should try and survey the field from the point of view of what's been done and what's left to be done that might be useful. I quite frankly didn't think it was a very good paper. I guess I like papers that have solid technical results in them, not blah-blah-blah papers, and there's a little bit of that in it. I would say that it had a fairly large impact. I used to get lots of requests later on about, "What's the latest state on this thing because I'm thinking of having a student work on it?" So it did have an impact; maybe it was worth doing.

HAIGH: Let me ask you a little bit about that. You survey a number of areas. You say here are the ones we are now have got robust software for, or are close to having robust software for, and those include stiff problems, variable-step and order methods, problems with imaginary eigenvalues, oscillating problems. Then you get down to some problems that you say we still have major difficulties in them, and those are large stiff problems, variable order methods, multi-rate methods, and low accuracy methods. This paper was 24 years ago now.

GEAR: Yes, and so it was written probably a couple of years before that, if it was given at a 1979 SIAM meeting.

HAIGH: So a little over 25 years, then. Looking back your suggestion that these were going to be the big areas that progress could be expected with them, has that kind of prediction come true?

GEAR: For some, yes. With respect to variable order methods, I don't think a lot has progressed. I guess some. From the point of view of the numerical analyst, which I was very much, then, there's been progress in the large stiff problems because people are thinking about iterative methods of solving implicit equations and appropriate pre-conditioning, and so on. So now there are very good methods for large stiff problems, but they have to be very domain-specific. Once you start thinking about pre-conditioning, you have to know a lot about the structure of the Jacobian and so on, and that comes out of the problem. What's really happening in all these areas where there has been some progress, is that it's been in the various application domains (except in variable order methods, I don't think a whole lot has happened). There are massive numbers of codes now that do electrical network analysis, structural analysis, and so on. They solve enormous problems, and they solve them on massively powerful computers. Some of these computers in the national labs have 6,000 processors or more, running massively parallel, and using usually using some sort of pre-conditioned iterative method if it needs to be somewhat implicit. So there has been a lot of progress there. My guess is that might has somewhat peaked out. I haven't been following it that closely, but I sense that there is not much more to be squeezed out of the way it's being done now. My current work interest is revisiting the issue of explicit methods of stiff equations, which might squeeze out more, but only in the area of very large problems. But let's not get into that now. I don't think the approaches that are being used now are going to squeeze very much more.

Variable order methods, maybe was not so important, but it would be nice to have variable order methods in an automatic code. All the time problems are big. If I only had a system of five differential equations, it really doesn't matter how inefficient the method is because I'm not going to spend much computer time in today's terms on it. It's when I've got a system of ten million equations that I start to worry about efficiency. When I'm dealing with a very, very big system like that, I'm usually only looking for a digit or two of accuracy. The whole model isn't that accurate. Consequently, variable order is usually to try to get the most efficiency with very high accuracy. So I really don't think it plays much of a role.

Multi-rate methods are used under various names now and generally in quite a number of codes. I'm trying to think of some of the phrases that people use for them. But basically, it's obviously applicable to a multiprocessor. In parallel processing, you've got different processors. Each processor can be dealing with a segment of the system with a different time-step and they just periodically interact. So I think what hasn't happened I think is significant theoretical developments there. It's not clear to me that one can come up with any nice theoretical developments because it really is so problem-specific.

Low-accuracy methods -- that was where we got into this. That was what I was interested in with these aircraft simulations, amongst other things. If one doesn't want much accuracy, what is it you do want? For ten years, I was simply out of research while I was running NEC research labs, so I kind of lost track of that. I'm not aware of significant advances in there; I think it still could be important because I'm sure that there is going to be more and more simulators built.

HAIGH: Over this same 25-year period, are you aware of any areas of significant progress that you hadn't anticipated—unexpected directions the field took?

GEAR: You notice in that list, although I had coined the phrase "differential algebraic equations", written the first paper about it ten years earlier, I didn't mention it as an area. I had kind of forgotten about it. It later became a cottage industry and maybe it is even more than that, now. So there's one very obvious one. I'm sure there are others that will come to mind if I think some more.

[Tape 4, Side B]

HAIGH: I think you had one more sabbatical or visiting appointment after Stanford and Yale, which was in Nice in France.

GEAR: Actually, I had one other that isn't mentioned there; I just realized that. After that, I spent a semester at Argonne in I think it was either the fall of 1984 or the spring of '85.

Okay. Nice. Let me be honest: Nice is a nice place, and the people there, and there is a small university, but it was not a choice I made in order to enhance my mathematical standing. As a kid I was hopeless at foreign languages. I think it was because I was hopeless at any non-science subject—I wasn't interested and I wouldn't concentrate on it. I thought I really ought to learn a language a little better than I did, so I thought let me take a sabbatical in France and force myself, by immersion, to have to learn some of the language—more than I had in high school. And of course France is a wonderful place, so that was the main reason to go.

As I mentioned before, Illinois has an option of actually taking one semester at full pay, or you could take two semesters at half-pay. You could take a semester at full pay and a semester leave of absence, in which case you would need to find some more funding. Well, Nice offered me the opportunity to teach, but it would have been in French. I realized that if I had taught a course in French, I would have done nothing else except prepare for that course because my language skills were really inadequate. So I opted for a one-semester sabbatical. I had an association with Nice. As a matter of fact, I went down there once or twice a week for the weekly seminar, to use the library occasionally, and to use the Xerox machine. These are the important things, you learn! Otherwise, we lived 15 miles away or something. We had this fabulous apartment—somewhat expensive, but it was hard to find something—eight floors straight up from the ocean, at this place called Marina Bay des Anges . It went right through the building; the other side had a view of the pre-Alps, snow-covered in the winter and on the other side we looked out over the Mediterranean. It was a fabulous location with fabulous food. I was rewriting my first book at

that time to produce the second edition of *Computer Organization and Programming*, and I had a couple of small projects that I wanted to do. So I really didn't get anything professional out of the University of Nice interaction except meet some pleasant people and go to some interesting seminars. I even gave one seminar there. I got my book revised; I got a paper written. My schedule was very nice: in the morning, I sat in a fantastic apartment looking out over a plant-covered balcony over the Mediterranean, working on my book or a paper; in the afternoon, we would go out and drive around and look at all the fantastic sights there and just across the border in Italy; in the evening we would go out and have a fabulous French meal; it was a pretty nice life.

HAIGH: We have something very similar in Milwaukee [laughter], perhaps just a little flatter.

GEAR: So that was a good time. We were there a total of five months, actually. I think decided then, well, I didn't feel the need to put so much effort into it; I would try and satisfy a few other things. I had most of my life up to that time doing very little but work. It was probably one of the reasons my first marriage broke up: my wife left me because I didn't spend much time with the family.

HAIGH: Let me ask you. You mentioned that, in your school days, you had been very competitive. Did that carry through to your later career?

GEAR: It certainly carried through into college, and undergraduate college in Cambridge in England where it was still very much that way. I think less so in the American climate. Cheating always has been a problem in classes in America. In one or two, there was rampant cheating. One of the things I tried to do was to appeal to students. The better students would be helping the poorer students cheat, and I was trying to suggest to them that the grades were somewhat competitive, so if you cheated to help somebody who wasn't as good, you were actually hurting yourself. I don't think it made any impression. In Cambridge, I think, that wasn't even the viewpoint. The reason you're not cheating was that it was a competitive situation. It would be like turning around in a track race and giving the guy behind you a help over the next hurdle, or something. It may be a nice thing to do but, when you're in a race, you don't do it. The educational system is not viewed as a race here, I think. Maybe it's socially better. In some ways, it's nice that students get together to study and help each other; maybe that's a better view of it.

To get back to your question, I think I got less competitive from that point on. There's always a certain competitive instinct in trying to get papers out and be first out with new ideas and so on, and be the first to do something. I think if you didn't have that, progress would be a lot slower. What is it that drives most academics? It's not money, although I think we're treated pretty well. It's the desire to achieve at something. Most of us are lousy at sports, so we fulfill ourselves that way, so we enter some other race to try and prove ourselves.

HAIGH: So you remained very hardworking, but it was less directly competitive and more generally you were in a race with yourself.

GEAR: Yes. I think one almost starts to feel that your students are your output, too, so you can get pleasure in your students succeeding. I don't mind telling you, when a good student comes out, you feel a little pride in it. You start to see success in a broader variety of things. Even getting a good review, even feeling that the students appreciate you doing a good job has a certain level of pleasure. It doesn't have to be some sort of teaching award, but just if you have the sense that the students really appreciate your teaching, and some of them gave you that

feeling. You know, my primary job was to teach. That's what you're hired for in a university, although some of my colleagues didn't feel that way.

HAIGH: So tell me about Argonne, and then I'll ask you more about your teaching and your textbooks.

GEAR: Well, Argonne, that was in the spring of '85. I was scheduled to take over as head of the department at Illinois in summer of '85, and I had one sabbatical left. I realized that once I became head, there would be no time to do that for a while. So I took a final sabbatical. I took at Argonne National Labs for two reasons: they had a group, and a rather minimal parallel processor that I could work with and play on. At that time, the whole of the 1980s, my wife was working in Chicago and so she had to commute there. She had the sort of job that didn't require her to be there all the time; it was in the publishing area, mostly. Sometimes she would go up on Tuesday and sometimes come home as early as Thursday when we were living in Champaign. We had a condo in Chicago right downtown. By that time we had series tickets to the symphony and the opera and stuff like that in Chicago, sometimes a play, some theater group. I used to go up there quite a bit for weekends. So being in Argonne, I just simply lived in that apartment with my wife for the semester and commuted back out of town to Argonne. So there were two reasons, really, for the choice. One was the parallel computer group I wanted to work with, but the second was that it meant that my wife didn't have to move or I didn't have to go far away from her.

That was a pretty good time. As I say, it was also going to be my last opportunity to really get away from Illinois for some time, because I was going to be loaded up as the head of a department. We had massive student pressure by then. The undergraduate body, maybe it was only 1,600, I don't know. I think I said 2,500 the other day. I forget what size it had gotten to, but it was very large. We had the highest student-faculty ratio in the college—probably in the university, but the college by a long shot. Unfortunately, the dean took the attitude, and what he didn't say was, "Computer science isn't really engineering, so they should have that sort of ratio. They don't deserve any better." He wouldn't give us many more slots to bring us in line with the other engineering college, but he did allow us a lot of slots. We had about 30 when I started as head, and when I left five years later, we had 45 faculty, I think it was, so we added a lot.

So I took that last sabbatical; I saw my last opportunity to go off and do something by myself and also to do some real research.

HAIGH: What was the parallel machine that they had at Argonne at that point?

GEAR: I can't think of the name of the company—was it Denelcor? That doesn't sound quite right.

HAIGH: That was a company that made parallel machines.

GEAR: It made a couple of machines. They were all unusual machines. But this was in the very early days.

HAIGH: I think they only sold them to a couple of machine labs.

GEAR: Yes. And it wasn't bad. There wasn't much available yet. But it provided the opportunity to try out some of the concepts of parallelism. So I started looking to parallel methods of ODEs at that point. I was in the computer science department my whole career while I was at Illinois, and the difference between the various areas in computer science, in some sense, is broader than in most other disciplines. You take theoretical computer science, where people write a lot of very

short papers and take forever to review—the publication delay in theoretical computer science is like four years or something. You have people in software who have masses of students all writing pieces of large projects. Your faculty member might have twenty students they're directing, and they're not doing any real work themselves in detail—you can't.

So in numerical analysis I found, personally, that I had climb inside every problem that the student was working. I typically had about three Ph.D. students in numerical analysis at a time. I thought that was about the maximum I could handle. Sometimes I had as many as four or five in the other areas I was interested in—the graphicy stuff and the related areas and the system related to it. With those people, you just talk about general concepts and they report on things, and you try and move forward. But with the numerical people, if it's not going anywhere, you have to find out, and the only way you find out is find exactly why the theorem won't work or why the thing is unstable or whatever. Of course, that varies pretty much with the students.

For example, my best student in numerical analysis was Linda Petzold. She was a delight to work with because she would come in each week and say, "This is the area where I'm having some problems, and this is why I'm having the problems," and outline the problem. So with her, one didn't need to do much preparation. With other students who were weaker, you couldn't get anywhere. The only way to do that was for me to sit down for several hours, and in some cases every day, and try to move the thing forward myself. And in the case of the really weaker students, get extremely frustrated, because I would reach the point where I could solve the problem myself, but my job now was to try to get the student to come up with the solution. But like I say, I was teaching them; I was being paid to do that.

HAIGH: So I'm hearing those thoughts were prompted by thoughts of the unusual diversity of computer science, which in turn were prompted by exposure to parallel computing at Argonne. There was a machine called the Denelcor HEP, Heterogeneous Element Processor, designed by Burton Smith. [The Denelcor HEP was the first commercial system designed to apply multiple processors to a single computation, and the first to have multithreaded CPUs]

GEAR: That's right; that's the machine we had.

HAIGH: Did your original observation, perhaps, have something to do with being exposed to another side of computer science at Argonne that made you think about broad the discipline was?

GEAR: Well, I'm not certain actually why I went on that train of thought there. My purpose at Argonne was to expose myself a little more to parallel computing. There was activity there in Illinois, and I could have done it there. ILLIAC IV had been there and I hadn't been involved in that and other things. But it's the issue of being away so the students can't come walking in your door with issues and the staff can't come in with problems. You're not teaching. It's really just a question of getting away from all of that so you have time to think about something new, which takes a lot of thinking.

So now I remember why I went off. For me, when you're in a faculty position and doing the things faculty have to do, like supervising theses and teaching and so on, you're spending an awful amount of time; there's really not much time to dig down deep. You have to dig into the students' topics and you don't have time to pursue your own research, pretty much. And that was the last opportunity, because I was about to become department head and we were expanding rapidly.

HAIGH: So did this exposure to the work that was going on at Argonne and to parallel computing lead to any publications or big developments?

GEAR: Sometime around then I had two or three papers. I talked about parallelism across time and parallelism across space because I had been invited to various conferences. It certainly came out of my thinking about parallelism while I was at Argonne. [Massive Parallelism Across Space in ODEs, J. App Num Math, 11, pp27-43, 1993 and Parallelism Across Time in ODEs, J. App Num Math, 11, pp45-68, 1993]. I don't think they were very deep papers; they were very shallow papers. I have to say, however, virtually all the papers I read were also very shallow. [Chuckles] In the early days, you were still grasping for ideas, and there was not much theory in any of them. It was not, I felt, an area where you could get a lot of valuable theory at that time. We were still trying to ask: What is it we need to know? What is it we need to show? What can we do? So I think those concepts were picked up by quite a lot of other people and used. My guess is if one hunted through literature enough, I'm sure other people must have come up with the same concept, maybe with a different name.

HAIGH: Were there any people at Argonne that you were working particularly closely with?

GEAR: No, I wasn't. There were no people there, as far as I knew, that were pursuing differential equations. People like Jack Dongarra were still there at the time, I'm pretty certain. Well, he certainly had been involved in that. But they were mainly interested in generating the next layer above the system to support parallel processing—message passing interfaces and all that stuff. So they were mainly interested in trying to create a system or systems that allowed you to do the parallel processing, the debugging tools you need. I was really more interested in the applications.

HAIGH: Actually, I think you had mentioned earlier the specifics of your earlier visit to Argonne in terms of this ability it gave you work on stiff methods and the software that came out of that. I don't think I asked you just generally about your impression of Argonne as a place where people were doing research. And then as you returned in the '80s, it might be interesting to hear about whether anything had changed in terms of the kinds of work that was being done there or the conditions that people were working in from the '60s to the '80s.

GEAR: Well, I remember one of the wonderful things in the '60s about Argonne: research money was pretty much unlimited at that time—post-Sputnik, the US government had ramped up funding for research because they were losing the space race and the arms race and everything else. Argonne was a wonderful place. There were few competing labs then. The Fermilab hadn't been built yet, and there were other places that hadn't ramped up. Reactors were still important and we were doing stuff there. And I think one of the most impressive things about Argonne, initially it'd been an old estate. In fact, I think one of the few herds of albino deer in America roamed it. It started to do some of the reactor work and associated with the work of the University of Chicago—the atomic reactor. And so initially there was just a bunch of old Quonset huts and crummy old housing like barracks housing—you know, the sorts of thing that you built during the war-time and thereabouts. And by the time I got there in '65, Argonne had a lot of nice new buildings, and they were all for the science. All the administration was in this crummy old stuff. You know, and that sent a very strong message. You go to some places, and the administration is in the latest, jazzy new building, and the scientists are tripping over each other in some dreadful stuff. I've seen that in a number of cases. When you see those two things, it tells you something about the place. Here really were a group of administrators who saw their job to make it possible for the scientists to do the science. And that's wonderful. And you really had that sense at Argonne. The administration was there to serve you and make it work. So there was a lot of good stuff going on then.

Back in that time, I was in what was then called AMD. It was the Applied Mathematics Division, but in fact, it ran the computer center, because in those days, the computer center was in most universities and places, still being run by some academic or research group that happened to be the first to want to use computers. So it was very integrated there. The same was true for Illinois at the time; the computer center was run by the Department of Computer Science. I used to get fairly sick of going to parties and having somebody come up and say, "Oh, you're in computer science. Let me tell you about the problems I'm having with the computer." [Laughs] I said, "I'm having the same problems."

So, by the time I was there in '85, as it happened everywhere else, these organizations had been split. So I think it's now MCS or something (Math Computer Science), the research part of it. And it was somewhat smaller. Of course, the advantage in the old system for a lot of people was that there was a lot more money associated with the operating centers. So some of the money is fungible, so it often enabled a lot of other things to be taken care of very easily.

And there's another problem. I don't know if it happened at Argonne, but it was a strong problem at Illinois later on and a strong problem nationally in the CS, which affects computational science, too. That is in the view of most of the university away from CS, supercomputing is computer science. So for example, when Illinois started NCSA and got all of this money from NSF to put a supercomputer there to be used also by other disciplines, the view of many people outside of those closely associated with it in the federal government agencies in the university was, "Oh, CS has all this money. They don't need anymore support now." Well, you know, CS wasn't getting a penny of it. It wasn't doing the stuff we were doing; it was providing computer service. So there were some drawbacks to that. I don't know if that was a problem at Argonne. I would say that you had a sense that the research was more focused by then. Of course by then, the government had started to put a lot more pressure on research groups—you know, Budget Memo A21, which goes back even into the 1970s, and all sorts of other things that commanded that the research be serving a particular need. I'm more in favor of research being useful. I'm a very strong proponent of scientists trying to solve some real problems. We can actually talk about that when I come to the NEC stuff; it's particularly relevant there. But I think the government approach to it during the late 1970s, '80s, and '90s was very, very misguided. Let's delay that topic for a little bit.

HAIGH: All right. Well, that then, I think would conclude discussion of Argonne and your various sabbatical and visiting appointments. So I think you've spoken already a little bit about your philosophy of supervising graduate students, and we know your feelings: even if that's frustrating, it's an important part of the job that you're being paid for. Perhaps you might have a little bit more to say about your philosophy as a teacher. Then I think that will also go naturally to a discussion of your various textbooks.

GEAR: Well, coming from an undergraduate degree in Cambridge in England as I did, my initial thing was to want to approach teaching that way, which is lay the stuff out in front of the students and be prepared to help them, and in the end, say it's your responsibility. I modified that view over time to feel that it was part of my job, also, to build them up to a maturity level where they could make those decisions. And I couldn't start that way. You know, when I went to Cambridge in England, I think the population that went to universities in England was maybe about 3%, and at the time I came to America, something like 30% of US students were going. I don't know what the numbers are now. But so you've got a very different cross section to deal with, and you've got to recognize it.

By the way, when you're talking in doing introductory courses (and I taught things like introductory computer science from time to time), my initial approach to teaching was you don't be dogmatic. You say, "Here's a way to do things. Here's another way to do things. Here's another way to do things." You know, there is no absolute truth in those. I mean, two plus two is four. That's an absolute truth. But as to ways you can do things, most of the time, there are many ways, and we're really making a selection. And that failed miserably at the lower levels, and I think I know why. I can't help but think the right idea is that the first time through, you say, "This is THE way to do it." People can't stop to compare things until they have a base to compare against. To my mind, for example, the topic of structured programming became very big in 1980-something. I don't know.

HAIGH: No, a little earlier, I think. I think Dijkstra's famous letter to the editor was something like 1969, and there were books on it by '72. [Dijkstra, Edsger W. "Letters to the Editor: Go To Statement Considered Harmful." Communications of the ACM 11, no. 3 (March 1968): 147-48].

GEAR: Was it? Okay. So then it'd be by the mid-'70s that it started. So you know, top-down programming became the big thing. And I have never subscribed to some of those concepts for classroom use. They're fine later on. But to suggest that you do something top-down suggests that you already have enough knowledge to lay out the structure in broad terms and then to start going down. Until you have a sense of what's going to be down at the bottom of each of those, you can't do that. So from an educational point of view, I think you really have to start bottom-up. And this tree (an upside-down tree in computer sciences) has many, many branches and all the possible ways of solving a problem. I've now changed the use of the tree. [Chuckles] But you first off teach a student starting from one bottom branch, one leaf, and moving up and showing all the things, and this is one way to do it. Then you start showing some of these other ways, until they get the bigger picture. So after some failures in the way I was trying to teach, I abandoned that and went very much to saying, "Okay. This is THE way to do it."

Let me give you an extreme case. I was teaching introductory computer science several times. And I had been asked at one point by SRA to write a textbook in that area, which I did. And I thought, undergraduates had pretty good entry requirements by then; Illinois was getting more selective, well, one shouldn't be teaching just how to write Fortran programs or how to write programs. You should be teaching something higher-level. So I thought, well, why don't I try to teach the concepts and then illustrate it with simple language. I'll teach a course where I teach them, you know, "Here's how you do it in Fortran. Here's how you do it in Pascal," as I went through. That was a real disaster. You absolutely cannot do that. I thought, well, I can do this comparative thing, which is to say there are various ways of doing things, and these are how they appear in different things. You can only do that, I think, by first of all teaching one of the languages, and then coming back and saying, "Okay. Here's another example." I think I'd say that that applies to almost all sorts of levels. You have to give the details and make sure people understand something well before you can move up to give the broader picture.

Just the other day, I was reviewing the draft of a manuscript for a publisher. It was a high-level, almost research-monograph level, I'd say. But nonetheless, it's the thing you expect to be used by grad students. Like many, many books I've looked at, it suffers from the problem that I don't think the author, when he writes, has thought about, "Okay. If I want to talk about this now, what have I already told the student? Do they have the underlying details?" To my mind, that is a problem with all of teaching, especially from the top-down model. You talk about these broad

ideas, which are absolutely meaningless to a student. The human mind seems to take small details and build bigger structures up on them. It doesn't go, I believe, the other way.

HAIGH: And that's true. I think to some extent, though, the attractiveness of that approach may have been driven by the need of computer science to prove that this is a real discipline, with its real high-level ideas or general principles, and it's not all just about teaching students how to program.

GEAR: Well, possibly. No, no. Structured programming is important after you've learned of all the details. If I want to go and write a program now, I'd do it top-down. That's because by now I have a… Well, and that's not always true. If I'm going to write a program where I want to do something (and I've spent a lot of time in the last few weeks looking at a particularly numerical issue), if it's a very straightforward thing, if I was writing this like on a piece of accounting software, where there's no question about whether you can do the arithmetic at the bottom, then I would lay it out very broad level. But if I was solving some very complex technical problem, where I didn't yet know how or whether I was going to be able to do the stuff at the bottom level, first of all, I would need the mathematical theory that says, "Okay. This is how I can get it done. And what do I need to do it?" And then I'd just subtract these things, until at the top level I could say, "Okay. This is the organization." Now I can start laying out the program. I have this main program and these subroutines, and start filling in the details. And in that case, I may often write a program at that point in a typical structured fashion, I'll write the main program and the subroutine calls, and the subroutines will simply be calls back to me type something in at the terminal just to check out that I've got the right structure before I get into details. So it's very important, but not when you first start programming, and I think that was a problem with a lot of the approaches to instruction.

HAIGH: Were you teaching these introductory courses because you wanted to, or because everybody had to take a turn?

GEAR: One of the things I liked about the department at Illinois, well first of all, the course load was one course per faculty member each semester, so it was very nice. But we took the attitude—and it was applied almost universally—that everybody had to take their turn. It was not a case where the senior faculty just taught their specialty, so senior faculty picked the courses they wanted first, and then the junior faculty took all that was left over what was left over. In fact, generally speaking, when the junior faculty came in, in their first year or two, they would give them some time off and give them much more opportunity to teach their specialty only and not asked to teach the lower-level classes.

Another thing was, later on, the introductory computer programming for engineering or for business, the students had one faculty lecture a week in maybe to a class of 100-something people, and there were five such things (you know; there were a thousand students), one TA session in a much smaller class, and then time on, I think they used PLATO for some part of the time or other stuff like that, and then, of course, they did a lot of computer homework. So the faculty load, for example, for teaching, since there were five class sessions, all the same thing, was the person in charge who had to supervise all the TAs, of which there were a hoard, and so on, taught two of those one-hour classes, and the second person taught three identical classes. In fact, on one occasion, I taught the same class on Monday at 10 a.m., 11 a.m., and 1 p.m., and it was taught over in a building which had a big running track in it. I went and relaxed for an hour running between them. So Sunday night I prepared my overheads, gave the same lecture three times to three different groups of 100+ people, and by Monday afternoon at 2:00 I had the whole

week ahead to devote to my grad students and do research. So it's really an easy load, which I found very attractive. So I didn't mind teaching it at all, because with all the TAs and somebody else in charge, one didn't have a stream of students coming in, because you probably couldn't have helped them with a lot of the questions they had about, you know, "Where do I turn in my homework?" So it was a pretty good deal.

HAIGH: Then you also produced this stream of textbooks beginning with *Computer Organization Programming* from McGraw-Hill in 1969.

GEAR: Yes. That was the first one. When I went back to Illinois, I was running the ILLIAC II software effort, and there were very few computer courses. I mean, when I've been in there before, there have been like two; I think, or a couple more now. I think by then we were teaching programming for the IBM machine, probably at the sophomore level, something like that. Computer Science then, you know, there was a machine language programming course basically, and a computer organization course was the next course. And I was teaching it, and I taught that initially on ILLIAC II, and then on the 360, then on a PDP-11, and that was a part of the typical CS curriculum in those days. Since I was heavily involved with doing machine language programming for the ILLIAC II and other machines, it was a natural thing for me to teach, and I was interested in it.

And I found that whenever I taught something, I always tend to start making notes. Not to use in class, but I find that even with overhead transparencies, I can never use the same ones twice; they just have to be fresh. But I started making a lot of notes to myself, and it became pretty clear that when publishers came around, a) trying to sell you books, and b) trying to find out if you have some, that there was really nothing available at that time that was any good. I didn't have a textbook, so I probably had enough notes to start distributing, so publishers started to glom onto it. In fact, I said, "Okay, maybe I'll turn it into a book," and then it was almost manic. I remember one week where about three different publishers came through, wanted to take me out to lunch; wanted to take me out to dinner; wanted me to sign a contract. I kept saying, "I'm not going to sign a contract until I've finished it and I know can do it and I'm happy with it. Then I remember I went to one Joint Computer Conference in San Francisco, and again, there were no books out there at the time, and the publishers were desperate for a book. I remember one, I think it was Prentice Hall, took me out to a fantastic dinner and poured way too much wine into me, then at the end of the meal, pulls out a contract for me to sign. I said, "I never sign anything after a drink!"

Yes, so finally, I signed it a little earlier than I'd intended to because I was losing so much time to these damned publishers coming through. I thought, "I've got to get them off my back so I can finish writing." So I guess it was probably finished about '68, and its production takes nearly a year and it came out in '69. It was written mainly about the course I was teaching, and I found always that to write something like that, I had to be teaching and have a strong sense of what it is you want to do in the book; otherwise, I don't think I could write a book. I'm thinking about writing a book about the stuff I'm doing, collaborating with this guy in Princeton, and I'm not certain I could do it without teaching a course on it. I think it requires the discipline of trying to bring it together and organize it for an audience to do that, for me anyway,.

HAIGH: So the first book, that would have been that combination of computer architecture and assembly language programming?

GEAR: Yes. It was primarily assembly language programming, but it was architecture in the sense of what architecture was in the day. Those days, I talked about zero-address machines, which were stack machines: one-address machines, two-address machines. In fact, some of the problems in the end of the chapter there are from examples that I created with the IBM 360 committee to analyze some of the various designs we were considering. So some of that book came out of that work, so you never know how something you might be doing might turn out to be very valuable for you later on.

HAIGH: Did the publishers and their enthusiasm for the book translate into the kind of sales and impact they had hoped?

GEAR: I don't know. I mean, it was not an introductory level book, and computer science wasn't yet much of a discipline. But it did very well. I think the first edition sold some 50,000 copies, which is okay. Back in those days, I think in 1969 my University salary was like $20,000, and so the book royalty was a significant hunk of income for me. I think one year it didn't match my salary, but came close to it. So in those days, that was quite something. I hadn't written the book to make money, but I didn't object to the checks!

HAIGH: Did you get any feedback ideas of where it's been adopted, how it was used?

GEAR: I guess I used to get a list of that; I didn't keep it. I mean, I think it had been used quite a lot because there really wasn't any competition initially. Fairly quickly thereafter, well, a year or two later, some other books came out. I won't mention the name of one that came out—I know the person. I would say, to some extent, it had been written because the publishers say, "Hey, we need a book like that." I looked up one thing in his book-- I'd found an error in a table in my book, a table that showed BCD codes, ASCII codes, and other stuff like that for various things, and I had some stupid error in it. I looked in this book and it had the same error—it just copied my table, and didn't acknowledge it. But you know, I copied my tables, I guess, from some IBM manuals or something or whatever else they are, and I'm sure I didn't acknowledge them. So when you're copying a table like that, it's just common knowledge, it's not peer review. It's not like you're stealing a theorem or a research result.

HAIGH: That's not something you think of as having creative input in the first place.

GEAR: Right. Well, it's creative—I'd miscopied the stuff!

HAIGH: More creative than you realized, then!

GEAR: Yes. I wonder if that error's still around in some book, having been copied from one to another. [Laughter]

HAIGH: So you don't have any interesting stories about being approached by people who were using the book, or hearing comments from students later, or anything like that?

GEAR: Well, there was one story. That book was written in the days before there was sensitivity to gender issues and things like that, and in fact, most of this stuff was initially written probably before 1967 and in the last days I was just cleaning it up, and I should have cleaned this piece out. I quoted something that was probably said around DCL when I went there in '56. In fact, I think it was probably first stated in the first programming course I took or something like that. It was trying to distinguish between the actions of programming, which is laying out things; coding, which is moving into some sort of machine language, in those days, a machine-language expression; and just entering the thing into punched tape. So it was a definition in which "I program, you code, she keypunches." In the 1950s and early '60s, women wouldn't bat an eye at

that, and I put it in the book, which I regret. And early in the 1970s, I got a letter, an ornery letter—I think it was from one of the Michigan universities; maybe Lansing but I don't remember—from a woman complaining about this blatant sexism. She was absolutely right, and I wrote and apologized and changed it.

[Tape 5, Side A]

HAIGH: So I think that's covered the first edition of the textbook, *Computer Organization and Programming*, and then there was a second edition in 1976, a third edition in 1980, and a *Personal Computer Edition* in 1985. So how did the book evolve over that period? Obviously, the field wasn't nearly as open as it had been when you came up with the first edition, so did the book evolve into a more particular niche?

GEAR: Yes. The first edition, as I recall, mentioned two specific machine architectures, maybe it only mentioned one. I don't recall what they were, but at various times, I talked about the 360, I talked about the PDP-11, I talked about CDC-6600, and I don't know if I talked about any more. I talked in general terms about architectures, and that changed with time. Oh, and then finally, the Personal Computer Edition, of course, talked about the 8086/88, et cetera. By then, I have to say that I was getting pretty bored. The publisher was pushing me to revise it, but there was more and more competition. When I first wrote it, it was still talking about stuff that was pretty much on the leading edge of what was going on in those days. You know, it talked about various organizations of channels and swapping, and things that were leading edge. But by the time I got around to the *Personal Computer* edition, it was very routine stuff and it had very little interest for me. It was not my area of research. I just didn't feel like it was worth putting in more time. It was financially worthwhile, but if I wanted to do things to make money, I probably wouldn't have gone to the University in the first place. So, simply, I said, "No more." Let's see, 1986 was the last?

HAIGH: 1985 was the PC Edition.

GEAR: 1985. Well, yes, okay, so at that point I was about the become head of the department. That was taking all my time on some major projects there. What time was left over I had to deal with students, so I quit textbook writing.

HAIGH: So then your second textbook was *Introduction to Computer Science*, published by Science Research Associates of Palo Alto in 1971.

GEAR: Actually, the ODE book was my second book. It was sort of a textbook. The ODE book, as I mentioned, was done while I was at Stanford. [Numerical Initial Value Problems for Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs, 1971].

The next book, which was for introductory computer sciences, a lower level textbook, may have violated my own ideas that I said earlier about only writing books when I felt very strongly about. What happened was that I came back from Stanford in August of 1970, and in early 1970-- something seems wrong with this sequence. So it must have been in the fall of 1970, so it came out fairly quickly. They were in a hurry to put it out, I guess. Sometime in the academic year of 1970-'71, probably late '70, representatives from a company called SRA, Science Research Associates, which I think had been a spin-off from IBM and was getting into the textbook business, came and asked me if I'd consider writing an undergraduate introductory textbook.

Why did they ask me? Well, the guy who was the primary editor there at the time was the guy that had been the editor when I'd signed for McGraw-Hill for my first book, so he knew I could

write something that would sell. And they were starting out with a plan to get into the introductory development in a number of areas, and then their business approach had been to look at what seemed to be important, what they wanted to compete with, and to get an enormous number of reviews from various places using those books to see what people thought was wrong with them, what was needed. So they had all this material as to what they wanted, and would I write it? And I probably would have said no, because this is not the way I think I really would like to write a book. Now, this was in late 1970, and in October of 1970, my first wife had left me, gotten a divorce to go with somebody else. I made a financial settlement, which had left me with somewhat less cash than I had before, which hadn't been a lot anyway. Basically, we settled where I kept a third, she took a third, and I put a third away for the kids for the future. So they were offering a very attractive advance, attractive for those days, and I was probably a little seduced by the money at that particular point in my life. And, you know, there was all this support for it, and I had written two books already, so I felt like I could probably do it. So I sat down and I wrote it.

HAIGH: Did they come to you with a table of contents that outlined what would need to be in it?

GEAR: Not quite. They had their main competing book, which had been done by a committee that had been formed by George Forsythe. But it was an approach where there was a hardcover book, that covered the principles, and then there's some associated programming manuals in different languages. So when you used it, say you took the book, and if you want to teach FORTRAN, you took the FORTRAN manual, and so on. What happened was I wrote the book…and I'd have to actually go back and look. There was a BASIC manual, a PL/1 manual, a FORTRAN manual, I think, and then maybe one other. I co-authored some of them, somebody totally wrote one of them, and I think I wrote one myself, but it's long enough ago, I don't recall.

HAIGH: All right, so I think this is actually two separate books. So what you're showing on your résumé is *Introduction to Computer Science* as a stand-alone book in 1971, and then in 1978, there's, *Introduction to Computers, Structure Programming and Applications Series*, which is this set of books with multiple volumes on different topics.

GEAR: Do I have stuff in the wrong order? You want to stop the tape a minute and go look? [Tape stopped]

HAIGH: We now have the book in question in front of us, and so we're seeing that the 1971 book is a stand-alone text, and then we have here one volume that is not written by Gear, *Introduction to PLI Programming in PLC*, and you believe there was also a FORTRAN introduction.

GEAR: There was a FORTRAN version of that, which I wrote, and there was a BASIC, which I'd co-authored with somebody else.

HAIGH: Yes. Now, looking at the list of topics in the table of contents, we have introduction; a chapter on computer organization; two chapters on flowchart language; a chapter on computing systems, which is giving you an idea of what an operating system and a monitor and an I/O device are; errors and debugging, data structures, non-numerical applications; and numerical methods. This was very early in the history of computer science, so would everyone have agreed that those were the nine main things that should be covered in an introductory textbook? Or do you think other authors would have taken a different approach and it was still partly unclear what should be covered in an introductory course?

GEAR: I think people would have agreed on some subset of what was there. You know, when you have a textbook, you don't just say, "This is the course and all of it's the course." You will allow some variations. Some people stressed more the numerical stuff; some would have probably stressed the non-numerical stuff more. It was more material there than you could put in a typical course, and I think you need to do that in a textbook so the instructor has some flexibility. I tried to produce, for example, in the non-numerical/numerical chapters some example problems that illustrated some concepts that you could use.

HAIGH: My impression is that a little bit later, Chapter Seven, Data Structures, which has got recursion and strings and things, and then Eight, which has got some trees and so on, would have kind of expanded to be thought of as more central. Like Wirth, *Data Structures and Algorithms*, that kind of thing. Mundane things like debugging and flowcharts might have shrunk in terms of people's impression of what computer science was.

GEAR: Yes. These days, most students come in knowing already a hell of a lot about programming, so a whole lot of this stuff is redundant, and you don't deal with computers at the lower level we dealt with them then, so we don't see that.

HAIGH: You have the dedication there?

GEAR: My first book, the *Computer Organizing and Programming*, the first edition was dedicated to the name that I called my first wife by, which was Sue. By the time the second edition came around, my first wife had left me and I was with my current wife, and so that's dedicated…it says, "To Ann, the second edition." There's a little story behind that one [indicating the Introduction to Computer Scence book]. This one says, "To the near Ms." Of course, MS means either manuscript or Ms., or could you pronounce it just, "miss"? The time, as I mentioned, that I was approached about writing that book, I was single. My wife had left me, and some of my money had left me, too. I had a brief involvement with another young lady who, actually, I met on the evening after my divorce proceeding, which I went to—I figured if I was paying for it, I might as well see it in the court. Then I met a male friend of mine at a local bar on campus, and a friend through a girlfriend he had at the time came in, and somehow she cottoned on to me. But I think that was her plan. So I was quite heavily involved with her for about a year while I was started on that book, and in fact, most of the time I was writing it. It was fairly clear that this was not something that was an appropriate long-term relationship for me, and I think she had other ideas. In a nutshell, at the end of the year I said goodbye. So you can interpret that as the near miss., or you could also interpret, because after that, I met Ann, my current wife, and you could also interpret it as, she was, by then, the "near Ms."

HAIGH: That's one of the more enigmatic dedications.

GEAR: It doesn't mean anything to anyone other than me. I always have to explain it. I don't think I'm going to be able to write another book until I come up with a good dedication!

HAIGH: So you had this kind of mixture of more theoretical and more practical things, some flowcharting. Did you then have an opportunity to use this book in your own teaching to see how students reacted to it?

GEAR: Well, two things. I don't like teaching out of my own book. I find the same thing that happens to me with overhead transparencies: I make them when I give a class; if I try and use them again the next year, I find I don't like it, I want to do it a different way, and they're not fresh in my mind. I have used one or two of my books, and I may have used this, but I try to avoid teaching a class once I've written a book. The other thing, you know, I feel awkward

assigning a book I've written to a class for financial issues. I asked the University about trying to find some mechanism to get around this. You can't find out what the sales are or anything like that. There's no way of finding out how much money you've profited. So I got into the habit of making a donation to the University foundation any time my book was used in a class at Illinois, whether I was teaching it or somebody else was teaching it, to try and be clear of any accusation of cashing in somehow. I probably used it once or twice; I don't remember.

HAIGH: You've explained that basically you wrote it for the money. But were you satisfied with the end product?

GEAR: Fairly well, and it did pretty well. It certainly satisfied one of the goals for writing it. Not an enormous amount, but it recovered me from my temporary indisposition. And I liked it enough to be willing to revise it. The next one wasn't exactly a revision. It was repackaged into a series of small books, paperbacks, the idea being that the instructor could then choose a few of several books. We had applications in different areas, so it could now be if you were in engineering, you could take the applications in engineering; if you were in business, you could take the applications in business. But they were tied back to the other sections, so in the sense, it broke that book down, took the application stuff and specialized it to certain areas. I stripped that book into pieces and updated each piece. [Introduction to Computers, Structured Programming and Applications Series, Science Research Associates, Palo Alto, 1978.A set of books that included: (1) Algorithms and Applications in Computer Science, (2) Algorithms and Applications in Science and Engineering, (3) (with F. Gustavson) Algorithms and Applications in Business, (4) Computers and Systems, (5) Programming and Languages, (6) Pascal Programming, (7) Fortran and Watfiv Manual, (8) PLI And PLC Manual , (9) Basic Manual].

HAIGH: Do you have any sense of how successful the revised multi-volume version was?

GEAR: No. At first I used to keep track of sales numbers—it's like keeping track of how many papers get published, I guess. I stopped doing that after a while. Of course, hey are all long since out of print, and I had no wish and no desire to write anything like that ever again. If I wrote anything more, it would be strictly about the sort of stuff I'm doing in research now.

HAIGH: This might be an appropriate time, then, to jump for a little while outside your personal teaching and research career and talk about your involvement with the broader mathematical software community and the various professional associations. After we've covered that, we'll come back and talk about your time as department chair and then your time at NEC. We already broached that topic just in the context of computer graphics, and the interesting fact that you were for two years editor of the SIGGRAPH newsletter. I don't think we talked about your involvement with other parts of ACM at all, so that might be a place to start.

GEAR: Well, sometime in the early '60s, ACM started to have special interest groups, SIGs, and one was formed in numerical analysis. I think Joe Traub formed it initially. I don't quite remember.

HAIGH: That's correct. I've interviewed him, and I actually was able to get some of the early issues of the newsletter scanned into the ACM Online digital library as a result.

GEAR: That's interesting. Anyway, so that was my first involvement. I joined ACM early on, and I joined the British Computer Society, too. There are only two societies that fall in that area. So I was somewhat active. What was SIGNUM? Well, it was really just a newsletter, and I think we'd get together. I assume there was some sort of board and the chair. I'm sure I was on the board several times, which I've never even bothered to record because we didn't do anything;

we'd just get together at meetings and talk about stuff. And then we started to use it to organize one or two meetings. In that time, one reason for running meetings was it's a fundraiser for that group, and so we organized a couple of ODE meetings. That was a secondary group of people interested in numerical ODEs. We had a meeting in Urbana, which I organized, which we had something like over 100 people. In fact, we'd planned for a maximum of something less than 100, and demanded that people pre-register.

HAIGH: Let's see. Before that, you chaired a couple of meetings in graphics in 1967 and one in 1969 on microprogramming.

GEAR: I was one of the early members of SIGGRAPH when it consisted of very few people—half a dozen, a dozen people would get together in a meeting and talk about things. As you mentioned, I put out a newsletter, but it was just a mimeographed report on computer science, but I didn't have report numbers and I don't think it's been filed anywhere, and those are gone. You know, I just agreed to put the thing out because I had the resources to do it at Illinois.

And in 1967, it was that first conference in Illinois. What happened was 1967 was the 150th anniversary of the University of Illinois's founding, so there's some big plans to have all sorts of meetings and people were asked for their ideas. I think David Pines, a physicist, was kind of in charge of synchronizing stuff, so I proposed the meeting in some area, and he came back, typical thing about having some very general meeting that nobody would have been the least bit interested in, so I just kind of ignored that. But we did go ahead with the meeting. I knew most of the people in graphics then—it wasn't a very large community. So we organized a meeting. I'm sure I must have gotten some federal support to pay for it, and we had about a dozen or so invited speakers, and so we must have paid their way and expenses. We had probably a couple hundred people for a couple of days at Illinois, all invited papers, and we put together a conference proceedings. I don't remember if I co-edited either of them or not. I often got somewhat younger faculty to do that, not because I was trying to slough off the job, but it was useful to them by then to have the name "Assistant Editor" on someone like that. It was a very successful meeting, so we ran another one a couple years later.

HAIGH: All right, so the first one was "Emerging Concepts in Computer Graphics," and then the second one was "Pertinent Concepts".

GEAR: Yes. I was interested in some of the mathematical questions in computer graphics, mainly, and the field was moving much more to the *gee-whiz* stuff at that time, so I kind of lost interest in a lot of that stuff. But I was pretty much interested in ODEs at that time.

About, the microprogramming workshop: I knew something about microprogramming—I helped work on the first IBM machine. And there's a bunch of grad students who wanted to run an ACM microprogramming workshop, and I agreed to chair it for them and help organize it.

HAIGH: So that was the fifth annual workshop on microprogramming in September 1972.

GEAR: I think so. I think that was probably SIGMICRO, something like that, although I don't think I was even a member of SIGMICRO.

And ODEs are always active, so we had a number of meetings. In '75, I organized one and it was held in Sunnyvale; NASA provided the space for it. There was a lot of interest in it, people wanted to come, and we probably invited about 100 people or lower, maybe 80 or so. Given that people would provide the free facilities to have the meeting and we didn't have a whole lot of expenses and we could get away without charging an enormous registration fee, like $10 or $15,

which was considered high in those days, we could get a small amount of budget for SIGNUM out of it. So it was an opportunity to get together with people in the field and also help SIGNUM. I don't know if I was chair of SIGNUM at the time. I was chair at one point and I don't remember when.

And then later on in '79, I decided to have a meeting in Urbana since I was very active in ODEs by then, and it'd probably be a good place to have one. So we got a lot of attendance there. We'd gotten space in the hotel that would accommodate maybe 90 or so people adequately, and we demanded pre-registration—not for a big fee, but to try and control things. And a whole bunch of people that had flown in from Europe showed up without having pre-registered. Well, you can't exactly tell people who've flown from Europe to Champaign-Urbana, "No, you can't come in." So we really had space problems. I had to increase the counts for meals at the last minute. We were really chasing around. But my recollection is it was a very successful meeting.

HAIGH: So would the attendees at these kinds of meetings all be researchers who were involved in coming up with methods for differential equations and writing software, or would some of them be people who were users who were interested in getting a hold of the latest thinking for their own application purposes?

GEAR: I don't think there were any users. This was the worldwide ODE theory and software community, which at that time was 100-something people, 150 people, and so it was an opportunity to get together periodically.

HAIGH: Through this period in the '70s, who would you say were the leading people in it?

GEAR: Well, I think the biggest figure was probably Germund Dahlquist from Stockholm. He was certainly there. By that time, Urbana had a pretty good reputation in the area. Sometime in the early '70s, Bob Skeel had done some outstanding work. He's now retired from Urbana and gone to Purdue. He did a PhD at University of Alberta in Edmonton, and I guess they require an external examiner on the final. His thesis was about similar to the stuff I'd done on a Nordsieck method and stuff that was coming out in my book, and so I was asked to be on his thesis committee. I wish they hadn't invited me to Edmonton in February—it was cold. But it was a good piece of work, and I had some ample research money at that time, so I brought him in as a post-doc visitor for a year, and then the department hired him. He's done some very nice work.

We had Dan Watanabe, was at Illinois then. He'd originally been hired by CSL and moved to computer science, and he was part of that group. Although it's a Japanese name, he was born and grew up in Hawaii. In fact, it's interesting. He went to Japan on a sabbatical and everybody assumed that he could speak Japanese because he had a Japanese name, but he didn't. You know, so Stockholm was probably the number one center for ODEs at that time with Dahlquist and the rest of the people around, and I would say that Urbana was a very close second; there were some other very good places too around the world. Actually, I guess Petzold would have just left by 1979, but I think she finished in '78, if I recall. So there have been some people around who are making some waves.

HAIGH: Okay, so that's those conferences that you have organized yourself. So your impression is that, other than those meetings that it sponsored, that SIGNUM had mainly consisted of a newsletter and perhaps some sessions at the ACM annual meeting and people just getting together and talking at the annual meeting?

GEAR: That's correct. We tried to put some sessions, I remember, from time to time in the annual meeting, but rapidly became less and less interested in ACM and that end of the computational route. Although it had TOMS which was, of course, John Rice's work.

HAIGH: And then, presumably coming from your work with SIGNUM, you served on the ACM SIG/SIC board.

GEAR: Yes. I would go to many of the ACM meetings. I was asked to serve on that, and that was quite a position, and I was there for quite a while. I don't remember how long. And I think it was important. I was trying to get things set up. At that time, I still felt that it would be nice if we could make computational science (though we still probably called it numerical computing) an important part of ACM. It was a lost cause.

HAIGH: So how did the SIG board work? I think the members were coming from all the different SIGs, right? What kind of issues did it deal with?

GEAR: Well, ACM was a rather formal organization. There was a chair of the SIG board who served on the council, and then there were the members of the SIG board who were appointed. Now, I don't think there was necessarily a representative from every SIG, because that would have been too big and it wasn't that big. I think it was people who were prepared to spend some time in doing stuff. Basically, the SIGs had to ask approval, for example, to sponsor a meeting. Now, there was a board, there were also meetings chair and so on, so they probably would process that. But you know how societies are—they get all these rules and then groups that have to agree to let things happen and so on. A lot of it was Mickey Mouse stuff, I imagine, but there were some important issues. I don't remember them much anymore.

HAIGH: All right, so it was there to oversee the SIGs?

GEAR: Right.

HAIGH: And then, so you served on that 1974 to '76, and then '78 to '80?

GEAR: Yes. The gap, though, was I went on sabbatical. In 1976 and '78 I was away quite a bit also.

HAIGH: Then apparently you were also an ACM council member 1975 to '78.

GEAR: Well, that was interesting. ACM, like a lot of societies, has a rule that there have to be two candidates for every office up for election, and the council were elected. Well, that's not quite true. I don't know what the current rules are, but it used to be the people like the SIG chair were an automatic member of council and so on. The treasurer was probably appointed. But then a lot of the seats were regional representatives, and Illinois was in the north-central region. There was a guy that was already a council member who was running again out of Chicago. I forget, he worked for a company of some sort. And they needed another person to run, so I guess my name was around because I'd been doing various ACM stuff, and I was called up, and would I run? You know, "Sure." It wasn't something I knew anything about. I mean, I've been on nomination committees, and the trouble of getting people to fill out your ballot.

Well, academics are always an advantage, especially when you've written a book and your name is on it. My name is probably much better known than this other guy's name. He may have been a much better person to be on the board since he'd been active and that sort of stuff. But I won, so I had to start finding out a lot more about the ACM council. It was kind of way too ponderous,

and I think too self-important, and I can't say I was totally happy with it. There was a period there where there were some rather controversial figures around.

HAIGH: Oh, you'd have been there during the period Herb Grosch was president.

GEAR: Exactly. It was an interesting experience. I'm not saying it was valuable to me. Anything other than dinner time stories. Then, Jean Sammet came before Herb, right?

HAIGH: Yes, 1974 to '76 for Sammet, and then 1976 to '78 for Herb Grosch.

GEAR: Yes. And then was Peter Deming next?

HAIGH: Dan McCracken.

GEAR: Oh, Dan McCracken, right. Well, it was an interesting time. Deming was involved in some level over there, because I remember some long exchanges between Deming and Sammet. Well, it was an interesting time, but it left me feeling that ACM wasn't going to provide the sorts of things we needed in our community.

HAIGH: Now, I know some of that was merely attributable to Grosch's personality. My impression is that during that period there was also a more philosophical kind of disagreement about whether ACM should be a really large organization to try and represent everyone who's working in any kind of computing, including data processing, or whether it should be more of an academic-oriented kind of group for scientists.

GEAR: Well, that was certainly one of my concerns, that it didn't represent the academic end, though I would say the predominant view was that it represented data processors. The battles between people like Sammet and Grosch were not, I think, over what the focus of the organization was; it was more of a question of style. And Sammet was at the extreme of looking at every detail. I don't know how she found the time in her life to go over all the stuff she did. And Grosch was the extreme opposite type who probably didn't give a damn about anything that had been written down on paper and just liked broad ideas. The big clashes, I think, came there. I mean, I think Sammet thought that Grosch would destroy ACM because he wasn't worried about the details. But it takes a happy medium between those two.

There was a definite chance that ACM was not serving the academic community particularly. There was certainly a lot of screaming that the *Communications* isn't serving the community, and the underlying thing was that there were too many of these academic papers that are no use to anybody.

So yes, there was some of that going on, but I would say the battle had already been lost.

HAIGH: And the academic people had lost, in your view?

GEAR: Well, yes. I mean, ACM was not an academic society in that sense. In fact, I think we're going to come to it later. That's really the reason for the Computing Research Association was founded.

HAIGH: That side will be interesting to hear. Now, in these clashes on the council, did you play any particularly active role, or were you more of an observer between warring camps?

GEAR: For the first year or two, I started learning my way as to what all the politics were. I would say I was perhaps a little more of an observer. I didn't hold any particularly strong views. I would like to have seen it provide a bit more service to the academic community, and I felt there were a lot of pressures. When people said that we needed more articles in *CACM* that serve

the ordinary data processors, they might not realize there aren't such articles, unless you're prepared to pay enormous amounts of money to very qualified science writers. Academics can't write those articles; they don't understand the community over there. The people who are in that community typically can't write them; otherwise they probably wouldn't be in that community, they'd be doing something different. So it's wonderful to say that's what we need, but if you only published that, you wouldn't have a journal to put out, and the only way you get good stuff like that is to pay professional writers.

HAIGH: Just finishing up with ACM, then, were you elected to council just once?

GEAR: Twice, for two-year terms.

HAIGH: All right, so you spent four years on that?

GEAR: Yes. Actually, there's a final small story there. I'm no longer a member of ACM now.

HAIGH: So you can speak freely! [Laughs]

GEAR: No, I don't read this stuff anymore, and I lack space now, with my office at home, to store many journals. I don't need anymore paper. If I go to meetings, which in many cases that would be on my own money, those are not ones I'm going to go to, so I felt there was no point. But about a year after I retired, I still remember, I got a call, left a message for me, from somebody I knew from ACM days, saying they wanted to talk to me about serving on council. Well, why would they want me to serve on the council again? So I got back to them, and it wasn't about serving on council. They wanted to know if I'd run for president. I said, "*What*?" You know, I haven't had anything to do with ACM for years, other than just pay my membership fee. Well, anyway, I looked at what the situation was and it immediately became clear. I've served on nominating committees for other candidates. The candidate they already had running for council, was current vice president running for president. Maria Klawe. She's done a good job. She was clearly the candidate; nobody else wanted to run against her.

ACM, by the way, had quite a large number of women members because there's a fair number in that profession. Plus I think people generally agreed that women had been not well treated historically. Dan McCracken, I remember, phrased it very nicely once. He said, "All the people who are smiling in the pictures in CACM won the election." Well, you know, that meant all the women. When you take a picture of somebody, women smile and men don't particularly. So clearly she [Maria] was the right candidate. She was very active, and obviously, people would have voted for her.

Yes, so why should I run? I knew the problem the guy was facing—he wasn't able to get anybody who was perhaps interested in running against Klawe. It didn't mean a thing to me. I was retired. I didn't care if I lost that election. I said, "Okay, you need a fall guy, I'll do it." Well, the next thing that happens is they need a statement they publish in CACM, a statement by each of the candidates. It was obvious with Maria, she could talk about all the programs she's worked on and what she's been doing. I didn't know what ACM was up to anymore. So I have to come up with some statement. Now, I could say, "Please don't elect me," but that seems to be a bit…it wouldn't quite do. So I thought for a while about what are some of the issues, and basically my statement, in a nutshell, said, "Computer Scientists are not appreciated." And I think it is a problem -- that a lot of the community outside does not understand what our profession does, and thinks it does something else, and I think there is a significant education process that needs to be done as to what the community, what the profession, is providing to the world. It's not what most people think it is. So I built my statement about, "We're not appreciated." Maria is now Dean of

Princeton, so I see her occasionally. She told me that I put quite a scare into ACM, because that statement really appealed to people and I almost won! I'm glad I didn't, because I didn't really want to spend two years doing that rather than what I'm doing now.

HAIGH: You should just have said you were going to return numerical analysis to the core of the society!

GEAR: Oh yes, that would have gotten me out, right! [Laughter]

HAIGH: So you were on the ballot as the losing candidate. I didn't know that. Now, through the time you were involved with ACM, did you form any opinion of the general administration and headquarters staff of the society?

GEAR: Actually, I worked with some people who worked very well there, I thought. I interacted a lot with a guy named Fred Aronson (I don't know if he's still there or not), who I think was in charge of a lot of the SIG stuff, so I tend to interact with him a lot. There was the guy, what was his name? It's 25 years ago; I don't remember some of the names anymore. But generally speaking, I thought they had a pretty good staff.

[Tape 5, Side B]

HAIGH: Then from 1989 to 1994, you served on the Turing Awards Committee?

GEAR: Well, yes, it was an ACM committee. It has, I think, five members, each of whom serve five years overlapped, in and the last year you chair it. It's one of the many things I feel that this is a duty to the profession. You know, any committee like that, what people try and do, probably, is to have a representative selection so that the five people represent different areas, and I was representing the scientific computing community, I suppose. While I was on there was when Vel Kahan got the Award, for example.

HAIGH: So do you have anything to say about that process?

GEAR: Well, I've been on other types of committees like this. I didn't think it was significantly different. All these things have a certain amount of politics, and everybody wants to push their own area or other areas. You can't get around that. But I think there's a sense, "Okay, there was a theoretical computer science area last year, so we should consider some other area this year." There's that view. So it's not some absolute measure, okay, here are the top 15 people in the world at the moment and this is the rank, and then the first three already have the Turing Award, so we take number four. You can't make comparisons between people that way. Theoretical computer scientists try and make comparisons. You know, there was a story about a guy at the University of Chicago, carried around a little list in his pocket of the top 10 theoretical computer scientists in ranked order. But in most areas, you can't do that. People don't agree on what's important in the software and artificial intelligence so much.

HAIGH: So my impression is that nominations come forth from the SIGs. Is that correct?

GEAR: No, they come forward from anybody who wants to put them in. People at Berkeley put forward obviously the nomination for Vel Kahan. I remember a drawer full of folders from various places, and they can come in from all over the world. Robin Milner from Scotland, was a nomination while I was on that committee. I can't remember any others right now.

So anyway, it was not an onerous committee. It was all done by email.

HAIGH: I saw some papers from one year, and at that time they were using a system of multiple rounds of voting, and you had a certain number of points you could allocate between different candidates.

GEAR: I think it was up to the committee each year. I certainly remember doing things like that certainly the year I was chairing. By then email was everywhere, so I think we would start off by saying, "Okay, everybody gets so many points, and you can allocate it anyway you'd like." And then you look at the thing and throw away the bottom few and then continue. But the problem is that there are all these results on voting systems—it's safe to say no matter what you do, there's a problem. There's a danger that some people would start gaming the system early on, so if one person decides to really allocate the votes the way they feel, "Well, this guy's a little better than this one," blah, blah, blah, and so they spread their votes somewhat uniformly. I think we had a lot of points, 50 or 100 or something to allocate over five or six people. I don't remember. It might be a problem if somebody said, "Okay, by God, I'm going to try to get this person in, so I'm going to just vote for this person only the whole time and to hell with the rest." So it's always a problem. What you're really trying to do is, of course, not have a voting system which people can game, but to try and get the consensus, and I think the voting system is only a way to try and flush out how people feel initially.

HAIGH: Would you say that the committees were successful in reaching a consensus?

GEAR: I don't remember anything really contentious there. The Vel Kahan Award was a little contentious later on. I remember getting some letters. There were letters to CACM, I think, complaining about it. But, you know, Vel Kahan is a very demanding person intellectually. He's very good. I have high respect for him. But people like that occasionally offend some people. I seem to recall there was a certain letter in CACM that was something like, "How can you give the award to somebody who treated students so bad like that?"

But Vel Kahan didn't have much time for stupidity. Very early on in the 1960s, I was going to some meetings, and we were going to one meeting in particular on partial differential equations at the University of Maryland. I think it was NSF [funded]. Well, whoever the funding was, we had to write trip reports if we had government support to go to a meeting, and at that meeting, Vel Kahan had really demolished a speaker who was saying nonsense. And I wrote in my trip report, "NFS's best investment would be to send Vel Kahan to as many meetings as possible, and it would stop these speakers saying nonsense."

HAIGH: And more generally, not necessarily during the time that you were on the committee, but through the present day, are there any outstanding people you think should have won the Turing Award and haven't?

GEAR: You know, in the last 10 years, I really haven't even been following who's winning, so I don't know. I've lost interest in ACM because of this area.

HAIGH: Okay. And I know as well that ACM's put a lot of effort into marketing and branding the award. Now whenever it's referred to, they like to include the phrase "the Nobel Prize of Computer Science".

GEAR: They need to up the size of the award, then!

HAIGH: They got some money from Intel, so I think it's a little more generous than it used to be.

GEAR: Last time I saw, it was $25,000.

HAIGH: A hundred thousand now, I think. But did you have an impression any time you were on the committee of how well the news got out and how much prestige was accruing to the thing?

GEAR: Well, I think it was of immense prestige within the community. I don't think it meant a thing outside. All sorts of prizes are billed as "the Nobel Prize of…" and it doesn't usually have much impact at all. But there's one person whose name I won't reveal, on his resume he got the X Award, and then it says, "The Nobel Prize of…" He says wishfully. [Chuckles] You can't control the public imagination for things, and the public imagination for the Nobel Prize is it's really done something. In some cases, it's kind of awful, because you see some Nobel Prize winners, once they've won the Nobel Prize, seem to assume that they're experts and that type of thing.

HAIGH: Hence they start writing letters to the editor and setting up genius sperm banks and that kind of thing. I interviewed Charles W. Bachman, who won the Turing Award for database management systems, quite early in the award's history. After I asked him about it, his wife dusted off the bowl and served us cashews out of it. She was pleased to have found a use for it at last.

Now, returning to other kinds of professional activity, that wraps it up for ACM. You've talked about those two meetings you organized.

Talking to some of the other interviewees, many people have mentioned the early 1970s as an important time where a community of mathematical software people started coming together, and one of the things that they've talked about are the mathematical software conferences organized by John Rice. I know that you were presented unpublished papers in the proceedings of the first and third meetings. So I was wondering, do you have any general impression of those meetings, and do you remember making any contacts there or having the idea that that was a place where a community was forming?

GEAR: Well, what I remember about those meetings was being very impressed and seeing that this was going to be an important step forward. At one of those two meetings—I forget which one—John had his little organizational meeting for TOMS. I remember sitting at that. I believe I may have been an Associate Editor, initially. I don't have everything in my résumé, and I don't remember. But I certainly was involved in those meetings. He ran that for a long time. Maybe he still runs it; I don't know, I haven't looked at it recently. So he made a big contribution by doing that, and those meetings were, in some sense, a kickoff. He produced TOMS out of it, because that would require the approval of the ACM process, which has to be done very carefully because of the big financial commitment. I don't know whether I went to the second meeting and didn't give a talk or whether I didn't go. Maybe I wasn't around. I don't remember. The first meeting was, I think, was it '71?

HAIGH: Let me check the dates. If I remember correctly, the second meeting did not produce a book because the papers from it were used as the seeds of the first volume of TOMS, if I'm remembering that correctly. So this will be listed as a paper in the proceedings, or is it a chapter in your book?

GEAR: I don't recall. I'm not certain I have a paper in TOMS, so maybe I didn't even go; I don't remember. The first meeting must have been in 1970 or something, because I seem to recall flying there from Stanford when I was on sabbatical, but maybe I'm wrong.

Anyhow, back to the substance of the meetings. There was some good talks at those meetings; there were some terrible talks. I seem to recall one of them I was sitting next to Lloyd Fosdick, who's been a friend for a long time. He was at Illinois, then went to Colorado. And somebody gave a talk where he said, "Well, I just sketched these ideas out on the plane," and I think Lloyd whispered to me, "…and it looks like it." It was just an awful talk. But I remember another talk there, and I can't remember exactly who it was that gave it. I think it was a guy from IMSL, but I'm not positive. And I keep thinking back, when I talked about the issue of low-accuracy methods, how do you measure error, and he talked about you're computing something, and he drew a fast curve, and then he would look and say, "That's about right. It's got the right number of rises in it and it's got about the right shape, and that's pretty good." And time and time again, I recall that. The issue is, how do you characterize that statement? What measure is it that you can put to two curves that captures that, that says, "Yes, these are close in this sense," that an engineer or a scientist will say, "Yes, that's a good approximation," when if you applied any standard L2 measure, the difference is terrible. I'm struggling with that problem right now. I'm doing some consulting for some government stuff, and it's exactly that problem still. Is this a good agreement between computation and experiment or not? Because if you applied any standard mathematical measures, they're not the least bit alike, the L2 measure or any other standard norm, it's enormously different. Enormous.

HAIGH: I found the publication date for the book from the first conference; it's 1971.

GEAR: Okay, so I think it took place in '70, even the fall of '69. I remember flying back there and it was in the winter, because I flew to Chicago and drove down, Lafayette, being not the easiest place to get to.

HAIGH: Prior to that meeting, had you already known most of the people who were working on mathematical software? Or was that a place where you met people and were exposed to new ideas for the first time?

GEAR: I think I knew most of the people at the time, because I think what Rice did was to create the idea of mathematical software as a piece…. I mean, basically, the numerical analysts who wrote programs suddenly become people doing mathematical software—it was not another group of people.

HAIGH: So it was a new subset of a group of people, many of who knew each other already, and a new identity.

GEAR: Yes. Actually, I think it was that first meeting-- it's the first time I had a close interaction with Dahlquist, I believe. I met Dahlquist the first time, the time I gave my first talk on stiff equations in 1968. [The Edinburgh IFIPs meeting in 1968.] He gave a paper there too, so I certainly met him then. But he was at that Purdue meeting. Purdue has like a hotel in the Union or something, a house there. I remember us sitting in my room along with somebody else—I don't remember who it was—talking about ODEs. Now, I was still somewhat junior then. I guess I had only just been promoted to full professor the year earlier, and I remember I kept thinking, "Wow, I'm sitting in the room with Dahlquist, this big man in the field, and he seems to be interested in what I have to say about that. I'm not just listening to what he has to say." So that meeting obviously attracted quite a lot of important people.

HAIGH: Yes. Then after the third conference, my understanding is that some of the same dynamic shifted to the newly created IFIP 2.5 working group.

GEAR: Well, I think the function of it was a little different. I was a member of that for quite a number of years, and we were organizing various meetings, discussing issues of mathematical software. I'm not certain how much it really accomplished other than to be an interaction point for people from different nations, and the whole IFIP process is a little cumbersome, I think, about setting up and approving things. You know, it was a subgroup of TC2, I think. But, you know, we put together a few meetings and produced some proceedings, and it was interesting.

Actually, I think perhaps the most important thing that happened in IFIP 2.5, and I don't how many people agree with it, we had a technical meeting (I think in Switzerland) then we had a meeting of the working group afterwards. And Vel Kahan at that time was in the process of pushing its floating-point standard. It was getting a lot of opposition from the entrenched manufacturers like DEC who didn't want to change their system. I think he was probably most fully supported by Intel. I think he was a consultant to them, or maybe it was HP. I don't remember exactly.

HAIGH: He consulted for both, but by the time the standard was being set, he'd already designed the 8087 floating-point unit for Intel. So he had the ideas there and had proved to himself that they would work. He couldn't share all the details on the implementation to explain why he knew it would work.

GEAR: Well, there was a lot of opposition from some of the other computer manufacturers because they saw it was expensive and they thought useless, and he made a special trip to come and talk to us in the working group, basically to get our backing. And I must say, prior to that, without having seen any details, I felt somewhat negative. You know, pretty expensive. Why all this fuss? And he made a presentation that convinced me. You know, he's done a lot of good stuff. He hasn't published enough, but he's done some important stuff. But that, to me, is the high point of what he's done, his greatest crowning glory. That would be a contribution to almost anybody in the field, I believe—an enormous impact on the world. How many people use a floating-point chip every day without even thinking about it? And now they have far fewer problems to deal with because of it. Even when they're moving their spreadsheet from one computer to another, it still gives the same answers.

HAIGH: So other than that work with floating point, your sense is that the IFIP 2.5 working group was a pleasant thing to be on, but didn't achieve an enormous amount?

GEAR: I don't think it did a whole lot. It had people like Brian Ford on it, who was running NAG. It had some commercial interest. Other than that, it was mostly a bunch of academics.

HAIGH: Do you think that's a good thing or a bad thing?

GEAR: Well, I mean, if you don't have the library people agreeing with what you're going to do? They're the ones that are going to implement it, so it's better to have them involved. I don't recall if there were any hardware people, computer manufacturers on it. I don't remember any; maybe there were some. Of course, court cases long ago had made IBM unbundle software, so I think most computer manufacturers probably didn't think mathematical software was of any concern to them whatsoever. It wasn't for them probably a very big segment of their market. Oh, I'm sure it did something useful. I think it was mainly an opportunity to interact with people.

HAIGH: And you mentioned to me without the tape running that you hadn't been to any of the Gatlinburg meetings.

GEAR: That's correct. They were numerical linear algebra.

HAIGH: Right, so that didn't overlap enough with your differential equations?

GEAR: Right.

HAIGH: Okay. So we should talk about SIAM. I guess, while we're talking on this topic of mathematical software. Let me just ask about your feelings of how the field developed. So it seems there were at least two major transitions: the first one occurring during the 1970s towards the large libraries, particularly NAG and IMSL; and then the second one in the late 1980s, away from libraries that you would call from FORTRAN programs and toward self-contained systems like MATLAB and Mathematica. So do you have any thoughts on how the mathematical software field as a whole has developed over the last 30 years?

GEAR: Well, I think it's just an evolutionary rather than revolutionary process. Libraries are very important. In the very early days of computers, the whole idea was first broached in the book by Wilkes, Wheeler, and Gill, that was the origin of it. [M.V. Wilkes, D.J. Wheeler, and S. Gill, *The Preparation of Programs for an Electronic Digital Computer.* Reading, Mass.: Addison-Wesley, 1951]. You know, I mentioned yesterday, ILLIAC I had a very valuable library from all its users, and that was a big part of its value.

Then there was, for example, IBM SHARE group, which was the group of IBM users that got together and shared software. So IBM put out some minimal libraries initially, but SHARE provided a lot more with software. There were a lot of quality control issues, and then when things got beyond IBM, there were unbelievable compatibility issues between different FORTRAN compilers. So that was one of the issues that was being pushed earlier on, too. In the early days of mathematical software, there weren't standard constants defined. There were names of things, like what's the minimum round-off, and I can't remember what the title was for it. We don't think about that so much anymore because everybody assumes the IEEE floating-point standard. But early software had to be written so you could write in your program and find out basically what the floating-point arithmetic looked like, and then you could write your code to reflect that. It sounds pretty primitive now, and maybe WG2.5 had something to do with that. I can't remember. I don't remember who developed that.

HAIGH: I know there was a workshop, I think in 1976, at Oak Brook on portability, and really I think in the '70s, with the expansion of platforms they had to start worrying about minicomputers and vector machines, so portability became more important. There was the Bell Labs PORT library and the PFORT verifier, and I think PORT came up with a scheme that was widely adopted for a particular way of enumerating the machine constants.

GEAR: Well, I remember there was a set of standard definitions. There were names that you would use in your FORTRAN program, and the assumption would be that the compiler would provide predefinitions of these on a specific computer. I am not certain whether they came out of WG2.5 or they came out somewhere else. You're right, there was that software meeting in Oak Brook. I remember I went to that. And I'm not certain that we didn't have a WG2.5 meeting at that time, because I vaguely remember one. Oh, then there's one thing WG2.5 got involved in, only in giving, say, the Good Housekeeping seal of approval, and that was the initial proposal for the BLAS was put before the WG2.5 early on. So that was a positive thing, because that's had a big impact, I think. You see it used a lot.

But okay, so we're back to the question, which was in the progression. I'd say these are just evolutionary from early, very machine-specific libraries to an attempt to get libraries that run across a range of machines using machine constants.

HAIGH: One of the interests you've expressed is in how people are actually using these things. I mean, one of the issues perhaps is the kind of mathematical understanding that you get from using the different kinds of approaches, writing your own software or coding a library or just do something in MATLAB and not necessarily knowing what's happening underneath.

GEAR: Well, I think the answer, the more users we have on computers, and the number is growing every day, the less reason there is to expect the average user to be able to understand some of this stuff. Because if someone is an expert in finance then they only have a limited amount of time to try and understand numerical methods. The further you go away, the more it has to be automated. We went from the fairly machine-specific libraries and system-specific libraries that were provided by manufacturers in the early days. Then there's unbundling, so then, naturally, in the American system, people have tried to start to cash in, someone selling compilers and selling libraries. So you had places like IMSL and of course NAG, and you're in the business of trying to sell libraries because there was no market. And you had places like Bell Labs for making stuff available for free. But libraries were limited value because you have to understand very carefully how to use a library. It's not easy to. So the origin of MATLAB, which is one of the first of these, was simply Cleve Moler, it was his little toy, trying to put together the simple ways to call all the stuff that was being done in EISPACK and all that other stuff at Argonne at the time under a big NSF grant. It was a major mathematical software effort to provide reliable codes for these things. That just grew and grew, and I guess Cleve's students made a very successful company out of it, which he eventually joined. I use MATLAB almost entirely for my software programming now, because when I need to solve nonlinear equations or do an optimization, all the stuff is there easily, so it's got the library built in it. And since all the stuff I worked with is vectors and matrices, it provides a very easy interface to do a lot of this stuff. If I was doing a different type of stuff, I might use Mathematica, but MATLAB is the most suitable for me. And I can't afford both of them—they're very expensive.

HAIGH: So let's move on, then, and talk about your involvement with SIAM. When did you first become involved?

GEAR: Well, very early on, I became an Associate Editor. I don't remember whether I have all that stuff down, but I think I was an Associate Editor for a while for *Numerical Analysis*. I'm not actually positive of that. When Gene founded SISSC [SIAM Journal on Scientific and Statistical Computing], I became an Associate Editor for SISSC for a while.

HAIGH: Ah, here we are. Yes, 1968 to 71, *SIAM Journal on Numerical Analysis*.

GEAR: Okay, yes. These things are serving the profession: you start out by refereeing, and if you're one of the referees who actually responds in a reasonable time as opposed to never responding, which a lot of people do, then eventually somebody says, "Oh, I need another Associate Editor in this area, and this guy's being somewhat reliable," and then you get it. So I then became an Associate Editor. I can't remember the exact sequence of events. Well, you know, actually, it's interesting. The SIAM thing was another example. I've always been involved in a number of journals; some of them I'm still involved in. Yes, so I was on *Transactions on Mathematical Software*, but it was much later; 1978. You say yes, and sometimes when you get a bit overloaded and the stuff is not of so much interest, you say, "Okay, maybe I'll drop that one; I don't have time to do it."

HAIGH: So your earliest involvement, then, was with these specific journals in areas that you're working in.

GEAR: And I probably served on some program committees.

HAIGH: Yes, the program committee is 1978 to ’86, so that was selecting papers for the annual meeting was it?

GEAR: Well, I think the program committee more just meets at annual meetings and discusses what the topics of the meetings are going to be, then there’s a committee for a specific meeting that does all the work. I mean, the program committee approves appointing of the people who are going to run a meeting. But I think I was probably involved with at least one meeting, but I can’t remember.

HAIGH: From 1980 to 89 you were on the SIAM Council, so that must have been more of a step towards dealing with the organization as a whole.

GEAR: Yes. SIAM has two bodies: the Board, which deals more with the financial organizational issues, and the Council, which deals maybe a little more with the thematic.

HAIGH: Before that you had been on the ACM Council and you’d also been involved with the ACM SIGNUM.

GEAR: So, wow, I’m wasting all this time.

HAIGH: Well, that’s one thing. But I was also thinking that must have given you an opportunity to compare the styles of the two organizations.

GEAR: Very, very different. But, you know, ACM, at the time I was there, had 60 or 100,000 members; SIAM had six thousand.

HAIGH: Yes, so what did that translate into in terms of the way that they worked?

GEAR: Well, we were running much smaller meetings, much less of a business, and much more of a… well, it was an academic society. I mean, it appealed to industry, too, but it was concerned with intellectual issues. ACM was concerned with much broader issues.

HAIGH: So you’ve mentioned that the ACM Council was very political. Did you have any controversial issues to deal with in the SIAM Council?

GEAR: I don’t recall ever in ACM having the sorts of issues which have people pitted against each other, fighting battles, and in SIAM, it was very collegial. We sometimes had slightly tough issues. I remember once, and I can’t remember what position I held in SIAM at the time, but there was one person who had in previous meetings—at least one prior SIAM national meeting, maybe more—given a submitted paper. And SIAM accepts any paper, basically -- I don’t know if it’s still true. And this person on a faculty had submitted a paper that looked okay, but then got up and proceeded to make a whole bunch of anti-Semitic remarks, which, you know, is a problem for organizers. What do you do? And I remember having a long debate on the SIAM council. I may have even been president at the time; I don’t remember. I may have been. Whatever my role was, I know I was heavily involved in trying to decide what to do (actually this person submitted the paper but did not come to the meeting) but we had to decide what to do. So the first thing we did was to schedule it as the last paper on the last day, because most people have left by then. The discussion was that we didn’t feel we could refuse to accept it, because we weren’t making decisions on refusing acceptance of any other paper, and we really didn’t have a case to reject. The paper, on its surface, was okay. The most annoying thing about it was I stayed around an extra night at the place where we were in order to be at that session to see what happened, and the person didn’t show up. So at the next meeting, we passed a

resolution saying essentially that if people had submitted papers and didn't cancel and didn't show up, we wouldn't accept anything for another two years.

HAIGH: But you didn't pass anything about no anti-Semitic remarks?

GEAR: I think it goes without saying that there shouldn't be anything that's not germane to the thing, and I don't think one should single out—now I'm just talking personally. No, we didn't put them in. I don't think it would have been appropriate to pass a resolution like that. I mean, it may have been a more general…. I guess my feeling is an awful lot of those things I think you expect the community to understand. If you pass a law that says you can't do this, somehow it suggests that anything else is open until you pass a law against it. It's like in sailing, for example (I used to sail and do a little racing, not very successfully), there's a final rule which is essentially against ungentlemanly conduct. You don't expect to have to lay down in minute detail what's allowed and what's not allowed.

HAIGH: And then, you had a term as SIAM President in 1987 and 88. So how did one get to be president? Was it the same deal as ACM with a nominating committee that would pick two people?

GEAR: Right. And, in fact, I had gained the feeling that it might have been kind of like that. Gene Golub had been president prior to me immediately. I got a call when I ran for president. I said, "You know, I don't think it's appropriate to have two people who are essentially very similar, from the SIAM perspective working in essentially the same areas," because SIAM covers more than just numerical analysis, you know, "be president back to back." And they needed a second candidate. The person running just had been at NSF and just gone to Minnesota. He seemed to me like a very appropriate candidate and a strong candidate. I didn't think I could beat him, and it didn't seem to be appropriate for me to be there right after Gene. So, again, I finally agreed to run more or less because they needed somebody to run. I mean I was on the nominating committee at one point and had exactly that same problem. You know, you've got a candidate you really think is ideal and very good, and you don't want to put up [as a second candidate] nonsense, somebody that's clearly not an appropriate candidate. That looks pretty bad and demeaning to people. So you have this problem. So, again, I think it came in, because I probably was a little more academic and different. I won; I didn't expect to.

HAIGH: As president, now, did you have any particular initiatives, or were there any problems that you had to deal with, anything you did that would put your stamp on events?

GEAR: I didn't have a view about major initiatives. I hadn't gone in there with a mission. My believe with professional societies is quite strongly that you have a good, professional staff at headquarters, the Executive Director and so on, and someone who's just in there for a year or two years (the term is two years at SIAM) shouldn't do too much unless there's something very wrong. I mean, it's more there to be supportive of the staff and help them achieve-- and if the council or board says, "We want to do so and so," there's also that. I don't see the president as somebody who comes in with a mission to turn things around. To some extent, I think the problem with ACM, which we referred to earlier, was there were people that had a mission to change things. If you run an organization for a long time, like when I was running NEC, it's okay. You have an objective and you set out to do it. But as president for two years, you're not going to really be able to implement much, so you shouldn't start something that somebody else won't want to continue. So we had a few issues, some of them personnel related, but nothing

very (I mentioned already the issue of the anti-Semitic remarks, and I think it probably didn't happen while I was president) but otherwise I don't think I was at that level of involvement.

There was the big issue that was starting to come up that hadn't really surfaced, which was electronic publication. The Web wasn't quite there yet in terms of general use. And an enormous part of SIAM's budget income came from publications. I think all the professional societies were facing these issues of what to do. But there was a problem about the computer system. I was very unhappy with the computer system that Ed Block had been a big supporter of. I can see how one might make that choice, and I certainly made those same sorts of choices in the past and been wrong. But he'd gotten himself locked into something with a contractor where SIAM had its own system, and it did everything, but you were stuck, there was no place to go. So we had lots of discussions about that.

HAIGH: And how would you describe Ed Block?

GEAR: I liked him. Of course, he was Mr. SIAM. He'd been in at its founding, and always in a case like that, there are some problems in that people really don't want to see things change too much, and so some people were pretty unhappy with the fact that we were… He was pretty resistant from his computer issues; he was very worried about changes in the publication process because SIAM's budget relied on it and so on. And we all, I think, as we get older, get resistant to change. There's some things in my life now I know would be a lot easier if I changed, but for the length of time I have left to live, it may not be worth making the effort to change. I think you start to think in those modes. So, well, now he's retired, and I'm sure Jim [Crowley] has changed a lot of things.

HAIGH: Anything else to say about SIAM?

GEAR: Well, it's still my favorite society. I remember Gene used to make the comment, "It's everybodies' second society," and I think that was a problem for SIAM—lots of people who are members of the SIAM are members of other engineering societies, and SIAM is not their primary society. SIAM has always been my primary society. I like it. And now it's gotten heavily into the numerical stuff, computational stuff. I think it's great.

HAIGH: You were Managing Editor of the *SIAM Journal on Scientific and Statistical Computation* [SISSC, now called the *SIAM Journal on Scientific Computing*, SISC] from 1985 to '89. So is "managing editor" like "Editor in Chief"?

GEAR: Editor in Chief, yes. Now, that was one of the two science journals that Gene Golub founded. SISSC was in some ways a competitor of the *SIAM Journal on Numerical Analysis, which* got more and more theoretical numerical analysis. So this one had a bigger emphasis on computation. In fact, one thing I'll say about that, I did have a slight mission when I took over. I'd been Associate Editor for some time, and one of the things I've always been on about is that referees don't get much recognition for the work they do. Now, when I was Managing Editor, and I think most Managing Editors at SIAM, we encourage them all to offer to write letters to department chairs for everybody who had written referees' reports. So each year I'd say, "You know, if you'd like me to write a letter thanking you for your blah, blah, blah, tell me who to write to," and so on. You know, so that'd be something minimal you'd do. But I proposed the following to deal with that issue and another issue I'll mention, and that is, I wanted on each article we published to say in the footnote on page one, "Refereed by…" Now, this would be recognition for the support they'd done for it. And I wanted it to say who the Associate Editor was. In the SIAM structure, the Managing Editor looks at the incoming papers and then assigns

them to Associate Editors, who then get referees. Depending on the journal, some Associate Editors make decisions themselves. In my case, the Associate would make a recommendation to me and I'd make the final decision as to the publication. So I wanted the Associate Editor name down, too, so they got recognition. But I had another reason for doing it, and that is there were some very sloppy referees, and maybe there were some Associate Editors who didn't always take their time. So if there was a real dud published I wanted people to just think, "Wait, do I want to publish this with my name?" The interesting thing was nobody agreed with me to do it. I couldn't get any support for the idea at all, and I thought it was an outstanding idea. But I gave up. I didn't feel like could insist on it.

[Tape 6, Side A]

HAIGH: All right, so we're running into some time constraints here. You might want to say something about the Computing Research Association that you alluded to earlier.

GEAR: Okay, very quickly. What ACM, because it wasn't very tuned into the academic community, didn't provide was a meeting that could be a recruiting meeting for the CS departments. Many professional scientists, like AMS, and the American Physics Society, have meetings early in the calendar year that serve as recruiting meetings, and ACM didn't have such a meeting. So a group of department heads, including people from Illinois like James Snyder at the time—this is early '70s, I think—started something that became the Computer Science Conference or something like that. It was a meeting run very much along the lines of an AMS meeting: a number of invited talks, and then anybody could submit a paper and give a short talk. And it was a venue for a recruiting. It was simply run by a committee of department chairs from various universities. Over time, that meeting became bigger and bigger, and I think it became an important recruiting venue for the smaller departments; the top departments, the Stanfords and the Illinoises, I don't think ever used it for recruiting, but smaller departments used it for recruiting in computer science. ACM eventually took over the conference. I forget when. In the 1980s, maybe in the '90s, I don't remember. It became a regular conference. But the group that was the board for this meeting became the first self-appointed board, and I'm not certain when they chose the name, "Computing Research Association." Maybe not until later. But eventually, it became a representative board of departments, so it grew up that way. It represented the interests, basically, of computer science departments, and then eventually computer engineering departments, too.

HAIGH: So you were on the Board of Directors from 1988 to 2000.

GEAR: That was an elected position. Initially in 1988 as an academic and then I became an industrial. They tried to have a balance between academic and industrial members. I was also for a while on its predecessor, whose name I can no longer remember, when it was not elected, just appointed.

HAIGH: So how did the association develop, and what areas would you say it was most effective in?

GEAR: Well, it was trying to represent the research community. Of course, it didn't have research publications. It has a newsletter that I would say is nearly as good as *SIAM News*. I mean, it tends to provide information about what's going on in Washington affecting grants, and a few articles of general interest. Lots of ads for positions.

And something that happened very early on, a thing called the Taulbee Survey done by a Professor Oren Taulbee from…I can't remember for sure. It was basically a salary survey and

other stuff of computer science departments. He's certainly retired; he may have already passed on. But the Computing Research Association have took that over, and they still call it the Taulbee Survey, and it runs that annually, and it's a very valuable thing for department chairs to know what are salaries. It gives you, for example, the maximum, medium, and minimum, and all sorts of stuff at different ranks. So the top twelve universities, the next twelve, the next twelve, the other hundred-and-something that respond. And it gets a very high response—I think they get 85%, 90% response rates from that survey, which is pretty unusual. It's very valuable to know what's going on, and it's expanded a lot. It tells you a lot about Ph.D. production, number of students. It's an incredibly valuable data resource to the community, to the government, so I think it's very important.

While I was there on the board in the '90s, while I was at NEC, I tried to start a similar (and I think it's still going on a bit) survey of industrial research labs, the ones that are really into research. That's a little different but more difficult to get at, because in universities there is a clear definition of what is an Assistant Professor and an Associate Professor is and so on. Things are much, much muddier in industry. Pay comes in many forms in industry. I mean, somebody's got big stock options, how do you compare? You're comparing apples and oranges, so it's much more difficult to get information. But we got some, and it was useful. And there's much more sensitivity to release of that. Had a lot of trouble with some companies that they felt like they couldn't release the information.

HAIGH: So you'd say, in terms of formal programs, the salary survey is the thing that stands out?

GEAR: Well, that's been of tremendous value to everybody. They were very active in promoting diversity, particularly of women and minorities. They've been very active in promoting the CRA women's things and doing things to push some of the gender problems, lack of support there. There's a whole bunch of stuff. Recently in SIAM (and I think it's still going on; I haven't followed it much) while I was still there, they were just starting up something called the Federated Research Conferences. You know, there were a number of SIGs and things like that that ran annual meetings of a couple of days or something, and what they've done is they've organized things so that several people who have annual meetings run them more at the same time or back to back in the same place. So there's a number of meetings in a continuous way, you might want to go to more than one, or in some cases they overlap and you may want to mix and match a little bit because of your interests. I think those have been successful, though I haven't followed too closely what's going on. That was one of the things that was happening.

They do a lot of stuff like congressional testimony, so they've been a valuable resource for representing the academic research community, which ACM didn't represent at all. So I think it provides a very valuable service, and it runs on a fairly modest budget. The dues depend on the size of the department or the amount of research money, so that even very small schools can afford it. The lowest-end dues are maybe $500 for a department and the highest may be $5,000, because they try and hit industries up a bit more.

HAIGH: Was there also an informal component just to bring together deans and lab directors and so on?

GEAR: Well, some of the people in the committee were lab directors and heads and so on, and some weren't; some were just faculty. So it wasn't anymore a meeting of people of similar

administrative responsibilities. I think the industrial lab people tended more to be the lab directors, like myself.

HAIGH: Were there any initiatives there that you were particularly involved with promoting or in some ways responsible for?

GEAR: Well, I said the first industrial survey I essentially ran, now, because that was problematic because I couldn't have the data. I couldn't ask AT&T to send me their salary data. So I basically had to form the structure and then hand it all off to a CRA employee to take the data and sanitize it.

HAIGH: You have served on a large number of blue ribbon committees and advisory panels over the past two decades. What attracts you to this kind of work, and what skills or experiences make you such a desirable committee member?

GEAR: Several things encourage me to do these sort of things, and different committees have different amounts of each component.

The first is that I believe one has some obligation to perform this voluntary service to the broader professional community - after all, I expect to make use of similar services from others, whether it is refereeing/editing papers, or providing advice on something I need to know. The second is that from many of these committees one can learn a lot. One can learn from the other members - they often have insights that can change or modify the way you think about things. In committees that review, for example, other departments or laboratories, one learns about the way they are organized and deal with things you may be struggling with. The third is that in some cases one can feel that one is actually helping a group move forward, and that's a good feeling. I recall being on a committee that recommended the merger of two departments that were certainly resistant to it, and in the end I think things improved. That was particularly true for the Army Research Labs (which consumed a lot of time over a seven year period) but which had clearly become a much better place over that time. It might have happened without our advice - but I like to think we helped.

I think the things you look for in a committee member are ones with some breadth of experience as well as some specific knowledge; a willingness to take a broad view, listen to others, and not push for ones own particular viewpoint to the exclusion of all others; and to be responsive/reliable in following through (otherwise the report never gets written).

HAIGH: One of the first of these was the COSERS Numerical Computation Panel around 1980. What was the objective of this panel, and how would you characterize its eventual impact?

GEAR: Since the meetings on that took place over 25 years ago, I don't recall exactly what the motivations were - and I am not certain that reading the introductory material again would tell me because frequently there are a number of hidden agendas in any report like this. Ostensibly I suspect it was "sold" (for funding purposes) as an effort to lay out what the field was so that a broader audience (e.g. congressional staffers and university administrators) would understand what it was. A major motive for such reports is always (ultimately) to make a case for more funding in the field as a whole and a lot of scientific disciplines from time to time try to define "what they are."

Thinking back to that time, I would guess that the primary purpose was to make a case to other scientific communities that computer science really was a discipline worthy of respect because there was certainly the issue that many other scientists viewed CS as "programming." And, since

the scientist had once written a Fortran program, clearly it was a trivial thing that anybody could do. (One reason that this was a serious problem was that it ultimately discouraged much collaboration. In those days if some one in CS was asked to assist with a research project in another discipline, at the end of the day the person might not be considered a collaborator when it came to publication.)

HAIGH: You singled out the NRC Army Research Laboratory Technical Advisory Board, which you served on from 1995 to 2002 and chaired from 2000 to 2002 as a particularly important experience. What was the mission of this body, and what was your personal contribution to its work?

GEAR: A number of organizations, mostly governmental, contract with the NRC to produce reports on various topics. NRC is the operating arm of the National Academies (which were founded to provide the government with scientific and technical advice when requested, but they only get funding for each requested activity). Two organizations, NIST and ARL, have contracts to provide periodic reviews of the quality of their scientific activities. Such reviews can be very helpful to management in their internal assessment process and also in providing guidance for important new directions. By the way, all NRC boards and committees are voluntary work. Ones gets expenses (at government maximums which sometimes leaves one a little out of pocket) and the questionable prestige of having been on one. Probably valuable to a junior academic at promotion times!

This particular board was a generally enjoyable and valuable experience for me for several reasons. The first was that the Board (which, with its six panels was large, nearly 150 people involved) did think that it had a significant impact. Since ARL was an organization responsible for a hunk of tax-payers money (between a half and one billion, a very little bit of it coming from me!) it was a "good feeling." When we started the process, the lab had only recently been created from a bunch of other groups, some of which had been relocated, following the first BRAC (Base Reallocation And Closing). A lot of people left in this process, many who had been moved were unhappy, budgets had been slashed, and the whole thing was a mess - not really of their fault. In fact one had to admire the relatively new Director of the Lab (John Lyons, who had previously run NIST) for contracting for this review process under the circumstances. He had had one in place at NIST (in fact, I currently serve on that one) and I think felt that this process could help him put the pieces back together. I learnt a lot about a large laboratory operation (30 times larger than the one I was running at the time) and saw some neat things. It also helped me formulate some of my own ideas about running an "industrial lab" and the way it might differ from a university.

HAIGH: In 1991 you were elected to the National Academy of Engineering and in 1996 to the AAAS. Have you been active within these organizations?

GEAR: As an aside I should say that AAAS stands for two organizations: (1) American Association for the Advancement of Science (I am a Fellow and currently chair of Section T, Information and Computers.) It is not a particularly important society in the computer science/mathematics field. It publishes SCIENCE which is mostly bio-medical with some physics and chemistry. (2) American Academy of Arts and Sciences (I am a Fellow). This publishes Daedalus and has some meetings but is primarily a "honor."

I am not "active" in AAAS (#2) or NAE, but I have served/am serving on a lot of NRC committees/boards, and they like to have some academy members on each board/panel, so in some sense this is service to NAE.

HAIGH: Well, we should probably about jump back now from that to the related topic which is your period as chair of the Department of Illinois from 1985 to 1990.

GEAR: Well, the big topic there was, we'd been desperate for space, unbelievably short of space, and we got approval for a massive building extension. It increased the size of our space by about 150%, and it was wrapped around the building we had. Well, the problem we had was space. The original building was built in like 1957 or something like that, and had been extended twice, each time doubling it. In fact, the second time it was a little bit more than doubling it. It was a small rectangular building. Another piece the same size had been added on extending it one way, then it went the other way and doubled again. A small stub had been put on where the architectural plans showed for a connection to a 14-story tower for the next extension. Well, by the time we got around to that, the Beckman building had been built north of the area and it was a fancy five-story, $50 million building or something, and they weren't going to allow any building to dominate that; they decided nothing could be more than four stories south of that. So that was out, anyway, and it probably wouldn't have been a good design, because high rises are terrible for interaction between faculty. It's well known, you can go quite a long ways on the same floor, but people tend not to go up and down floors to meet people. The choices were to tear the whole building down and start over, and that was out of the question because we didn't have any other space to move to; or to do what we did, which was to build around three sides of it and up over it. So it was a mess for several years.

Then it was finally completed. I'd expected to move into the new department's head office before I left; in fact, it wasn't ready until two weeks after I left, so I never actually occupied the building myself. But I spent an unbelievable amount of time involved with it. I mean, this is not something you just hand off to an architect. I had an associate head who spent all his time on it, and I was amazed at what you have to do if you want to get something nearly right. In the old building, for example, one was always knocking down stuff to put more cable, because you kept on having new technology to connect offices. So I was determined that we would have cable runs that would be easy to feed into, big enough, so I took the largest size of anything that was going to be used, assuming things didn't get bigger, so that we could clearly get one of the old Ethernet connectors down through a thing and into a wall outlet, and then cable runs above. All these sorts of things to take care [of]. So the contractors had the principle and they put all these cable trays up, and they put them in positions where it's easy to just loop the wire up there and you don't have to keep threading it through places. But it comes to a major beam, they just left it and then started it up on the other side of the beam with no way through it. So they didn't understand! They didn't understand they need this tray—they can't put it there, they forget it. They don't understand what it's for. That was one we didn't notice until the last minute. Then the sprinkler system was put in after the cable tray, so it was underneath, which meant now we had to feed the cable over the top of this pipe for the sprinkler system.

I didn't realize, because I had an extension done to my house, but you if have built a house or have an extension, code specifies how many electrical outlets you have to have, for example, so they all go in and it's pretty much automatic. If you worry about it, you might say, "I must have one here," otherwise, they have to be every so many feet. Not so in a building like this. I suddenly found myself -- I spent almost a whole weekend with a set of the plans for this

building, and I don't know how many hundred thousand square feet we had, personally marking down the position of every electrical outlet. Because my choice was to try and find somebody else and explain what was needed, but it was just simply easier to sit a whole weekend myself and mark them. Not a very intellectual effort, but if it's not right, it's a disaster.

HAIGH: So that's what you found taking up most of your time?

GEAR: I spent an enormous amount of time on that; more so than trying to supervise students and keep some grant money coming and move forward. And it's a heavy recruiting period, as I mentioned before, which meant that it really cut into my research. I had very little personal research during that time.

HAIGH: You alluded earlier to the NCSA and the CSRD being established on campus during that period. One of the things you had said was that people assumed that they had all this money so you didn't need any. Beyond that, were there close ties between the computer science department and those two groups, or were they pretty much isolated from each other?

GEAR: Well, with CSRD, virtually all the faculty in it, Kuck, Duncan Lawrie, Ahmed Sameh, had appointments in CS, so they were really CS people. The big problems there, and even more of a problem with Beckman [Institute for Advanced Science and Technology] also came up in that time. It had to do with the way Illinois handled its funding, which I think in some ways is a pretty good way. In the university, when you get a research grant, there's overhead on it, on salaries and for certain other costs, and it covers the university's other expenses that aren't billed directly. A lot of universities like to encourage people to get grants, obviously, and the university's willing to put some money up in various ways. And at Illinois, what happened was 30% of the overhead was given back to the department. When I was running the Department of Computer Science, we had a salary budget of, let me say, four million dollars (I forget the exact number; about that). We had an operating budget of something like $200,000, and it covered nothing. We didn't have to pay for the toilet paper, you know, all the janitor and all that stuff, he's taken care of. But anything for a secretarial or anything, so there's almost nothing else. And if your department doesn't get research contracts, you don't have anything. That's why math departments are so poor, in fact. But we got a lot of money from indirect cost recovery, and that provided for a lot of things in the department.

Well, when faculty moved out and formed their own center, they took their grants and their indirect cost with them, but they were still members of the department, still wanting service and still teaching courses, and we were, unfortunately, using some of that money to cover that stuff. When Beckman came along, Beckman gave what actually at the time was the biggest grant to a state university. It was like $50 million, and they founded the Beckman center. And this whole indirect cost thing became an unbelievably contentious issue, and it was unfortunate because Beckman was a wonderful idea but departments were facing economic ruin. If all your people had wonderful marble floors—and it was a privately paid building, so it could be much better than the state building—people wanted to be out there. They took their grants out there, and the department would be left with zero money to cover other costs. So I dealt with a lot of problems like that, problems that were caused by the funding system, which in some ways was a good system, but these exposed some of the weaknesses of the system. So I was dealing with things like that. Plus I was on the internal advisory committee for Beckman, a committee of department heads advising Beckman, helping recruit the director and stuff like that. So those are the main things. Heavy recruiting. I remember one week eating something like twelve meals with recruits.

It didn't help my waistline! One week we had two recruits in on one day and one recruit in on every other day.

HAIGH: And was there a general kind of consensus among the faculty about what areas recruits were needed in?

GEAR: Well, we had enough slots—we had seven open slots one year—that we didn't have this terrible contention, you know, "We've got to have theoretical scientist," "No, we've got to hire a numerical analyst." If we found somebody reasonably good, there was enough money to…

HAIGH: Yes, you could take one of everything.

GEAR: Practically, yes. And we also had a pretty good department. We'd have faculty meetings and discuss it. I was head, so I didn't have to take a faculty consensus, although I believe in things like that. You know, whatever it is you're running, you run with the consent of the people who you are running; otherwise, it isn't going to work. So we'd take votes on some things. I remember I made a case for one guy in an area we were weak in and we needed to hire, we didn't really have anybody, we needed to move into that. Very few people felt supportive of this particular thing, and it turns out they were right. I don't think it would have turned out well, but I felt we had to take a chance. And in the faculty meeting, as I recall, the vote was almost entirely abstentions and there were maybe one or two negatives and I'm not certain there were any positives. But even one of my greatest critics at the time, and it probably was one of the no-ers, was probably saying, "If you think you really need to hire this guy, you're the head," and I think that's the sort of department you want. I was wrong on this one, but I was right in some other ones.

HAIGH: So you had mentioned CSRD and the Beckman Center. Was NCSA associated with the Computer Science Department?

GEAR: There were one or two computer scientists who were members of it, but no. NCSA was more of a service organization. And Larry Smarr was a wonderful operator, was a physicist or astronomer or similar background. But they were getting all this money and selling the idea that people would get this big supercomputer and they were going to solve the problem of cancer.

HAIH: It had been suggested to me that relations with Wolfram and the group making Mathematica might have been a source of contention during this period.

GEAR: Well, relations with Wolfram were interesting. Wolfram had a zero time appointment in our department and in a couple of other departments, I think. You know, he was pushed by, I think, a physicist; I don't remember. Wolfram had had a falling out with Caltech over software, so he came to Illinois, and it gave him a lot of freedom. I think he sort of used Illinois, however. And I'm not going to talk about it. There's a lot of stuff I can't talk about. But he brought in a group of people, and in fact he hired a couple of my grad students, too; one of them still works there now, in his company. One guy he had me hire in the department as a regular hire, a guy from Mathematica, Steve Omohundro, very bright guy, very nice guy. Though totally American, where'd the name come from? In fact, he left there at some point and I hired him again at NEC later on. Very sharp guy, very interesting guy. But Wolfram was pretty single-minded about developing Mathematica; making a lot of money from it was his goal.

HAIGH: So then, after a long time in the department and five years as chair, you finally accepted one of these offers to go elsewhere.

GEAR: Well, several things happened. One is I decided I didn't think I wanted to be chair any longer. I wasn't doing any more research. And I was concerned just stepping down as chair. I knew it would be very difficult not to be chair and watch the new chair do everything wrong but know that you couldn't say anything, because obviously only I knew how to run the department now! I couldn't accept somebody else running it. But I mean I knew that would be personally difficult for me to handle, because somebody else is chair, they're going to run it the way they like, and you may not agree with it. So I was concerned about that.

Then I got a call from a friend. It was Beresford Parlett, I think, called me, and he said, "Would you be interested in…" he mentioned something new starting. He didn't say what company it was, and I assumed it was something in California since he had called me. I think he said, "When you finish the assignment you're in at present, would you be interested in considering…?" So the call would have come in'88. So I said, "Well, maybe." You know, I've got to decide what to do with my life, right? And so then sometime later I got contact from here, and the reason that Beresford called me: the person who was already president of the new NEC Institute was trying to set it up, who had been from Bell Labs, had a close friend who was in Berkeley and knew Beresford.

HAIGH: So who was that person who was setting the lab up?

GEAR: Somebody named Dawon Kahng. So I got a phone call, would I come out to New Jersey to talk about this? New Jersey? You know, we all know about the jokes about New Jersey from the late night shows. And I must say the only reason I said yes (I didn't know much about it at the time) was I had a weekend to kill in Chicago. My wife was working, and she had some meeting in Chicago, so we couldn't come home, and I was flying back to Chicago from some place on Friday and I was going to stay there and wait for her to finish up, and then we're going to drive down on Monday. So I said, "Well, I've got nothing to do this weekend. I was going to just wander around Chicago museums or something. I'll fly out if you can see me on the weekend."

So I came out, not really intending to take it, but just curious, and I got very interested. Nothing was yet in place. They had a temporary office 30 or 40 miles north of here, Somerville, just an office building. The president I mentioned, Dawon Kahng; there was a vice president for physical sciences, Joe Giordmaine, also from Bell Labs, who'd been hired; there was an Executive Vice President who was from Japan; and some office staff. And they had already contracted for a building. They'd been looking around for a while for a building and they'd settled on one just outside of Princeton that was to be occupied some time in 1989. What I was offered was to run the computer science division, which didn't yet exist. And it's not often you get given the chance to start out something from absolute scratch, so it looked pretty good! At the time, you may recall, Japanese economy was booming like crazy; the American economy was in the doldrums. NEC was making money hand over fist, and at that time, NEC was the biggest chip manufacturer in the world. It no longer is. It was a major manufacturer in computers. It was a fairly big manufacturer in home electronics. It had grown enormously and had all this money to spare, and they were funding a basic research lab.

HAIGH: So was the idea that they wanted something like the Xerox PARC Research Center? Is that what they had in mind?

GEAR: Well, I'm not certain they had that as a model, but they wanted to do very fundamental research. So this seemed very attractive, so that was of interest to me. And it resolved this issue

of what do I do with my life? What life is there after being a department head? So I came back very much more interested in the possibility.

It also solved another issue, which was not a big one, though it was in my mind. Illinois is a state funded pension plan and not part of TIA, and there's a defined benefit plan. The state legislatures for years under-funded the pension plan. At that time, the projection was it would be that we'd be bankrupt by about year 2000. They keep pushing it back by refunding it. So I was a little worried about my retirement. Also, as many universities were at that time, were providing early retirement options to try and encourage people to retire early because of budget crunches. Actually, a friend of mine in another department had said to me, "You know, there's a lot of people taking early retirement here; they just haven't told anybody." Normally at age 55 in Illinois you could retire, but with a seriously reduced pension, like I think a 30% reduction. So a plan they'd introduced was that if you paid a fair amount of money, and I think I had to pay about $30 grand into it, you could retire at age 55 without that reduction. Now, it was reduced because the pension was based on the number of years there, so I was there 10 years less. So I thought, "Well, if I left now, I can get some pension." A year or two before, I'd been approached about a job which was attractive, but I said, "I simply can't afford it at this time. I could not possibly expect you to repay me what I will lose in pension benefits by leaving at this point in my career." It's a problem for the non-TIA. You know, TIA credit goes with you. This didn't. It was non-transportable. It would have made an enormous difference to my pension, so it was just out of the question when I was like 53.

So it was an opportunity I could actually move out, and I thought, "Well, if I left and started taking the pension earlier right now, and simply invested it while I work for 10 years, even if the fund goes broke, at least I've got that out of it and put away towards my retirement." So that was an additional incentive. Although my wife was working in Chicago at that time, and I thought she would say, "I don't want to leave my job." So I went back and met her and we drove down the next day, back to Champaign, and I told her about it and what was involved, and she said, "You know, I'm a little fed up. I'm ready for a change." She took what she called a marital fellowship. She's been working on various publications, so she'd been working here and enjoying it, and so she felt she didn't have to go to work. So she decided it was okay, so we came back out for another look.

HAIGH: Why did NEC want to locate its research institute in Princeton rather than Silicon Valley?

GEAR: This was mostly due to the views of the founding President, Dawon Kahng. He felt that Silicon valley was too "commercial" in search of new products, while NJ had the old time core research culture - like Bell Labs where he was from. Certainly we were making more of an emphasis on basic science than the typical Silicon valley lab - including PARC. It may also have been that he didn't want to move to California but we'll never know! Had there not been the high-tech boom of the 90's, it would probably have been a good decision (I not saying it wasn't a good decision, but other events that I'll talk about overran us). We set up a lab where the scientists had a tremendous amount of individual freedom to do what they thought was important. For example, each scientist had a research budget (for lab supplies, summer students ...) that was automatically allocated - no research proposals or approvals in order to get funds. My objective was to get scientists to think about things that would be scientifically important AND ultimately valuable to our sponsor, but let them make the decisions - since they had the detailed knowledge. (That is true "unfettered research.") I didn't want to apply the terrible

pressure to publish that afflicted almost all academic positions - and resulted in a large number of trivial papers in the literature. I believed (and certainly saw evidence of the same) that scientists got the sense of satisfaction and achievement from producing anything that had an impact. We, as a community, were sort of conditioned to view a top paper as the highest form of output, but computer scientists had certainly started to see the widespread use of a piece of software that they had written as a contribution they could be proud of, and those that got patents which had some commercial potential clearly wanted to see the result appear in products. There were, however, three problems with my "model." The first was scientists were still concerned about keeping their resumes burnished for future jobs, so they stressed publication anyway. The second was the boom of the later 90's. They saw computer scientists from universities becoming millionaires overnight (and, of course, the press doesn't report those that failed) and quickly felt that they should have the same opportunities - have NEC pay for the development of a new technology and then be free to go off and start their own company based on that technology. The alternative approach, that many commercial labs were trying, was to "spin off" start-up companies with investment from the parent company and the "sweat equity" of the principals - the scientists who came up with the ideas, and would take their chances with the start-up. I think that there were a few successes around the country but most failed. (I haven't followed it at all since I retired in 2000, and I suspect that most to the remaining ones bombed in the downturn.) With a company like NEC it was much more difficult to adopt this model, although we tried one start up. First there is just the issue that any large company is not as flexible. If a small company develops a new technology that is not part of its core business, it might make sense to spin it off (or sell it or license it). NEC was involved in practically everything electronic - and it had gotten there by being one of the largest Japanese accumulators of patents. It was, therefore, very reluctant to let go of the rights in any patent it held in case it later blocked an internal development. That is what killed our start up. NEC would not turn over the patent to it, even though NEC owned a part of it. (I am not saying that this was the wrong decision.) The second issue for us was just the enormous cultural differences which meant that the culture of a start up was virtually incomprehensible to Japan at the time.

HAIGH: Had you ever visited Princeton before the offer?

GEAR: I've been here on NSF site visits as part of review groups. I'd been to meetings. That history meeting you looked up was here at the Hyatt, and I'd been to other meetings. Actually, I've visited very close to here. Somebody who at one point was my brother-in-law lives just seven miles over here, and he moved here in 1958; I'd visited him on and off at that time.

HAIGH: So you already had a general sense of the area.

GEAR: I had some sense, but that's changed; it is very different now than it was then. Yes, so I say, it was an opportunity and it turned out that it fit very well. Actually, it wasn't quite time, because this was being done in late 1988; I visited just before Christmas; I made the decision at the very end of the year, and I started as a consultant in January 1989. I didn't move here until May 1990. I became 55 in February of 1990, which was the first time I could leave, but I wanted to finish up the semester and finish up the building.

HAIGH: So you've explained the processes that led you to accept the offer. Now, what do you think were your qualifications or qualities that made them seek you out as a suitable person to build this new center?

GEAR: I don't know. The president was a physicist, had done some very important work, some memory stuff. He didn't have any understanding of computer science. I think they were probably having trouble recruiting anybody, too, so maybe I was the first sucker to come along to say yes. They were doing all the things you do: you call around, you try and find names of people a number of places, and you go and you look up their citation in there, you look at them in the *American Men and Women of Science*, whatever it's called. My citation index looks pretty good because of all those references to the stiff stuff. I'm more wary of those things because I know that there are people who do very good stuff that doesn't get cited so much. So they invited me out. Now, the interesting thing about the job was it wasn't really defined what it was to be at this point.

HAIGH: The job itself or the center as a whole?

GEAR: What was the focus of the computer sciences? The job was pretty well defined, you know, how many slots I was going to have, which reduced with time.

HAIGH: So yes, what you've said so far is that they want to invest some of or all this money they had sloshing around to do really good basic research in computer science.

GEAR: Right. Now, I'll come look back a little bit to what they wanted in a minute, but at that time I was hearing what they wanted from Dawon Kahng, who was the president, and his views. Since he didn't know much about computer science… He had a strong interest in things like intelligence and things like that, which was not my strength at all. But what I felt was important was to be doing things to look for things that might revolutionize aspects of NEC's business, which were computing and communications, not things that were going to incrementally improve it. It wasn't going to be a big lab. The initial projected strength was maybe some 120 scientists. We never reached that level. So there was no point in just developing a team who wrote slightly better parallel software. Lots of people can do that. If there's going to be a fair amount of freedom and a fair amount of money, there's a pretty healthy budget for the rest of it. Let's go after some things that are likely to change the way we do things, because that could be of value to the company and also important to science and society. So I developed a plan. We did have a big effort in learning. NEC was really well known very quickly as having hired some very important people in computers and then physical sciences and several areas in computer sciences, and they got a very good group in learning early on.

HAIGH: Learning as in artificial intelligence?

GEAR: Artificial intelligence, yes. Computational learning theory, artificial intelligence and stuff. My primary effort was hiring very bright people, also people with some vision as to where they want to go. There are many fresh Ph.D., you ask them what they want to do and they say, "Well, in my thesis, I didn't quite do so and so and I'd like to finish that off." That's a big no-no. I would say to a person, "If you were successful to the most you can imagine, what do you think you're going to be doing five or ten years from now?" I want to see if they've got something they're striving for, that they want to achieve. I mean, are they driven to try and move somewhere?

So we set up in our areas. Neural networks was an area that I was not very taken with, but actually the guy we hired there put together CiteSeer and stuff with just a tremendous effect. Some of the people we hired to work with him did that. So I felt we had to have some people in systems. That was actually an interesting point of contention between me and the president, because he felt, "You want people doing programming? Isn't that development?" He had no

conception. He thought computer science ought to be totally theoretical and we didn't need any lab space, and he couldn't see why we needed a network. Well, so he really thought that fuzzy logic was probably the most important thing we should do, and this was driving me crazy. Fuzzy logic is for fuzzy thinkers.

HAIGH: The Japanese like that. We have a Zojirushi rice cooker that claims to do fuzzy logic.

GEAR: That's an advertisement on the front, isn't it? [Laughter] Actually, fuzzy logic is an easy way to define sequences in low-level control systems, and it works very well for those.

HAIGH: So did you have in mind any institution or set of institutions as a model for what you wanted to accomplish?

GEAR: Well, one thought about the Xerox PARC (sort of failure to Xerox, they never got anything in production out of it) as somewhat of a different model. They were oriented around building things, and we had a structure that had a bunch of independent scientists. The president had some very strong ideas, which I disagreed with. He did not like postdocs. I think he'd been a postdoc. Postdocs in physics, in physics and in biology, are really treated very badly in many cases. So he wasn't going to let us have people like that. And he wouldn't even let people have groups; he wanted a bunch of individual workers, and that restricted some of the things one could do.

HAIGH: So how would that level compare with Bell Labs at its peak, or IBM Research?

GEAR: He had been from Bell Labs for a long time, so it was more like some of the Bell Labs research models, although there were certainly some team efforts there. But we were a bunch of individuals. Actually, I had a lot of difficulties with Dawon Kahng in terms of what we would be doing and how we should operate, and I'd pretty much come to the point after two years that I didn't think that I could operate within the constraints that he wanted to put in terms of what we should do and how we should be organized. I knew his contract was coming up for renewal, and he was approaching an age where I didn't know what NEC Japan was going to do. I pretty much decided if he continued I probably would look for another job, because I didn't think I could achieve anything. Then, regrettably, he died suddenly. Of course, then there was a big question for everybody for a long time, who is going to become president. They didn't tell anybody, and it wouldn't become public for almost six months. I think there is something in the Japanese culture that you don't do anything for six months or something like that, I don't know. But after a couple of months, I was asked privately, would I consider being president, and I pondered it, and I didn't know if I really wanted to get even further removed from the research function, but I decided it was probably better to do that than have somebody else come in who was a physicist who didn't understand at all what I wanted to do.

HAIGH: So were there basically two divisions, physical science and computer science?

GEAR: Two divisions, computer science and physical sciences. So I became president in November 1, '92. Then sometime about 1998 I found out they're going to demand that I retire at age 65, which was a little of a shock, but in retrospect I'm very grateful, because I've had a wonderful time since.

HAIGH: So then you were president for eight years. Did you try and change anything fundamental in the objectives or organization of the center?

GEAR: I made changes slowly. As I say, we were a bunch of individuals. There'd been a structure set up so it had something called a board of most senior titled people there, the fellows.

Which is odd, because a fellow in the university here would be a much lower rank. There are all sorts of operating rules requiring board of fellows' approval on hiring. The structures were too formal, and my attitude toward all those things is one should have a board of fellows and one should go and consult with them on certainly everything and we should try and achieve it by consensus, not by trying to win a vote, and we had some people on the board who set up various structures for exactly how many votes we needed for each thing, and that, in my mind, that's not a way to work. But most of the time I managed to get, relatively speaking, consensus with them, or they would even say, "Well, we don't agree with you, but okay." I felt like I was able to do that very well.

HAIGH: So what was the peak in terms of the number of scientists?

GEAR: The peak number, probably, a total head count was probably about 130. Remember, it was total company, so we had a couple of lawyers, we had four people in finance, we had a couple of HR, a safety person, some technicians. There were some temporary students, summer students and part time people, but regular people I think was about 120. So there were probably no more than 50-something scientists equivalent to faculty positions.

I'll tell you one thing that caused some consternation. Towards the end, because the Japanese economy started to go sour, so our budget got tighter and tighter, so one year, towards the end, by about say '98, I realized that we probably only had money for one recruit, and it was going to be physical sciences or computer science. I could make the decision right there, and the board said, "Well, we should just go out and recruit the best person." This is always a problem, because when you say that, the best person is always in a very classical area where it's very easy to evaluate, and that's not necessarily what we're looking for.

In fact, the computer sciences division I think pretty much decided, "Well, we'll never win this competition." They interviewed some people, some very interesting people, and they even had a meeting which I attended, because the division would meet and discuss the candidates and decide who was the best candidate, and they agreed on somebody. There was a very good physicist coming through we did hire. But late in the process, I said, "Okay, I can squeeze enough money that I'll hire both this physicist that everybody officially agreed, and I'll hire this computer scientist that the computer sciences said was the best person. When I announced that, they were furious, because if had they known there was going to be the possibility of the position, they wouldn't have possibly supported this guy; they'd have supported somebody in their own area. So I think it was probably a good way to act in that case.

HAIGH: So as the money flowed less freely, was there any suggestion that they lab should be reoriented to work more closely with products or with divisions?

GEAR: Let me just make a number of observations about science in general and scientists in general and this lab. In spite of scientists saying things about the need for unfettered research and so on and going back to the Vannevar Bush report, where that report was always taken out of context because Vannevar Bush wrote that report in terms of science serving certain things like defense and other national needs. When scientists have an opportunity to get something out of the door and it to be recognized, they really want it. We had cases of people that would come up with some new materials in physical sciences which had some major application potential, and boy, that scientist wanted to see that stuff out there and being used. Because most of us in these things, to do something or achieve something and be recognized for doing it, and I think we'd be just as happy to be recognized by seeing something we invented in every car in the world or

whatever it is as seeing it in the paper. So I believe that scientists are perfectly happy to try and produce, and the unfettered research comes in letting the scientists determine what's important and the way to get there.

So the motivation that I felt was to get people to think about the right sorts of problems, but not to tell them, "You've got to make this thing." Rather, "Think about what's important and tell us what's important, and then do it." And people are prepared to do that. The big problem we had, partly being very distant from Japan, was trying to get anything into NEC. And it's two problems. It's the big company problem, plus in their case, also the cultural difference problem, the two things. We had a lot of good stuff which we couldn't get into NEC, and neither would they let go of it for it to go outside. And we were trying to patent this material. Well, we always got patents—we got a lot of patents. We were trying to negotiate rights to the patents to other people. Or have NEC negotiate for us; we didn't have the right to do that. And NEC didn't want to let go of it.

We had a great idea and even formed a startup company in an area which could have been very important, and we had patents on the stuff that the startup company could use. NEC would essentially not let the startup company have sufficient rights in the patents for it to be viable, so it fell apart. It's partly a cultural thing. I can see there are other issues. However, there are lots of times when people have things and they want to start doing things, and big companies have a lot of exposure. If you're a little small startup company with nothing and you're trying something and something goes wrong, there's not much you can be sued for. You don't have much. You just disappear. If you're NEC and you've got several trillion dollars floating around in assets, you're a big target for a lawsuit or something, so they've got to be much more careful. So there's a whole bunch of issues that make it much more difficult for a large company. You can't be agile because the company has to worry about it. It's got lots of responsibilities and financial liabilities to worry about, it's got lots of agreements with other companies that limit what it can do—there are a lot of cross-licenses on patents that are automatic within companies, so even if we got a patent on something, it might mean automatically some other company has rights to it through the NEC connection. So there's all sorts of restrictions on being able to move quickly, and that made it very difficult.

And every lab has a big NIH factor, and Japanese labs are no different. One of the things I did earlier on, the previous president was very adamant there wouldn't be Japanese researchers in our lab, because this was an American laboratory. I said, what's going to be important in the long run for the viability of this in NEC is its good relations of people who have been at the technical level, so I started a program to have some younger, good researchers from Japan spend a year with us, and we had quite a flow of them; we had two or three at a time for a while. That, I hoped, would ultimately let…. I mean, people used to complain about Japan stealing our technology. They send people over to the universities and they learned a lot, and they took the information back with them, and that's the way you transfer technologies—in people's heads. And I wanted to get it transferred back to Japan in these people's heads. So that was my long-range goal. Unfortunately, the whole thing fell apart before that paid off, because that's a long-term approach to it. So it's just the usual problems with technology transfer, and it just didn't work very well.

HAIGH: How would you describe relations between the NEC institute and Princeton University?

GEAR: Princeton is an outstanding university, but to some extent, it suffers because of it. Many administrators (not those who are still "operating" academics, but the layers of associates deans

etc who handle outside interactions and other issues) seem to have a world view that nothing of any consequence happens beyond the campus. For example, one such administrator said to me once when discussing interactions something like: "Are you interested in our outstanding students, or are you interested in access to the ideas we generate?" Well, the answer was that I wanted to negotiate some way in which faculty (and graduate students) could collaborate with some of our scientists, but in his view it was inconceivable that a mere industrial company on the outside could be doing anything of interest to a Princeton professor. (We ran into this type of view in the administration even when a department had come to us and wanted a graduate student to be able to collaborate with one of our scientists in an area they didn't have covered and to be able to use some of our equipment. The administration wanted us to pay a lot of money for the privilege of having this student come over.) We had a lot of good interactions between Princeton faculty and our scientists, and there were many joint papers. With a couple of exceptions, however, we had no joint appointments because Princeton was very opposed to anything like that. (As far as I know, they have no adjunct appointments either, so they fail to take advantage of a number of top scientists in the region who could be helping with their graduate programs.) The only two exceptions were both physicists that everybody agreed were top level people and whom Princeton would like to hire, and who would only agree to come to the area with a joint appointment.

HAIGH: Are there any areas where you'd say you achieved, in terms of basic research, these objectives of really probing ahead and finding the next big thing?

GEAR: There's nothing there that is truly remarkable, that is a total turnaround. There are some things like CiteSeer, which is used by a lot of people now. It came out of NEC. People were telling me what a wonderful thing that is for a long time. We had a massive impact on the search market, which now Google does, of course. But I don't know if you remember back in the middle '90s and so on, AltaVista was making statements like, "If AltaVista doesn't find it, it's not there." And we had a paper appear in *Science* by the group that ultimately produced CiteSeer and some other things, and they built a meta search engine which we were using internally. It was fabulous. It basically polled all the available search engines: you put a query in and it modified it so it could accept all available search engines, and then combine the results to give you what you wanted, so you could try and find stuff that was out there. In doing this, they managed to get a lot of statistics which showed that, in fact, AltaVista only covered something like 40% of what was out there. This had a big, big impression.

So there were some important materials in physical sciences. There was a very interesting piece of work where we found that light could go through very small holes. The normal theory of light is when the hole starts to shrink below the wavelength of light, the transmission drops off very rapidly. In fact, they first of all found out that if you made a little plate with a lot of small holes in it with appropriate spacing, though each of the holes was much smaller than the wavelength of light, the amount of light that went through was much higher than predicted by the normal theory. And then they found out that you only need one of the holes. If you patterned the side of the thing with some little dimples in it in the right spacing and they are very small, you also got an unusual amount of transmission, which had some potential implications, and that was very interesting. There were a number of things we did. I just mentioned a couple.

HAIGH: Does the research institute still exist?

GEAR: Well, the guy I had hired as the vice president of the computer science department became president. About two years later they were unhappy and they brought another person in

to…. Well, no, the person that was there was made president and these people were made chief technical advisors or something. The person who was the vice president of physical sciences and the person who was the vice president of computer sciences now had left. Somewhat later, this other guy decided to resign early, although I suspect they pushed him out, too. They combined it with another lab that was here, which was more product development, and it now is much more focused on near-term work and working on specific problems. So it exists in some form, but nothing like the same structure. None of the original scientists are there.

HAIGH: From your experiences at NEC and chairing the Computer Science department at UIUC, what would you say are your strengths and weaknesses as a manager?

GEAR: Like Bush, if I have ever made a mistake, it must have been so minor that it was never noticed. But perhaps there are some personality quirks ... It's certainly easier to talk about weaknesses. I don't think I am good at seeing the potential of some developments right away - for example, prior to the PC first coming out, there had been a lot of talk about the possibility of putting the "power of a 7094 in the palm of your hand." I could not see how that could be of much value. Of course, on the other side, I have sometimes been too optimistic about the potential of other projects and gone on supporting them too long, and that may be coupled to my reluctance not to be discouraging to people. On one hand, that is one of my strengths, I think, but it needs to be combined with a willingness to sit heavily on someone when it needs to be done. Being supportive of people and trying to find the best in them is generally a good approach (I used to tell new assistant professors that now we had made the decision to hire them, part of my job was to help them do the things necessary to get tenure, and that we shouldn't have hired them if we didn't think they were capable of it). I wanted to view the department as collegial where each member would contribute to the whole. However, because of that approach, I had less tolerance for the type of person who (seeing themselves as the best in the universe) is very demanding and wants the most of everything (and often wants to be seen as getting more than others). Of course, if they are far from being the best, it's not a problem about what to do, but if they are very good, it's important to make them content in the department without destroying collegiality. If I did things again, I would probably move a little more to encouraging the best. (It's been my observation over the years that, at a superficial level, departments that are very collegial - where, for example, groups of people across discipline head out to lunch together - are usually not the top departments.)

HAIGH: How have your research interests developed since your return to active research following your retirement from NEC?

GEAR: I had essentially no research time in my ten years at NEC since my job was to keep the funding coming from Japan. After retirement I finally had time to go to some lectures at Princeton. I went to one because Linda Petzold had suggested that I should talk with this person (Yannis Kevrekidis) because we would probably have interests in common. (She had collaborated with him a little.) After the talk he asked me a technical question. My initial reaction to it was that there was ho hope for him being able to do what he wanted to do, and I really went away to "prove" why it wouldn’t work. (It amounted to the issue of explicit methods for stiff systems, something that Dahlquist's results say is impossible.) What I found was that there was a restricted class of problems for which it would be useful. (This is sort of interesting, because in 1966, 25 years earlier I had started with the non-existence of multi-step methods of order greater than two with A-stability - the Dahlquist result - and shown that there was a more restricted definition of stability - stiff stability - for which higher order methods existed. In that

case, the more restricted definition applied to practically all problems of importance, so it was very useful.) Our new result, which was the first of a number of papers I have written with Kevrekidis and his collaborators, only applies to a very restricted class of problems, so it is not yet clear that it will have an impact. However, today the big difference is that virtually all ODE/DAE problems that are large enough to consume significant amounts of computer time are generated automatically by systems such as network analyzers written to deal with specific classes of problems. Some of these classes have the characteristics that may benefit from this new approach, so I am cautiously optimistic that I may yet have another impact on computation - and that is ultimately the joy of doing research. Incidentally, one thing I have noticed now that I am doing research only for the pleasure of doing it and seeing the results is that I really don't care about publication. I publish because my co-authors want to publish, but this tells me that much of the drive to do this comes from the need/desire to get research contracts, promotions, and pay raises. I am not in line for any of them any more, and it's a great sense of relief!

HAIGH: I do want to ask two last short questions that I've asked all the interviewees. So the first one of those would be, looking back over your career as a whole, what do you think your single biggest regret would be, in terms of a decision you've taken or just in terms of something that you wish had worked out and it just didn't?

GEAR: I think just in terms of my personal approach to things, not spending more time to follow things through to the end—when I've got most of the way there, getting bored and moving on. I think I might have had more impact if I'd have pushed harder on stuff and really followed it through all the way to the end. But then I might not have done as many things, so who knows?

HAIGH: And then to reverse that, what single accomplishment over the course of your career would you say you are most proud of?

GEAR: Well, I would say it's one of two things. The differential algebraic equations, that whole thing, I think, which is probably something I'm not really known for, I think has had an incredible impact. That's from the research end. I guess I feel like I've done a pretty good job with students. That's something that I feel personally proud of. I think most of my students probably feel like I've treated them pretty well and been helpful. It's certainly something I've tried to do.

HAIGH: Okay. Well, I shall have to stop that, unfortunately, so thank you very much for taking part and making the time available for the interview.

APPENDIX

C. W. GEAR – brief resume

EDUCATION

Ph.D.	Mathematics	University of Illinois U-C	1960
M.S.	Mathematics	University of Illinois U-C	1957
M.A.	Mathematics	Cambridge	1960
B.A.	Mathematics	Cambridge	1956

EMPLOYMENT

2004- Senior Scientist, Chem Eng., Princeton University (Courtesy poition)
2002-2004 Technical Staff Member, Chem Eng., Princeton University (Part time)
2000- President Emeritus, NEC Research Institute
1992-2000 President, NEC Research Institute, Princeton, NJ
1990-1992 Vice President, Computer Science Research Division
1990- Professor Emeritus, University of Illinois
1985-1990 Head, Department of Computer Science, and
Professor, Computer Science,
Applied Mathematics, and Electrical and Computer Engineering,
University of Illinois at Urbana-Champaign
1969-1990 Professor, Computer Science and Applied Mathematics
1965-1969 Associate Professor, Computer Science and Applied Mathematics
1962-1965 Assistant Professor, Computer Science and Applied Mathematics
1960-1962 Engineer, IBM British Laboratories, Hursley

Other positions

Spring 1978 Visiting Professor, University of Nice, France
Spring 1976 Visiting Professor, Department of CS, Yale University
Spring 1970 Visiting Professor, Department of CS, Stanford University

Awards

2002 Dept. of Army: Outstanding Civilian Service Medal
2001 University of Illinois Alumni Achievement Award
1996 Elected to the American Academy of Arts and Sciences
1991 Elected to the National Academy of Engineering
1987 Honorary Doctorate, Royal Institute of Technology, Stockholm
1980 Halliburton Educational Foundation Award of Excellence
1979 ACM SIGNUM George E. Forsythe Memorial Award
1956 Johnson Foundation Fellow
1956 Fulbright Fellow

PROFESSIONAL SOCIETIES & ACTIVITIES

Memberships: NAE, IEEE (Fellow), Am. Assoc. Adv. Science (Fellow), Am. Acad. Arts & Sciences (Fellow), SIAM (past President).

Offices:
AAAS Chair, Section T, 2005

CICIAM	(Committee for International Congresses on Industrial and Applied Mathematics), Chair, 1991-1995
ICIAM	(International Congress on Industrial and Applied Mathematics): 1988-1991, President
SIAM	(Society for Industrial and Applied Mathematics) President: 1987-1988
SIAM	Vice President for Publications, 1990-1992
SIAM	Council: 1980-1989
SIAM	Program Committee: 1978-1986
ACM	Turing Award Selection Committee: 1989-1994
ACM	Council Member: 1975-1977
ACM	SIG/SIC Board: 1974-1976, 1978-1980
SIGNUM	Special Interest Group on Numerical Analysis (SIGNUM) of ACM: Chairman 1973-1975, Board of Directors: 1967-1970, 1972-1973, 1980-1983

INVITED TALKS

Science and Technology Innovators Seminar, University of Minnesota 2005
2004 MMC Distinguished Lecture, Center for Math. Modeling, Univ. Leicester, England.
Institute for Terascale Simulation Lecture, Livermore Nat. Labs, 2004
"The Culture of Research," keynote talk, Rutgers CAIP Industrial Affiliates Program, 1996
IRI Workshop on Academic-Industrial Relations, Duke University, 1996
Woudschoten Numerical Analysis Conference, Holland, Oct 1991
International Conf. on Parallel Differential Equations, Grado, Italy, Sept 1991
Computational Mathematics Conference, Ehime Univ., Matsuyama, Japan, Sept 1990
First Italian Conf. on Industrial and Appl. Mathematics, Venice, Sept 1989
1989 NATO Advanced Research Workshop, Snowbird, Utah, 1989
Retiring President's Address, SIAM National Meeting, San Diego, CA, 1989
Academy for Astronautics, Beijing, China, August 1988
Shanghai Control Institute, China, August 1988
Wuhan University, China, September 1988
Quing Hua University, Beijing, China, September 1988
Italian CNR, Naples, February 1988
Italian Academy of Science, Symposium on Vector and Parallel Processors for Scientific Computation, Rome, Italy, September 1987
Workshop on Numerical Methods for ODEs, L'Aquuila, Italy, September 1987
International Conference on Linear Algebra and Its Applications, Valencia, Spain, September 1987
Second International Conference on Computational Mathematics, Benin City, Nigeria, January 1986
University of Auckland, New Zealand, Australia, Mathematical Society Summer Institute, January 1985
Institute of Applied Mathematics, Stockholm, Sweden, Dahlquist birthday meeting, January 1985
Wilkinson birthday meeting, Argonne, Illinois, September 1984
International Congress on Computational and Applied Mathematics, University of Leuven, Belgium, July 1984
NATO Advanced Study Institute on Computer-Aided Analysis and Optimization of Mechanical System Dynamics, Iowa City, Iowa, August 1983
International Conference on Stiff Computation, Park City, Utah, April 1982
Chinese Academy of Astronautics, Beijing, and Wuhan University, Hubei, PRC,

Lectures on Numerical Analysis, October 1982
Workshop on Inverse Problems, University of Heidelberg, Germany, August 1982
Interamerican Workshop on Numerical Analysis, Caracas, Venezuela, June 1982
International Conference on Stiff Computation, Utah, April 1982
Conference on Matrix Pencils, Pitea, Sweden, March 1982
Royal Institute of Technology, Stockholm, Sweden, March 1982
Biennial Conference on Numerical Analysis, Dundee, Scotland, June 1981
Oberwolfach Workshop on Stiff Problems, Germany, July 1981
Australian National University and Woolongong University, Australia, Oct 1980
Royal Institute of Technology, Stockholm, Sweden, September 1980
Korean Institute of Technology, Seoul, July 1980
Workshop on Differential Equations, Bielefeld, Germany, April 1980
Siberian Branch of the USSR Academy of Sciences, Irkutsk, Russia, October 1979
SIAM National meeting, Toronto, June 1979
Computer Science Conference of the ACM, February 1979
Institute of Math and Applications, Manchester, England, December 1978

AREAS OF RESEARCH
Multiscale Modeling
Numerical integration of ordinary differential equations
Automatic problem solving systems
Simulation and network analysis
Computer Graphics & Vision

PATENTS
US Patent 3,246,315, ``Read Only Memory," Apr 1966
US Patent 5,579,401, ``Feature Grouping in Moving Bodies in an Image." Nov 1996

SUPERVISION OF STUDENT RESEARCH
M.S. Degrees Supervised: 34

Ph.D. Degrees Supervised: 22

Aslam, S., *Chaotic Waveform Methods,* 1990.

Juang, F-L., *Waveform methods for ordinary differential equations,* 1990.

Keiper, J. B., *Generalized BDF methods applied to Hessenberg form DAEs*, 1989.

Leimkuhler, B., *Approximation methods for the consistent initialization of differential-algebraic equations*, 1988

Shilling, J., *Automated reference librarians for program libraries and their interaction with language based editors,* 1986

Purtilo, J., *A software interconnection technology to support specification of computational environments*, 1986

Chronopoulos, A., *A class of parallel iterative methods implemented on multiprocessors,* 1986

Orailoglu, A., *Software design issues in the implementation of hierarchical, display editors*, 1983

Vu, T.-V., *Numerical methods for smooth solutions of ordinary differential equations,* 1983

Gallivan, K. A., *An algorithm for the detection and integration of highly oscillatory ordinary differential equations using a generalized unified modified divided difference representation,* 1983

Wells, D. R., *Multirate linear multistep methods for the solution of ordinary differential equations,* 1982

Gannon, D. B., *Self-adaptive methods for parabolic partial differential equations*, 1980

Speelpenning, B., *Compiling fast partial derivatives of functions given by algorithms*, 1979

Petzold, L., *An efficient numerical method for highly oscillatory ordinary differential equations,* 1978

Runge, T., *A universal language for continuous network simulation,* 1977

Wilkins, S., *Generation and comparison of equivalent sets in a general purpose simulation and modeling package*, 1975

Chung, W. L., *Interactive display of numerical information,* 1975

Brown, R. L., *Higher derivative methods for ordinary differential equations*, 1974

Larsen, L., *Automatic solutions of partial differential equations*, 1973

Tu, K. W., *Stability and convergence of general multistep methods with variable step size*, 1972

Richardson, F., *Graphical specifications of computation*, 1967

Shao, T., *Numerical solution of plane viscous shock reflections,* 1965

EDITORSHIPS OF JOURNALS

1985-1989	Managing Editor, SIAM J. on Scientific & Stat. Comp.
1983-1990	Transactions of Society for Computer Simulation
1982-present	Journal Computational and Applied Mathematics
1979-1984	SIAM Journal on Scientific and Statistical Computing
1978-1984	ACM Transactions on Mathematical Software
1973-1978	Advisory Board, Computers and Graphics
1968-1971	SIAM Journal on Numerical Analysis
1967-1969	Newsletter of ACM Special Interest Group on Graphics

PROFESSIONAL SERVICE
NRC NIST Assessment Board: Vice Chair, Info Technology Panel, 2003-2005, Chair 2005-

NRC Committee on Army Science and Technology for Homeland Defense 2002-2004

NRC Army Research Laboratory Technical Advisory Board: 1995-2002. Chair 2000-2002

NRC Board of Mathematical Sciences: 1997-2000

NRC Committee on Future Environments for the National Institute of Standards and Technology, 2001

Computing Research Association Board of Directors: 1988 - 2000

Univ. of Illinois Beckman Center Advisory Committee: 1994-

Princeton University CS Department External Advisory Committee: 1996-2002

NSF Advisory Committee for Computer Science Infrastructure: 1989-1992

Board of Governors of the Institute for Mathematics and its Applications: 1987-1989, 2000-2002

Advisory Committee for Mathematics and Computer Science Air Force Office of Scientific Research: 1986-1987

NSF Advisory Committee for Computer Research: 1985-1988

Board of Trustees, Universities Space Research Association: 1983-1989; Vice Chairman, 1984-1985; Chairman, 1985-1986

National Research Council Advisory Panel for the National Bureau of Standards on Scientific Computing: 1981-1983

Numerical Computation Panel for COSERS Report

Institute for Computer Applications in Science and Engineering (ICASE) and Research Institute for Applications of Computer Science (RIACS): 1978-1984, Scientific Council; 1982-1984, Chairman

NON-TRIVIAL UNIVERSITY COMMITTEES
Departmental Advisory Committee: 1978-1979, Chairman

Senate Educational Policy Committee: 1976-1977, Chairman 1983-1984

College of Engineering Executive Committee: 1973-1975

College of Engineering Policy and Development Committee: 1972-1975; 1973-1974, Secretary; 1974-1975, Chairman

INDUSTRIAL RESPONSIBILITIES

1992 – 2000 As President of the NEC Research Institute, a corporation wholly owned by NEC, was responsible for overall scientific direction of the two divisions (PS and CS) and management of the complete operation. This included planning and negotiating the budget with the parent NEC Corporation, and reporting to NEC.

1990-1992 As VP for Computer Science Research (and the first employee of that division) at the NEC Research Institute, responsible for setting up the Computer Science Division from the beginning. This included hiring all staff (approximately 20 PhD plus about 18 support staff) and determining research infrastructure such as computer and network system. Also was responsible for University Relations and Library planning for Institute.

1960-1962
As Engineer for IBM worked on the first IBM microprogrammed computer (not released) and lead design team to produce cost reduced version. Was one of the Hursley representatives to the Fred Brooks/Gene Amdahl design committee for the planning of what became the IBM 360 series.

OTHER

Chairman, 1979 SIGNUM Meeting on Numerical Ordinary Differential Equations, University of Illinois at Urbana-Champaign, 1979

Chairman, SIGNUM Meeting on Software for Partial Differential Equations, Sunnyvale, California, December 1975

Chairman of the 5th Annual Workshop on Microprogramming, University of Illinois at Urbana-Champaign, September 1972

Program chairman for Pertinent Concepts in Computer Graphics, University of Illinois at Urbana-Champaign, April 1969

Organized and chaired Emerging Concepts in Computer Graphics, University of Illinois at Urbana-Champaign, November 1967

Organized and chaired the Numerical Analysis session of the 1967 ACM National Conference, Washington, DC

Publications of C. W. Gear

Papers

Optimization of the Address Field Compilation in the Illiac II Assembler, Comp. Journ., **6**, #4, pp332-335, 1964

Singular Solutions at Boundary Intersections in Two-dimensional Quasi-linear Homogeneous Partial Differential Equations, J. of Math. and Physics, **43**, #2, pp100-110, 1964

High-speed Compilation of Efficient Object Code, CACM, **8**, pp483-488, 1965

Hybrid Methods for Initial Value Problems in Ordinary Differential Equations, SINUM, **2**, #1 pp 69-82, 1965

Numerical Integration of Ordinary Differential Equations, Math Comp, **21**, #98, pp146-156, 1967

A Simple Set of Test Matrices for Eigenvalue Programs, Math Comp, **23**, #105, pp119-123, 1969

An Ill-conditioning Problem with Numerical Integration (with C. A. Calahan), Proc IEEE, **10**, #57, pp1775-1776, 1969

Rational Approximations by Implicit Runge-Kutta Schemes, BIT, **10** #1, pp20-22, 1970

The Automatic Integration of Ordinary Differential Equations, CACM, **14**, #3, pp176-179, 1971

DIFSUB for Solution of Ordinary Differential Equations, CACM, **14** #3, pp185-190, 1971

The Simultaneous Numerical Solution of Differential-Algebraic Equations, IEEE Trans Circuit Theory, **TC-18**, #1, pp89-95, 1971

A Graphical Search for Stiffly Stable Methods (with C. Dill), JACM, **18**, #1, pp75-79, 1971

The Effect of Variable Mesh Size on the Stability of Multistep Methods (with K. W. Tu), SINUM, **11**, #5, pp1025-1043, 1974

Stability and Convergence of Variable Order Multistep Methods (with D. S. Watanabe), SINUM, **11**, #15, pp1044-1058, 1974

The Stability of Numerical Methods for Second-order Ordinary Differential Equations, SINUM, **15**, #1, pp188-197, 1978

A User's View of Solving Stiff Ordinary Differential Equations (with L. F. Shampine, SIAM Review, **21**, #1, pp1-17, 1979

Runge-Kutta Starters for Multistep Methods, ACM Trans. Math. Software, **6**, #3, pp263-279, 1980

Numerical Solution of Ordinary Differential Equations: Is there Anything Left to Do, SIAM Review, **23**, #1, pp10-24, 1981

Iterative Solution of Linear Equations in ODE Codes (with Y. Saad), SISSC, **4**, #4, pp583-601, 1983

Efficient Step Size Control for Output and Discontinuities, Trans. of Society for Computer Simulation, **1**, #1, pp27-31, 1984

The Numerical Solution of Problems which may have High Frequency Components, in Computer and Systems Sciences, Springer-Verlag, pp335-349, 1984

Solving Ordinary Differential Equations with Discontinuities (with O. Osterby), ACM Trans. on Mathematical Software, **10**, #1, pp23-44, 1984

ODE Methods for the Solution of Differential/Algebraic Systems, (with L. R. Petzold), SINUM, **21**, #4, pp716-728, 1984

Multirate Linear Multistep Methods, (with D. R. Wells), BIT, **24**, pp484-502, 1984

Automatic Integration of Euler-Lagrange Equations with Constraints (with G. K. Gupta and B. Leimkuhler), J. Comp. and App. Math, **12-13**, pp77-90, 1985

Error Estimation and Control in Solving Differential-Algebraic Equations (with G. K. Gupta and B. Leimkuhler), Australian Comp. Sci. Comm., **9**, #1, pp221-229, 1985

Computing the Structural Index (with I. Duff, SIAM J on Algebraic and Discrete Methods, **7**, #4, pp594-603, 1986

Maintaining Solution Invariants in the Numerical Solution of ODEs, SISSC, **7**, #3, pp734-743, 1986

Differential-Algebraic Equation Index Transformations, SISSC, **9**, pp39-47, 1988

Numerical Analysis, Encyclopedia Britannica, **15**, 37-48, 1988

On the Efficient Implementation of Preconditioned s-step Conjugate Gradient Methods on Multiprocessors with Memory Hierarchy (with A. Chronopoulos), Parallel Computing, **11**, pp37-53, 1989

S-step Iterative Methods for Symmetric Linear Systems (with A. Chronopoulos), J. Comp & App. Math, **25**, pp153-168, 1989

Parallel Methods for Ordinary Differential Equations, Calcolo: Quarterly on numerical analysis and theory of computation, **25**, #1-2, pp1-20, 1989

Real-Time Integration Formulas with Off-Step Inputs and their Stability (with D. Wang), Applied Mathematics and Computation, **31**, pp132-147, 1989

Treating a Single, Stiff, Second-Order ODE Directly (with Mohammed bin Suleiman), J. Comp. App. Math, **27**, #3, pp332-348, 1989

Differential Algebraic Equations, Indices, and Integral Algebraic Equations, SINUM, **27,** pp1527-1534, #6, 1990

The Analysis of Generalized BDF Methods Applied to Hessenberg Form DAEs (with J. B. Keiper), SINUM, **28**, #3, pp833-858, 1991

Waveform Methods for Space and Time Parallelism, J. Comp. and Appl. Math, **38**, pp137-147, 1991

Approximation Methods for the Consistent Initialization of Differential-Algebraic Equations (with B. J. Leimkuhler and L. R. Petzold), SINUM, **28,** #1, pp205-226", 1991

Does Variable Step Size Ruin a Symplectic Integrator? (with R. D. Skeel), Physica D, **60**, pp311-313, 1992

Invariants and Numerical Methods for ODEs, Physica D, **60**, pp303-310, 1992

Massive Parallelism Across Space in ODEs, J. App Num Math, **11**, pp27-43, 1993

Parallelism Across Time in ODEs, J. App Num Math, **11**, pp45-68, 1993

The Index of General Nonlinear DAEs (with S. L. Campbell), Num. Math., **72**, pp173-196, 1995

Multi-body Grouping from Motion Images, International Journal of Computer Vision, **29 #**2, pp133-150, 1998

Projective Methods for Stiff Differential Equations (with I. G. Kevredidis), *SIAM J. Sci. Comp.* **24**(4) pp1091-1106 (2003); original NEC Technical Report NECI-TR 2001-029, Apr. 2001. http://www.neci.nj.nec.com/homepages/cwg/projective.pdf

"Coarse" Integration/Bifurcation Analysis via Microscopic Simulators: micro-Galerkin Methods, (with I. G. Kevrekidis and C. Theodoropoulos), ", *Comp. Chem. Engng.* 26 941 (2002); original NEC Technical Report NECI TR 2001-106, Oct 2001.
http://www.neci.nj.nec.com/homepages/cwg/UCLA90.pdf

Telescopic Projective Methods for Stiff Differential Equations, (with I. G. Kevrekidis), JCP. **187**, #1, pp 95-109, (2003). See also NEC Research Institute Report 2001-122, http://www.neci.nj.nec.com/homepages/cwg/itproje.pdf

"The gaptooth method in particle simulations" (with J. Li and I. G. Kevrekidis,), *Phys. Lett. A*, **316** pp.190-195 (2003).

"Computing in the Past with Forward Integration" (with I. G. Kevrekidis,) *Physics Letters A*, **321** 335 (2004)

"Deciding the Nature of the Coarse Equation through Microscopic Simulation" (with J. Li, J., P. G. Kevrekidis, and I.G.Kevrekidis), *SIAM MMS* **1**(3) 391 (2003)

"Equation-free coarse-grained multiscale computation: enabling microscopic simulators to perform system-level tasks" (with I. G. Kevrekidis, J. M. Hyman, P. G. Kevrekidis, O. Runborg and K. Theodoropoulos), *Comm. Math. Sciences* **1**(4) 715-762 (2003); original version can be obtained as physics/0209043 at arXiv.org.

"Equation-free modeling of evolving diseases: coarse-grained computations with individual-based models" (with J. Cisternas, S. Levin and I. G. Kevrekidis). *Proc. Roy. Soc. London*, in press, (2004); can be found as nlin.AO/0310011 at arXiv.org

"Coarse projective kMC integration: forward/reverse initial and boundary value problems," (with R. Rico-Martíneza, and I. G. Kevrekidis), *Journal of Computational Physics*, **196**, #2, pp 474-489 (2004)

Equation-Free: The computer-aided analysis of complex multiscale systems, (with I. G. Kevrekidis, and G. Hummer), *AIChE Journal*, in press (2004).

Constraint-defined manifolds: a legacy-code approach to low-dimensional computation, (with I. G. Kevrekidis), in press, (2004); also physics/0312094 at arXiv.org.

Projecting to a Slow Manifold: Singularly Perturbed Systems and Legacy Codes, (with T.J. Kaper,
I.G. Kevrekidis, and A. Zagaris), *SIAM J. Dynamical Systems,* to appear, 2005

Application Of Coarse Integration To Bacterial Chemotaxis. (with S. Setayeshgar, H. G. Othmer, and I. G. Kevrekidis), to appear, SIAM J. Multiscale Modeling

Proceedings

Numerical Integration of Ordinary Differential Equations at a Remote Terminal, Proceedings 21st National ACM Conference, pp-49, 1966

Automatic Integration of Stiff Ordinary Differential Equations, Proc IFIP Congress 1968, pp187-193, North Holland, 1969

A Computer-aided Programming System (with F. K. Richardson), in Emerging Concepts in Computer Graphics, ed. D. Secrest, "Benjamin Books, pp171-188, 1968

The Automatic Integration of Large Systems of Ordinary Differential Equations, Proc. 1969 Joint Conference on Mathematical and Computer Aids to Design, pp27-58, 1969

Graphics in a Timesharing Environment, Proc on Computer Applications in High Energy Physics, pp552-564, 1969

Graphical Computer-aided Programming System (with F. K. Richardson), in Computer Graphics Techniques and Applications, ed R. D. Parslow, Plenum Press, pp109-117, 1969

Experience and Problems with the Software for the Automatic Solution of Ordinary Differential Equations, in Mathematical Software, ed. J. R. Rice, Academic Press, pp211-227, 1971

Transient and Steady State Numerical Solution of Differential-Algebraic Equations, Proc Joint US-Mexico Conference, 1971

A Generalized Interactive Network Analysis and Simulation System, Proc First USA-Japan Computer Conference, Tokyo, pp559-566, 1972

A Generalized Network Analysis and Simulation System, Proc Sixth Annual Princeton Conference on Information Science and Systems, **1**-3, 1972

Stiff Ordinary Differential Equations and Techniques Suited to Continuous System Simulation, Proc 1972 Summer Simulation Conference, San Diego, CA, pp223-226, 1972

The Stability of Automatic Programs for Numerical Problems(with K. W. Tu and D. S. Watanabe), in Stiff Differential Systems, ed. R. A. Willoughby, Plenum Press, New York, pp111-121, 1974

Estimation of Errors and Derivatives in Ordinary Differential Equations, IFIP Proceedings, pp447-451, 1974

Ordinary Differential Equation Techniques for Partial Differential Equations, in Formulations and Computational Algorithms in Finite Element Analysis, pp691-717, 1977

Simulation: Conflicts Between Real-Time and Software, in Mathematical Software III, ed. J. R. Rice, Academic Press, Inc, New York, pp121-138, 1977

Raster-Scan Hidden Surface Algorithms (with G. Hamlin, Proc Siggraph '77, **11**, pp206-213, 1977

Automatic Multirate Methods for Ordinary Differential Equations, Proc. IFIP '80, pp717-722, 1980

Initial Value Problems: Practical Theoretical Developments, in Computational Techniques for Ordinary Differential Equations, ed. D. Gladwell, pp143-162, 1980

Automatic Methods for Highly Oscillatory Ordinary Differential Equations (with K. A. Gallivan), in Lecture Notes in Mathematics, 912: Proc. Biennial Dundee Conference 1981, ed. G. A. Watson, Springer-Verlag, Berlin, pp115-124, 1981

Differential-Algebraic Equations Revisited (with Xu Xu Hai and L. R. Petzold), Proc Oberwolfach Workshop on Stiff Equations, June 29-July 3, 1981

The Automatic Treatment of Stiff and/or Oscillatory Equations, in Lecture Notes In Mathematics 968, Springer-Verlag, ed. Juergen Hinze, pp190-206, 1982

Stiff Software: What do we have and what do we need, in Stiff Computation, ed. R. C. Aiken, Oxford, New York, pp153-166, 1985

Smooth Numerical Solutions of Ordinary Differential Equations (with T. Vu), in Numerical Treatment of Inverse Problems in Differential and Integral Equations: Proc International Workshop, Aug. 30-Sept. 3, 1982, ed. Peter Deuflhard and Ernst Hairer", Birkhauser, Boston, pp 1-12, 1982

Differential/Algebraic Systems and Matrix Pencils (with L. R. Petzold), in Matrix Pencils: Lecture Notes in Mathematics 973, ed. B. Kagstrom and Axel Ruhe, Springer-Verlag, Berlin, pp75-89, 1983

Singular Implicit Ordinary Differential Equations and Constraints (with L. R. Petzold), Numerical Methods: Lecture Notes in Mathematics 1005, ed. Victor Pereyra and A. Reinoza, Springer-Verlag, Berlin, pp120-127, 1983

The Potential for Parallelism in Ordinary Differential Equations, in Computational Mathematics, ed. Simeon Fatunla, Boole Press, Dublin, pp33-48, 1987

Are Waveform Methods Suitable for Parallel or Vector Computers, in High-Speed Computing, ed. Wilhelmson, University of Illinois Press, pp143-145, 1988

DAEs: ODEs with Constraints and Invariants, Lecture Notes in Mathematics 1386: Numerical Methods for Ordinary Differential Equations, ed A. Bellen and C. W. Gear and E. Russo, Springer-Verlag, Berlin, pp54-68, 1989

An Introduction to Numerical Methods for ODEs and DAEs, in Real-Time Integration methods for Mechanical System Simulation, NATO ASI Series F, **69**, ed. E. J. Haug and R. C. Deyo, Springer-Verlag, Berlin, pp115-126, 1989

Parallel solution of ODEs, inReal-Time Integration methods for Mechanical System Simulation, NATO ASI Series F, **69**, ed. E. J. Haug and R. C. Deyo, Springer-Verlag, Berlin, pp233-248, 1989

The Speed of Waveform Methods for ODEs (with Fen-Lien Juang), in Applied and Industrial Mathematics, ed. Renato Spigler, Kluwer Academic Publishers, Netherlands, pp37-48, 1991

Research on Imprecise Computations in Project QuartZ, (with J. W-S. Liu, K. J. Lin and C. L. Liu), in Proceedings of the Mission-Critical Operating Systems Workshop, College Park, Md., Sep. 1989, ppF1-F10, 1989

The Development of ODE Methods: a Symbiosis between Hardware and Numerical Analysis (with R. D. Skeel), in A History of Scientific and Numeric Computation, ed. S. G. Nash, Addison-Wesley, Reading, 1990

Feature Grouping in Moving Objects, Proc 1994 IEEE Workshop on Motion of Non-rigid and Articulated Objects, Austin, Texas, Nov 11-12, pp214-219, 1994

Chapters in Books

High Speed Compilation of Efficient Object Code, in Computer Techniques, ed. W. Pollack, Auerbach, Princeton, pp211-224, 1972

The Automatic Integration of Stiff Ordinary Differential Equations, in Computer-Aided Circuit Design, ed. S. W. Director, Dowden, Hutchison & Ross, Inc, Pennsylvania, pp106-112, 1974

The Control of Parameters in the Automatic Integration of Ordinary Differential Equations, in Computer-Aided Circuit Design, ed. S. W. Director, Dowden, Hutchison & Ross, Inc, pp113-127, 1974

Numerical Software: Science or Alchemy, Advances in Computing, **19**, pp229-248, 1980

Numerical Computation (with J. R. Rice and J. M. Ortega and B. Parlett and M. Schultz and L. F. Shampine and P. Wolfe), in What Can Be Automated (COSERS report), MIT Press, Cambridge, 1980

Differential-Algebraic Equations, in Computer and Systems Sciences, Springer-Verlag, pp 323-334, 1984

Books

Computer Organization and Programming, Mcgraw-Hill, New York, 1969

Introduction to Computer Science, Science Research Associates, Palo Alto, 1971

Numerical Initial Value Problems for Ordinary Differential Equations, Prentice-Hall, Englewood Cliffs, 1971

Computer Organization and Programming, Second Edition, Mcgraw-Hill, New York, 1976

Introduction to Computers, Structured Programming and Applications Series, Science Research Associates, Palo Alto, 1978. A set of books that included: (1) Algorithms and Applications in Computer Science, (2) Algorithms and Applications in Science and Engineering, (3) (with F. Gustavson) Algorithms and Applications in Business, (4) Computers and Systems, (5) Programming and Languages, (6) Pascal Programming, (7) Fortran and Watfiv Manual, (8) PLI And PLC Manual , (9) Basic Manual

Computer Organization and Programming, Third Edition, Mcgraw-Hill, New York, 1980

Computer Organization and Programming, Personal Computer Edition, McGraw-Hill, New York, 1985

Computer Applications and Algorithms, Science Research Associates, Chicago, 1986

Books Edited

Selected Papers from the Second Conference on Parallel Processing for Scientific Computing, (with R. G. Voigt), SIAM, Philadelphia, 1987

Lecture Notes in Mathematics 1386: Numerical Methods for Ordinary Differential Equations (with A. Bellen and E. Russo), Proc. of Laquila Conference, Springer-Verlag, Berlin, 1989

Computation & Cognition: Proceedings of the First NEC Research Symposium, SIAM, Philadelphia, 1991

Reports of some interest

The Numerical Integration of Ordinary Differential Equations of Various Orders, Argonne National Laboratory, #anl-7126, 1966

Maintaining Solution Invariants in the Numerical Solution of ODEs, Argonne Report ANL/MCS-TM-40, 1984

Some Remarks on Inverses of Sparse Matrices, (with I. Duff, A. M. Erisman and J. K.), Argonne National Laboratory Report ANL/MCS-TM-51, 1985

Projective Integration Methods for Distributions, NEC Research Institute Technical Report 2001-130. See http://www.neci.nj.nec.com/homepages/cwg/pdf.pdf

Other

Maintaining the Growth of Science through Free Exchange of Information, ASM News, **56**, #2, pp64-65, 1990

The Automatic Integration of Ordinary Differential Equations, Current Contents, Engineering Technology, and Applied Sciences: Citation Highlight, **12**, #5, 1981

The Culture of Research, Keynote talk at Rutgers CAIP Industrial Affiliates Program and IRI Workshop on Academic-Industrial Relations, Duke University, Report 96-001, NEC Research Institute, 1996