THE BURROUGHS B 5000 CONFERENCE

OH 98

Conducted by Bernard A. Galler and Robert F. Rosin

on

6 September 1985

in

Marina Del Ray Hotel (Marina Del Ray, CA)

Sponsored by
AFIPS and Burroughs Corporation

Charles Babbage Institute
The Center for the History of Information Processing
University of Minnesota, Minneapolis

Burroughs B 5000 Conference
6 September 1985

Abstract

The Burroughs 5000 computer series is discussed by individuals responsible for its development and marketing from 1957 through the 1960s in a conference sponsored by AFIPS and Burroughs Corporation.

In the first session the technical aspects of the B 5000 and 5500 are discussed by a group of managers, engineers, and consultants. Topics include: the state of the industry in the late 1950s; the 5000's predecessors, particularly the ElectroData 101 and 201, B 205, and B 220; factors influencing the decision to produce the B 5000; reasons for designing the machine for ALGOL rather than FORTRAN and the effect of this decision on the computer's development and sales. The group reviews the MCP operating system, PERM, Polish notation, descriptors, stacks, the BALGOL compiler, and other innovations of the computer.

In the second session, the same group discusses the development of the B 5000 into a product, including the effect of the administrative organization on the project; the relations between hardware and software engineers; the interaction of project personnel and upper-level management, field marketing, and customers; the COBOL processor, the head protract disk system; the operating system; ALGOL; and documentation of the computer.

In the third session managers, sales personnel, and customers of the B 5000 discuss Burroughs' product line before the 200 and 5000 series computers; sales training and market reaction to the B 5000; acceptance of B 5000s at Ohio Oil Company and Stanford University; its rejection by the University of Michigan; reasons why the B 5000 was not marketed overseas; and Burroughs' presidents Raymond Eppert and Ray MacDonald.

Technical session participants included: Robert S. Barton, Bobby A. Creech, David M. Dahm, Benjamin A. Dent, Bernard A. Galler, John E. S. Hale, Erwin A. Hauck, Paul D. King, Norman Kreuder, William Lonergan, Duncan MacDonald, F. Brad MacKenzie, G. Clark Oliphint, Robert F. Rosin, Lloyd Turner, and Richard Waychoff.

Marketing session participants included: Henri Berce, George A. Collins, James Ford, Bernard A. Galler, John E.S. Hale, Joseph T. Hootman, Paul D. King, F. Brad MacKenzie, Ralph W. Pearson, and Robert F. Rosin.

BURROUGHS B 5000 CONFERENCE

DATE: 6 September 1985 MODERATOR: Bernard A. Galler

LOCATION: Marina del Rey Hotel (Marina del Ray, CA)

BERNARD A. GALLER: I'm Bernie Galler; I'm representing AFIPS and the Annals of the History of Computing in this discussion. The date is September 6, 1985, and we're meeting in the Marina del Rey Hotel in California. The subject of the discussion for which we brought a number of people together is the Burroughs B 5000 and 5500 systems as projects within Burroughs, as products from Burroughs, as innovations to the computing world. We've brought a group of people together for the morning session to represent the technical aspects of the systems; this afternoon we'll have another discussion with people who represent the marketing aspects, although I'm sure that both groups will cover aspects of each. I'm going to ask the people who are sitting around the table this morning to identify themselves and give some idea of their role at the time and their present affiliations. Let me first introduce Bob Rosin who also represents the Annals of the History of Computing. Bob is with Syntrex, and is one of the editors of the Annals. To my left is Rosamond Dana, who is the managing editor of the Annals and will be helping with the transcription process, and will be in touch with you all later regarding excerpting the discussion for publication in the Annals. We all understand that the actual tape and transcript will be deposited with the Charles Babbage Institute to become part of its archives. Let us now go around the table and introduce ourselves.

DUNCAN MACDONALD: I'm Duncan MacDonald. I guess I was director of engineering at the time; so long ago that it's hard to remember some of these things. I hope you today have a better memory than I do. I am with Burroughs from 1951 to 1972; now I'm running my own small company in computer storage equipment called DMI.

ERWIN A. HAUCK: My name is Erv Hauck. I was with the Burroughs Corporation from 1956 through 1982. The assignment I had during that period was as a hardware designer on the 5000/5500 systems. I worked principally in the area of I/O and central control. My current job is with a company called Technology Marketing in Irvine, California.

BOBBY A. CREECH: I'm Bob Creech. I joined Burroughs on February 6, 1962, as a senior automatic programmer. Automatic ones were the best kind. I worked as a member of the ALGOL compiler team and later worked on the implementation of the disk file MCP. I was with Burroughs as manager of engineering for the Systems Engineering Division on the East Coast.

F. BRAD MACKENZIE: I'm Brad MacKenzie. I joined Burroughs in 1959, and I'm still with Burroughs. During the period we're discussing, I was manager of Automatic Programming in Pasadena. I'm presently manager of Product Assurance in the plant in Tredyffrin, Pennsylvania.

PAUL D. KING: I'm Paul King. I joined Burroughs in mid-1954, and I left in August 1962. I came back in mid-1965 and then left in the end of 1969. At the time of the subjects we're talking about in this meeting, I was the project manager of the B 5000 in Product Planning. I'm currently with Citicorp in the Corporate Technology Office.

WILLIAM R. LONERGAN: I'm Bill Lonergan. I was with Burroughs the first time from mid-1953 until April 1961. At the time of this project, I was the manager of Product Planning for the ElectroData Division of Burroughs and had responsibility for planning and specifying the product line of the division. I'm currently a general partner in a venture capital partnership.

RICHARD WAYCHOFF: My name is Richard Waychoff. I joined Burroughs in March 1959 and left in September of 1979. I was an automatic programmer on the ALGOL compiler project, and currently I'm director of technology for RDS Systems, Inc.

JOHN HALE: I'm John Hale. I joined Burroughs in October 1956 in the Dallas district. My association with the B 5000 began in January 1962, when I transferred to Pasadena as manager of the COBOL compiler group. I retired from Burroughs in 1981 and am currently a self-employed consultant.

LLOYD D. TURNER: My name is Lloyd Turner. I was with Burroughs from June 1959 until May 1981. I started as a sales technical representative in the Dallas district, and when I left I was corporate vice-president of engineering. During the B 5000 project, I was manager of Scientific Systems Development that had responsibility for the ALGOL compiler and other scientific compilers and the disk file MCP. I'm currently president and chief executive officer of Floating Point Systems, Inc., in Portland, Oregon.

ROBERT BARTON: I'm Bob Barton. During the period of interest, I was not employed by Burroughs. As I recall, I had a consulting contract with Product Planning in the beginning, and later, I think, with Engineering, although it's beyond that period. I am currently a principal of an organization called Innovations and Inventions.

DAVID M. DAHM: I'm Dave Dahm. I worked for Burroughs in the summer of 1959 and the summer of '61. I became a permanent employee in June of 1962 through July of 1965. During that period I was a senior automatic programmer; I was part of the group, and project leader for the disk file MCP. Currently, I'm a Burroughs employee; and I'm director of Programming Research.

G. CLARK OLIPHINT: I'm Clark Oliphint. I worked for Burroughs from mid-1959 until 1964. My title during the period we're interested in was manager of Control Systems development. I was responsible for developing the first MCP on the B 5000. I'm currently an independent or multidependent consultant.

NORMAN L. KREUDER: I am Norm Kreuder. I came to Burroughs--actually to the ElectroData Corporation--in the spring of 1954; Burroughs bought us in the summer of 1956. I left Burroughs at the end of 1971. At the time in question, first I was engineering section manager, responsible for the hardware design of the 5000 system, and then later became engineering program manager. I currently have a little company in Santa Ana, California, called PC Horizons.

BENJAMIN A. DENT: I'm Ben Dent. I joined Burroughs in the summer of 1956 in Paoli, and I'm still employed by Burroughs. In Paoli, we worked on a somewhat similar machine. It was supposed to be a multiprogramming, multiprocessor machine called the D 825. I became quite interested in operating systems, and then I decided to move to California for personal reasons. So I actually joined the B 5000 project after most of the machine was designed, but before the first operating system or the first customer shipment. I have no idea what my title might have been at that time. I'm currently with Burroughs in an organization called Distributed Systems Group as director of engineering.

GALLER: Thank you very much. I'd like to start the discussion by getting the context in which all of this happened. And first a small procedural question. In the hope that more of you want to talk, I will ask you to signal to me. I will keep a list and call on you in order so that we don't interrupt each other if possible.

I'd like to start by briefly describing the state of the industry. The time period we're talking about is 1958 to 1962, and maybe a little beyond that as we get into the 5500. Who were the players, and what was the industry like at that time? Or what did you think it was like at that time?

HALE: At the time I was in the Dallas district. Burroughs had a machine called the B 200 which was basically a high-speed card reader (as I recall,that was its design). A competitive machine was the IBM 1401. One of the motivations for my transferring to Pasadena was because Burroughs had a machine called the 251, which had some monstrous ledger-card processing mechanism that was just starting to be delivered in the field, and I knew I didn't want to have anything to do with that. I was very happy to get an offer to go to Pasadena.

KING: Let me talk about some of the systems that existed at that time. IBM was currently in the 7070, 7090, the 1401, 1410. RCA had the 501. Univac II, III had been announced, I believe. Then there were all sorts of other companies. Underwood was still around, and a variety of others that dropped by the wayside. Bendix had the G-15. Honeywell had the BIZMAC, and that was about it. In terms of Burroughs, we had the 205, which was still sort of being produced; the 220, which was on its last legs; the 2111 had been killed; the B 200 was still in development at that time. It had not yet come out. The B 200, as originally conceived, never went any place. I think they built 50 of them. It was the B 300 that really made it because the B 200 didn't have any tape, no disk, no nothing.

KREUDER: Two card readers.

KING: Two card readers, a card punch and a printer. Only 60 of them delivered overall. The 2111 was killed at the end of 1959--the first of 1960.

ROSIN: Were these solid-state machines or vacuum-tube machines?

KING: The 220 was vacuum tube, the 2111 was going to be solid state, but it had a delay-line memory. That's within Burroughs. The 200 was a solid state machine.

GALLER: We'll come back to the 2111 as part of the decision-making process perhaps. Norm, you wanted to say something?

KREUDER: There was a question about Honeywell. They were in a joint venture with Raytheon called Datamatic at that time.

GALLER: Yes, that was it.

KREUDER: How's that for dredging back a bit? There are others in the field, of course, that aren't around any more.

GALLER: Sure.

KREUDER: RCA...

GALLER: Wasn't the BIZMAC by RCA?

KING: Yes, that's right.

GALLER: Just for the record: thank you.

KREUDER: RCA and GE were both spending a lot of money and Bendix, here on the West Coast...

?? Philco S2000.

KREUDER: Philco was a heavy player. The field was much more heavily populated than it is now. There's been a lot of shaking out as...

GALLER: What was Burroughs's position in the industry? What kind of image do you think they had? Who was the market for Burroughs machines at that time?

KING: It was a mix of scientific and commercial. I would guess, about 60 percent scientific, 40 percent commercial. Some 205s were delivered to people like Allstate and some of the banks.

LONERGAN: I was going to say that's really the ElectroData Division. If you say, "What was Burroughs doing at the time?", Burroughs was heavily in banks with electromechanical bookkeeping machines. That's where its main revenue was. Paoli had done some fairly good early machines, you know, for NSA and other people.

GALLER: Brad?

MACKENZIE: The 205 had been out of production for some time. It seems to me that the 220, Duncan would know, but it seems to me that started in 1959 or something like that and there were about 50 of them built. The 205s probably weren't built after 1958 or 1959. There were probably a couple of hundred of those.

GALLER: Do want to comment on that, Duncan?

MACDONALD: Well, I'll comment just a little bit on market. I've already apologized for my memory. I think the company was in transition at that time. The Pasadena operation knew quite a bit about scientific machines and not much about banking and commercial. The system architectures were in transition from knowledge about scientific operations to doing something about handling the commercial market with some speed. So, I think everybody was learning how to do that at the time. I don't think there really was a decent commercial system, but that whole transition was taking place. I don't think it really completed that transition until something like that 6500 came along. So, you're looking at a fluid and dynamic marketplace.

GALLER: Dave?

DAHM: I think one of the things that was going on in the time we're talking about was the invention of ALGOL. That had an enormously important impact on the design of the B 5000. I think Burroughs had the idea that ALGOL was to become the standard scientific programming language and Burroughs was wrong going about that. Basically, ALGOL, of course, did not displace FORTRAN. The B 5000 clearly was a machine that was designed with the idea of being a very good ALGOL processor, and that was due to this belief that ALGOL was basically going to become the scientific language of choice in the industry. You recall at that, at about that time, the *Communications of the ACM* had a policy that all algorithms that were published would be published in ALGOL. That was controversial at the time. There were some die-hards that believed algorithms should be published in FORTRAN also.

GALLER: We will come back to that question. Let me keep going.

LONERGAN: Are you going to have a separate topic having to do with what was happening inside of Burroughs?

GALLER: We sure are.

LONERGAN: Some of Duncan's are comments relative to that should go back and talk about the acquisition of ElectroData and BEAM 4 and 220 and all of that if you want to set some of that stage, and I can certainly do that...

GALLER: Let's try to get some of that background in now. Don't jump too fast into the B 5000; we'll get there. Let's get the background.

LONERGAN: I can speak to that because when I said I started with Burroughs in 1953, it was in Paoli, I was doing logical design on missile defense systems and such things as that on the military side. Then Burroughs did the E101 in Paoli, and then started doing a machine called the BEAM 4, which was aimed at trying to do what Duncan said earlier, namely to somehow or other get this computing talent in Paoli to produce machines which would be helpful to the main Burroughs thrust into banks and utilities and other such things. In about 1955 or early 1956, they acquired ElectroData, which had been owned by Consolidated Engineering at the time. Jim Bradburn had spawned it out of there, and was still running it. We had a lengthy series of sessions between Paoli and Pasadena relative to presenting the 220, which was further along than the BEAM 4, which was being done in Paoli. I think rightfully so; I was a party to those discussions and decisions and so was Ted Glaser and Ewald Kleim, and I'm sure Duncan, and Norm, and lots of people. In any event, I think the corporation made the right decision, which was to proceed with the 220 because there was far more knowledge in Pasadena about getting out a commercial product and supporting it and all that kind of thing than there was really in Paoli, which was been mainly making military kinds of systems. It still does.

GALLER: Let me interject one question, which I hope you'll amplify on. Were these machines, each one of them, seen primarily as commercial or as scientific, or were all of these machines somehow seen as both?

LONERGAN: They are seen as both, but I would say the intention of the BEAM 4 was to be a commercial machine. If there is a difference, there, it wasn't really clear other than that one had 701s and 702s and 705s. In any event, the emphasis was certainly to be a product which would go into the commercial market. Now, as has earlier been said, most of the 205 machines I think were in scientific kinds of applications like wind tunnels and other things so they were clearly on the scientific side. The 220, by design intent, had more characteristics in it which tended to make it more applicable for the commercial world. At any event, that happened, and the BEAM 4 was killed. Since I wanted to stay with the commercial side of things, I transferred to Pasadena in the middle of 1956 and was the first person in Burroughs, really, to move out. So I had fair connections with the company in Paoli and Detroit, and then came out to Pasadena. Then other people followed: Donald Stevens came, Howie Short was maybe the second person, I don't know. So the 220 progressed, and Paoli went on to other kinds of things.

GALLER: Ben, you're on the list.

DENT: Well, I was just thinking we might want to talk a little bit about the importance of Pasadena as a scientific center really based around Cal Tech, which spawned JPL. I always thought that ElectroData really started because of that organization and that was their background and thinking. That's where they thought machines could be used, and that's where the basic technology came from. Perhaps that's important. Perhaps it's important that all of the early computing centers started around universities. Boston, a lot in Philadelphia, Pasadena was one of those.

KREUDER: The original computer that was developed at Consolidated called the 201 was produced by Consolidated as a way of massaging the data that they were acquiring from the mass spectrometer. Actually, this whole thing was a spin-off from a mass spectrometer project, but then it turned out once they had enough of a machine to handle the mass spectrometer data, they could also do some other arithmetic and take care of the payroll. That's, I think, when Jim Bradburn saw the light and spun it off as an independent organization. Separately financed...

MACDONALD: Yes, that's right.

GALLER: I see. Paul, you're on my list.

KING: Oh, well, I could elaborate on that, but I wanted to go back to Dave's discussion about ALGOL. Also at that point in time, the COBOL committee was meeting and trying to resolve what COBOL was. It had not yet been specified. It was still that mysterious thing that was going to be this common business-oriented language.

GALLER: Right. That coalesced around 1961. In fact, we are now at the 25th anniversary, and are about to come out with a special edition of the Annals.

KING: Going on to Norm's point, the original 201 was going to be a matrix processor, and that was it.

LONERGAN: An interjection: the name BEAM 4 stands for Burroughs Electronic Accounting Machine. Clearly, the intent was to go after commercial kinds of things.

GALLER: Thank you.

ROSIN: While we're discussing this topic, I wondered if people would care also to interject their recollections about the mood in Burroughs, including the various divisions at this time. Optimistic, pessimistic? Reaction to what was going on outside?

GALLER: Brad?

MACKENZIE: Well, I wanted to say one thing on the 205, because I think that we set it up too much as a special machine. It was really a general-purpose computing machine. It was a peer of the 650; it was certainly a very well-designed machine, it was used in not only scientific work, but also commercial work--not large accounts in but commercial work--but there wasn't that much commercial work going on at the time. The 205 was a really good solid machine for its time, and I think that's provided the basis that Bill was talking about that led to the 220 and things like that. There was a real solid core of engineering capability in the 205 and some very rudimentary programming works that were going on. Bob Barton later brought it in from Shell Development; you had a 205 down at Shell. And they were all around. They were good machines.

GALLER: Bob?

CREECH: A couple of people have barely touched on the subject of programming. I viewed the world in those days--and still do--from a programmer's point of view. The state of the art in the industry in regard to programming: I even felt it then was appalling before I even found about Burroughs and the machines they were designing. They had just gone through the debacle of SOS. The operating system--as it was called for the 709 originally, which became the 7090--the squoze deck. The solution to symbolic debugging was to generate a squoze deck from a high-level language and then load that fast.

GALLER: That was the SHARE operating system (SOS).

CREECH: That was the point. I was in SHARE as a customer, and it seemed that in SHARE, the users were doing all the software development--although that was not completely intended, I think; I'm not sure about all of that. I remember FORTRAN was the high-level language in use. It was I think a 13-pass compiler where any one pass couldn't keep a 150 card-a-minute going. I was, as a result of seeing that, appalled at the idea of high-level languages and how inefficient they were and so forth. I was head of a systems programming group for a user of a lot of IBM machines when the B 5000 was announced, and we were going through an acquisition. I was viewing the process of "going out for bids" for this machine; there was really only one contender considered, and it wasn't Burroughs. I pleaded with my boss, "Why aren't we considering Burroughs?" And his answer was, "No software." I never forgot that. They were not considered because they didn't have any software.

GALLER: Paul?

KING: I wanted to follow up on Bob's inquiry, and that had to do with Burroughs. The fellow who preceded Ray Eppert as president of Burroughs was John Coleman. He was a very, very aggressive individual. He dragged Burroughs kicking and screaming from being Burroughs Adding Machine Company into the electronic world. He got them into what was called the Sensomatic, which turned out to be a gold mine for them. He also was the driving force behind the acquisition of ElectroData. He died, I believe in 1960 or 1961; he died in the office, as I recall.

MACKENZIE: I think it was earlier than that--probably about 1959.

KING: The whole computer business, I think, was losing money during that period, but Coleman kept pushing it. And then Eppert took over, who had been executive vice-president.

GALLER: Now, Burroughs had the 205 and the 220, and those were fairly successful machines. There is always a question, then: what is our next product? A 2111 was coming along. That was stopped. Why? What was going on at that time? What was that machine supposed to do? Norm?

KREUDER: The 2111 was a delay-line-memory machine. It was thought that it would be almost impossible to sell a thing like that in the core-memory world. It was probably capable of pretty reasonable throughout--a lot of bang for the buck--but if you can't sell it, then why bother with it? That was one of the main reasons, as I recall.

ROSIN: Was that also supposed to be a business scientific system?

KREUDER: Very definitely. It was the first machine we had worked on, I think, where we had really taken our understanding of commercial data processing into account in the design of the machine.

GALLER: What kind of delay-line, mercury?

KREUDER: No, no. They were...restrictive, torsional...

GALLER: Why in a core world was that being done at that time?

KREUDER: Well, we didn't know it was going to be a core world until the 1401.

KING: There was a rule that you could not build a core machine for under $30,000 a month. We wanted a machine that was $15,000 a month. So, it had to be something else as the memory. It started out with a delay-line memory, and the CPUs oriented around that. But then we added an I/O processor onto it--it was a core-memory processor--as some means of massaging the data. And then we put on auxiliary storage, (drums) and we had a number of things on it. But the fact that it had that delay-line memory, and the 1401 was announced.

GALLER: So, Burroughs at that time had a couple of maturing products: the 205 and the 220, and there was a need to do something else. What happened?

MACDONALD: Well, you've talked about hardware as far as the delay-line memory was concerned. I think probably equally important was the revolution occurring in the software side at that time. There were all things having to do with multi-tasking, with MCP control, and with virtual memory. All were being developed almost simultaneously. The state of development was in such a state of flux that when some of the, I guess trigger limitations, of the 2111 system were recognized, everything else necessary to put together the 5000 was sitting there on the shelf ready to go.

GALLER: On various shelves around the world.

MACDONALD: Well, I'm not sure. I don't feel that way. How does the group feel?

ROSIN: Where were they?

MACDONALD: They were right there.

ROSIN: In Burroughs.

MACDONALD: I don't remember.

KING: Things were not yet all in a coherent...

MACDONALD: No, but the ideas were there.

KING: The pieces were scattered around.

MACDONALD: I don't know what the source of them was. I don't think anybody thought about that at the time. At least I don't remember.

GALLER: That's our next topic, isn't it?

LONERGAN: One last comment about things going on in Burroughs. There was a machine already alluded to--other people called it the "Ledger-Snatcher"--which was an attempt to automate the handling of ledger cards which had been processed in the Sensomatic--and to do it automatically. So you would do file updating by having a big stack of these cards go through this huge mechanism. They had a lot of electronics with it and the idea was to preserve a ledger card as a visual record. In fact, it was called the Visual Record Computer (VRC). It was a monster, to put it nicely, I guess--difficult to keep going mechanically. Paoli was building the mechanical mechanism part of it, designing it, and they did have some good design mechanical engineering talent at Paoli. They built some very successful high-speed check sorters and other things. In any event, that product got overtaken by the fact that one could outperform it in a cost-performance sense on 1401-class machines or on what was by then the B 200. Prior to that in Project Planning, which I also had responsibility for, Dan Nielson was running the 3500 project. The VRC never got announced in Europe; Ray MacDonald didn't believe in it, so he didn't announce it in Europe, but in the U.S. it was out there and installed. It was clearly not the right way to do things; it was a dinosaur.

MACDONALD: I wonder how many people will remember how that came about. I remember being intimately involved with it. That monster was a pure political ploy. It had nothing to do with technical capability. The purpose of that was to co-op the Plymouth Division and keep them from getting into the computer business, because they had developed that monster...

MACKENZIE: They did pretty well, then. [Laughter]

MACDONALD: You remember that? They had developed that thing as an outgrowth of their ledger capability, and they were getting ready to release it as a commercial system. Of course, that couldn't be allowed to happen. So the monster was grafted on to a Pasadena computer at the corporate level as a political ploy to keep them out of the business.

GALLER: And it worked?

MACDONALD: The ploy worked! [Laughter] The system died a merciful death.

LONERGAN: There are lots of people left who still have scars from that machine.

GALLER: Well, we may get into that this afternoon a little bit. Let me read you part of an article by Lonergan and King in Datamation (May 1961) called, "Design of the B 5000 System." There's a statement of design criteria for the B 5000 system. I'd like to read those to you and talk about them a little bit. In particular, as I read them, I'd like you to think about whether there was general agreement on these as goals; whether in 1961 these were formulated after the fact as what turned out to be the goals; ...I will ask you whether these were goals at the beginning of the B 5000 project. Let me read them though. "Early in the design phase of the B 5000 system the following principles were established and adopted: program should be independent of its location and unmodified as stored at object time; data should be independent of its location; addressing of memory within a program should take advantage of contextual addressing schemes to reduce redundancy; provisions should be made for the generalized handling of indexing and subroutines; a full complement of logical, relational and control operators should be provided to enable efficient translation of higher-level source languages such as ALGOL and COBOL; program syntax should permit an almost mechanical translation from source languages into efficient machine code; facilities should be provided to permit the system to largely control its own operation; input/output operations should be divorced from processing and should be handled by an operating system; multiprogramming and true parallel processing (requires multiple processors) should be facilitated, and changes in system configurations (within certain broad limitations) should not require reprogramming." Now, the questions I'm asking: Were these generally accepted as the design goals of the project?   Were they articulated early? Were they recognized later? What happened? Norm?

KREUDER: Well, since I was the recipient of the spec from Product Planning, I can tell you that those were all in there from the start as goals. One possible exception that grew out of later considerations has to do with the expendability without reprogramming. The mechanism for doing that had been designed (in Burroughs Paoli machine; the 825) in the form of a thing called the switching interlock, which was the crossfire-type exchange that permitted processing units to talk to memory units without any hardware-type assignment. I remember Lonergan had said, "We'd sure like to have this thing in the 5000." Of course, being an engineer, my question was, "How much are you willing to pay?" And Lonergan said, "If we can do it for 5 percent, with no more than adding 5 percent of the cost of the mainframe." I disappeared for a while with a sharp pencil and did some calculations and talked to the guys in Paoli. I said, "We can't do it for 5 but we can do it for about 6 percent." He said, "Let's do it." And I said, "OK." So that was a goal that turned up when we were still... We hadn't put any designs down hard yet, so I guess you'd call that an initial goal. By the way, that's about what it came out to--about 6 1/4 percent.

KING: One of the instigators of that desire was some meetings that we used to have once a week in Don Stevens's office. It was Duncan MacDonald, Don Stevens, Bill Lonergan, myself, Ted Glaser, and Howard (Barton). We talked about what systems should be designed like, and Duncan pushed us toward the design of the telephone central office. I remember we used to go downstairs and look at a PBX and the way it worked and the way it processed things and the way you could knock something out and it would just keep working. We thought about how we could switch things in and out if we had the right kind of switching. But we could never get that right kind of switching until Paoli came along with it.

ROSIN: It had already existed for the D 825? Is that right?

KING: It was under development, wasn't it?

KREUDER: It was under development, and they were talking about it. Our execution of that principle was far superior to theirs in that their switching interlock was only a character at a time. They had a fast memory that was like the 220, which sort of destroyed itself because, although you had a fast core memory on it, they would then shift the characters around the system, ca-chunk, ca-chunk, ca-chunk. After about 11 of those, you'd have transferred the word from this high-speed memory to high-speed processor, but all the time was gone by that time. The Paoli execution, I believe, was also a character at a time. They were, of course, appalled at the amount of copper and germanium that I was going to use doing it a word at a time. But would have been crazy to do it any other way.

GALLER: Was the 825 delivered? I mean, was it a working system?

KREUDER: Oh, absolutely.

MACKENZIE: Yes.

GALLER: The reason I ask is that there were some of us who wanted to order civilian versions of that, and they never got delivered.

KREUDER: Well, you tried to place your order in the wrong town! [Laughter]

MACKENZIE: Ben, you worked on that didn't you?

DENT: I don't know that it was ever delivered commercially or even thought of commercially. Lots of them were delivered, but they all disappeared in the mountains somewhere.

ROSIN: Were there any D 825 alumni directly on the B 5000 project?

DENT: I don't know. I thought I was the first person to ever manage to work in both places.

LONERGAN: Well, Bob Barton was consulting to both groups. Weren't you really? Weren't you doing some consulting in Paoli at the time? I thought you were.

BARTON: I don't think anything significant, Bill. It was an amusing episode, but I don't think it's terribly relevant to this discussion. I'd say the answer is no.

LONERGAN: They did get a project going subsequently called the 4400, which happened to be the telephone number at Paoli. At the same time we had a project going in Product Planning which was known as the 4000, and subsequently those two things were kind of merged and became the 5000, although I guess it was probably 98 percent of the Pasadena product and maybe 2 percent of the Paoli one, or some number like that.

MACKENZIE: Paoli had the modular processor proposals at that time--the A, B, and C. Do you remember that? That was their thinking on this.

KING: Yes. Well, that was your work, Bob. Because Jim Anderson said that you were a consultant on that.

BARTON: No, I think his chronology was wrong. Any consulting I did with Paoli was well after this period; it was when Ted Glaser was running his little research group.

GALLER: Oh, OK.

BARTON: I'm sure there's a separation in time, but I don't know the details of the separation.

GALLER: OK. Dave?

DAHM: The goals that you were alluding to in May of 1961 were there from the beginning, but the ideas as to just exactly how those things would be accomplished were really fairly vague for a long time. There was a substantial amount of invention that was required to, in fact, turn those goals into deliverable product. Many of those things really didn't become commercially viable, so to speak, until maybe about 1965. It took a long time, in fact, to turn the goals or the vision into workable algorithms, for doing storage allocation, virtual memory, multi-tasking...

GALLER: And therefore the B 5000 itself, when it first came out. Well, would you say it didn't do those things or it did them, but poorly, at first?

DAHM: I would say that some of those things it really didn't do when it first came out and some of them it did very poorly. Eventually we got there but it required a substantial amount of invention to get from the vision as to where we ought to go to the end product. In 1961, I think at least, the ideas as to exactly how one would do storage management were really fairly vague. There was a lot of mechanism built in for the architecture to facilitate it without which it would have been impossible. But just exactly how you did this thing wasn't all that clear.

GALLER: According to the state of the art of the world, a lot of it wasn't clear at all and you should understand that a lot of this was innovation. Lloyd?

TURNER: In terms of setting the stage of what was happening at that time, one of the things is that certainly the 205 was a very old machine. The 220 was supposedly a competitor to the 7070, which is a transistor machine. So from that standpoint, the 220, although it was nice and easy to program, easier than the 205, and very similar to the 205, was anything but a world beater.   But the one significant thing I think that came out of the 220 was the BALGOL compiler which was done earlier than the 5000. That was all done and working nicely, and they actually even had an operating system working on that machine. When I arrived in 1960 on the ALGOL project, the machine was viewed as a machine that would execute ALGOL, that would have virtual store if you could call it that at the time, that would have dynamic modularity, that would have, in terms of memories and all the rest of the components you could put on (including processors), it would be a multiprocessing, multiprogramming type of machine. All of those things and all of those goals were designed into the machine by the time I arrived in 1960. So the 220, we'd have to say, was not a commercial success, but it did provide a vehicle for Burroughs to get into the language processing business.

GALLER: Maybe you could say a little more. What was on the 220? What characteristics did the operating system have? What was BALGOL like in terms of setting the stage then?

TURNER: Well, Dave worked on the BALGOL compiler, so he probably knows more about what it had and so did, I guess, Clark. Did you work on it too, Clark?

OLIPHINT: Very slightly.

TURNER: Certainly Dave did, and Bob was running the thing at the time, so he knows what it was all about.

?? The Burroughs Algebraic compilers?

DAHM: Well, BALGOL started off to be the international algebraic language which was later called ALGOL #58, but toward the end of the project, it...

TAPE 1/SIDE 2

DAHM: ...Pasadena.

ROSIN: Was it in response to customer requests or Burroughs instigation?

GALLER: The world was not clamoring for ALGOL at that time.

MACDONALD: Things at that time weren't done in response to external stimuli. [Laughter]

GALLER: That's very interesting.

MACDONALD: They weren't. They were internally created in an atmosphere that was highly creative, highly charged, and largely running on its own steam.

GALLER: And not necessarily market-driven, then?

MACDONALD: No, it wasn't. Absolutely not. Because you see, it was so advanced that there weren't any external stimuli. They didn't exist at the time to a large extent.

GALLER: How did you justify the effort within Burroughs?

MACDONALD: Didn't have to. Nobody else in Burroughs knew what was going on.

MACKENZIE: As long as you didn't try to hire more people.

MACDONALD: Right. As long as you stayed away from the administrative restrictions, you did what pleased. That's oversimplified, but the atmosphere was very much that way. Very unique. I've never seen it since.

MACKENZIE: There were about fifty 220s built and they got done, in a sense, by the fact that they were not a transistor machine. But there was one other thing on the 220 which I always thought turned out to have been very significant. It was that old clunker mass file, Bob. The notion was there that it became a repository for a place to store programs inside the system. It hadn't been built for that reason. I think it had fifty tapes and they were each 200 feet long, and it was meant to be a mass storage device. I don't think it was ever successful in that respect, although Allstate bought all kinds of them for a while there. But it became a repository: a place to put the utility programs, to store programs; in a sense, it was on-line to the system. I always felt, in a background sense, though I didn't do that work, that was in the back of people's minds when they, in a systems sense, were going at the 5000 later on as the types of things that ought to be done.

GALLER: Bill?

LONERGAN: I was going to comment on two things. One has to do with this question of objectives. Actually, with regard to the continuing operability of the machine in a sense of what is now the Tandem [Corporation] sense or other people, I had really wanted the system to be more that way than we were able to do. I think the state of the hardware art was not such that you could really do it, where you can pull out a card and not have a big spike on the power system and cause all kinds of problems. In terms of being able to grow by adding components without having to reprogram and keep the system running even when you had to take a processor off-line to service it, we really wanted to be more than, I'd say, that document even says. We had some aspirations beyond what actually got into print. One other comment I would like to make. I agree with Duncan that things weren't marketing-driven at the time. A little personal note: I had the most extraordinary array of talent in my group, in an IQ sense, or creativity sense, that I've ever had before or since--a remarkable group of people, all with some quirks like Jack Merner's tie which he only tied once after we bought it. [Laughter] He'd slip it up and down every day, and the tail end was about 8 inches longer than the front; finally some secretary after 6 months clipped it off with a pair of scissors.

KING: It was me. He didn't believe I was going to do it. I went clip! and he said, "You did it!" Ruined his tie!

LONERGAN: I'll try to stick to the question of outside contact; we really did have some. I remember meeting with Paul Armer at Rand and other people at Rand and met with the people at Stanford. Paul and I, I think, met with Alston Householder down in Tennessee. So we had sessions with people reviewing things that we were doing, but I agree with Duncan that fundamentally it was an internally driven project, not marketing.

GALLER: To what extent is that still true in the company? Have things changed?

KREUDER: Anybody still work there?

DENT: Things have changed a lot. [Laughter] I make the distinction between market driven and marketing driven. Burroughs has gone all the way to marketing driven, you only seem to be able to justify projects that marketing wants. And, of course, the problem with that is that they want what they can sell today or what they lost their last sale because of the lack of, which means that they're always asking for things that could be sold today. The problem is that is takes us a year and a half or so to build them. And so if they want this we start, and before that year and a half is out, they want something else. Now, acknowledging that, if you must try to project forward to what the market will need three years from now--we must fill that--it doesn't pass the marketing test. Nobody wants that. Maybe it's a good idea, but nobody wants it, there's no forecast for it, and therefore the project isn't justified.

GALLER: I would guess that with the rate of change in the industry now, that's true in many companies.

KING: On this thing of marketing driven. One of the things that did happen at that point was we tried to understand what the customers' requirements were because we found out that if you asked a customer or a programmer or anybody what do you want, all they could do is talk in terms of what they had at that time. I can remember within Burroughs we had a fellow, Chuck Macon, who was in charge of magnetic tape development at that time. We used to have a lot of difficulty in talking to Chuck about what the tape units should be like. So finally we got him to go and sit down in one of the big IBM installations out at Lockheed. He sat there and watched what they did with tapes all day. From then on, we got along beautifully. He saw what happens in a large installation using lots of tapes.

GALLER: All right. You cause me to ask, then, who were the customers for the B 5000? Who were the intended market?

KING: Our existing customer base. And people like that.

GALLER: What did they look like? Take the scientific/commercial split as a very crude one that was seen at the time as a category.

KING: C. F. Braun was an engineering company that built refineries. (Safeco?) was general insurance. Who were some of the others? Allstate...

TURNER: NASA was the first one, then United Technologies.

MACKENZIE: I have a list of them here. UTC was the first customer delivery on April 1st. NASA was the second one. Early on were Northern Natural Gas, Dow Chemical, L.A. City Schools, C. Braun; Stanford University got System 110, U.S. Army Signal Corps. That same summer there were two systems that were shipped to the First Pennsylvania Banking Company.

GALLER: OK. These are the people who ended up getting them. But when you first thought of doing the system, who did you think you were doing it for?

KING: Also, if you go back and look at the 205/220, those are 205 and 220 customers. It was that group of people that the sales force were talking to who were already Burroughs customers; it was a mix of scientific and commercial.

ROSIN: Let me follow up on that. We heard earlier that there was some agreement (I'd like to explore how much agreement) that ALGOL was going to become the standard scientific language out in the user community. Did the scientific users among this group we've just discussed share that belief? I'm curious who championed that belief, and I'm also curious who, if anybody, inside Burroughs resisted that belief?

GALLER: Did BALGOL catch on so much that it was obvious that you had to do it that way, for example?

LONERGAN: I was going to comment more generally about that. I think in looking back one could say we were fairly naive. On the other hand, at that stage of things, it hadn't become as patently obvious as it has since that IBM will not adopt industry standards unless it's to their economic advantage to do it, and it seldom is. They prefer to go their own way. They have now a twenty year history of doing it. That wasn't clear then; now it is clear. You better not do that.

ROSIN: So it was a business decision?

LONERGAN: But they had done the same thing with ASCII. If it's not to their advantage, they can't adopt it.

DAHM: I think it was a very complex thing in some respects. I remember when I was a summer employee in the summer of 1959, I was working on the (SPALGOL?) compiler with another fellow named Joe Erdwin, who was the project leader. And we were busily trying to do our version of IAL, and one day Bob Barton came along and he had a FORTRAN manual in his hand. It was a really nice, well-done FORTRAN manual done by IBM. He said, "Why don't you guys do this language because then you wouldn't have to write a manual?" We rejected that as being not a very good reason for picking a language. [Laughter] So, basically, I would say that the decision, that the compiler we would do would be ALGOL as opposed to FORTRAN was made by a summer employee and a project leader. I don't know that anyone else was really involved in making that decision.

GALLER: In defense of not having to write a manual, let me interject that with a manual, you have a specification of the language. I mean, we should all understand it's not writing a manual, just for the record.

DAHM: Sure. There existed a specification for IAL, too, in the sense of a document that was published in 1958 that described it. It turned out that there were some things that were almost impossible to implement.

GALLER: It hadn't been implemented and therefore...

ROSIN: What about the MCP in the same regard? You folks list it in this set of goals objectives that were hardly standard in the industry in those days--multiprogramming not the least among them. Did you talk with your customers about this? Were they interested? Were they resistant? Was there resistance inside the organization, or was it fairly smooth? Did anybody stand up and say, "That's a great goal, but we don't know how to do it; therefore, let's back off?"

DENT: I think that was true, but I'm not sure anybody said it.

KING: We saw the things like the 704s that were delivered. The 704 was delivered with 8K of memory, initially?

GALLER: 4K originally.

KING: 4K. Then what happened to them? These shops expanded them to 32K, and there were a lot of programs still running in 4K and they couldn't use the rest of the system. They had to get more production out, but they couldn't go back and rewrite the programs. So they sat there for a good portion of the day using a small portion of the system, and they couldn't do a damn thing about it! That was a crime. They couldn't expand in a real sense.

ROSIN: So, one of the goals, as I understand you for multiprogramming was to make more effective use of the hardware resources, not necessarily to provide a more flexible environment for the end user?

KING: Correct. Get more production.

GALLER: Lloyd?

TURNER: I have to say that I have trouble to buying all of this crap that's going on here. I mean it looks like there's nobody around that had any idea of what the hell we were going to do, and that just is not true. It is not true. In early 1960, all of these goals were out there. And we knew what the hell it was we had to do from the very beginning.

We didn't know how to do it all, but we knew exactly what we were going to do. It didn't have a damn thing to do with the market, it didn't have a damn thing to do with customers. They were not asked. If we had asked them, they would have said, "Hell, no. Give us FORTRAN." So that's all bullshit. You know?

GALLER: I don't think I heard anything else.

TURNER: We never asked the customer. We were building an ALGOL machine--pure and simple--that had all the storage allocation and all of the other things that we needed to make it a proper multiprocessor. From a functional standpoint, that was what we were after. And we knew all of that way up front.

GALLER: Did you expect your commercial customers to use ALGOL as well, then?

TURNER: No.

GALLER: Then how is this machine going to be sold to commercial customers?

TURNER: Well, we had a COBOL planned.

GALLER: Can I ask what the balance was between the emphasis on the two languages? How was that decided?

HALE: If you look at the machines, clearly it was an ALGOL machine. And it was a passing attempt to add some hardware facilities for COBOL which, to be generous, one could only say they did work--not very well, however. I'd like to add something to the question about goals with respect to how they were implemented. One of the big goals from my association was for the user to be able to write programs without having to be concerned with how much memory was on the machine. Initially, the plan was that we would be able to market machines with one module of memory, 4K words. Nobody really understood what the effect of overlaying from secondary storage was going to be. I recall in the summer of 1962 before we had a machine, we had a meeting which all the programming people--

Brad, Lloyd, Clark, and I--explained to the field technical people how they could run anything in one module of memory. Of course, that turned out to be grossly wrong. We really didn't understand the problem of overlays.

GALLER: What changes were made to try to accommodate COBOL better? You talked about hardware changes.

HALE: No, I didn't mean to imply there were any changes made in the initial design of the machine. Character mode, at least for my association, was always in the machine.

MACKENZIE: I think the character mode in a sense brings out one thing that you were getting at. I believe that showed up in the machine about the time the horrendous aspect of this being an ALGOL-only machine began to dawn on some people. Bill, I remember you went to Detroit with the proposal and kind of bounced a little bit on that one. We came back and we ended up with a character mode on the machine so it could now do COBOL and some things like that better. So that melding? was going on to try to make it, I think, more acceptable..

KREUDER: That really was pretty much of an add-on.

KING: It was in the 3000. The 3000 was a word machine and a character machine, and the 4000 was a word and character machine from day one. The problem is we didn't know what was required for COBOL. We just said, "A lot of COBOL appears to be manipulation of data, editing and such. We don't know what the hell we need for it." And I kept asking, and nobody could answer. Nobody knew what we need to have in a system to make it handle COBOL better. The only thing we could put in was a character mode. We said, "that will help, but we don't know."

MACDONALD: But the intent was to make it both; it's just that we were all too dumb to know how to do it right.

KING: Yes.

WAYCHOFF: Yes. On the selection of ALGOL: at that time, programming in higher-level languages all was still a very highly controversial subject. Most people were programming in assembly language. Burroughs decided that high-level languages were the way to go, that efficient computers could be written, and then we simply selected the best language around, and that was ALGOL. We didn't consider at all that there was the huge customer base in FORTRAN and the momentum that we'd have to overcome. ALGOL was simply a better language. So far as accommodating COBOL, when I joined the project, there were little ALGOL symbols in the character set for multiplication and the assignment operator, implication, equivalence--all those were in the character set. They all got wiped out in favor of COBOL characters such as dollar sign, percent sign, and some other things.

MACKENZIE: I think you should also remember that there were also uncertainties as to what the commercial, if you will, data processing language might become at the time, although the CODASYL activity had been organized, if I remember right, in 1958 or 1959. There was still a great deal of debate about COMTRAN, COBOL, things of that ilk; so there was that basis for uncertainty. I agree completely with what Lloyd was saying--that we were treating it too lightly. People did know what they were doing in that sense, and I think one of the things was you couldn't do with FORTRAN--or at least we thought people could not do with FORTRAN--the things that they wanted to do off that objective list that you were reading earlier because of implications of the language such as equivalence and things like that. I think that probably played a role in people's minds about the model language that they wanted to use.

MACDONALD: Stack control, for example, too.

MACKENZIE: Well, you'd have to ask Bob. I remember hearing Barton lecturing me on things like that at one time.

OLIPHINT: In terms of deciding about FORTRAN or ALGOL or this sort of thing, I think there was a general feeling, which probably turned out to be naive, that if we did a good system that was designed well and all fit together and accomplished these goals, it didn't make too much difference what the programming language was, that it would be a good enough system that it would be acceptable and sellable to people regardless of what the programming language was. That didn't really seem to be, from a customer acceptance point of view, a major consideration. Just that ALGOL fit with the design better, so it was the natural choice. It was a better choice since it fit better with the system goals and so on.

DAHM: I wasn't actually at Burroughs at the time the instruction set was designed, but my belief, at least, has always been that the instruction set was in fact designed to be the target of an ALGOL compiler. And that was a very, very revolutionary thing to do. In fact, if you look at almost all machines that are designed to do, their instruction sets are not really well designed from the point of view of being the target of a compiler. So I think it was an extraordinarily revolutionary thing today, and still, even today, has not really been adopted throughout the industry: the idea of picking a language and then designing a machine that is really well-suited for that language. In fact, that was kind of a Burroughs design philosophy for quite a number of years after this. The B 3500, for example, was a machine that was designed with the explicit idea of being the target of a COBOL compiler. It was a quite different architecture. Then later on Burroughs designed the B 1700, which was designed with the idea that one could change the instruction set depending upon what particular language the processor was processing at this moment. I think those are very, very revolutionary ideas, and in some sense, the industry as a whole has never really picked up on these ideas, which I think is a shame.

KING: Very early on there was a rule set that code would never be generated by human; code would only be generated by compiler, and we don't have to give any consideration whatsoever to make it easy for a programmer to generate machine language; it's all going to be done by compiler. Further, there was no consideration given to an operator in the sense of operating the machine like they did with the early 701s and 704s and the 205s and 220. It was always going to be operated under control of a control program. You couldn't run it without that control program. Running it under human control or running it with human-generated code were not design requirements.

MACDONALD: Backing up both of those, there was really a policy atmosphere set up so that the system would be software driven, not just in terms of compilation, but also in terms of MCP. The whole thing was going to be driven from software and from the application end. The hardware was intended to be a vehicle for both operating system and compilers--not totally realized, but remarkably well realized for that period of time, I think. Echoing what Dave said, I haven't seen it since. Maybe Lloyd's doing it at this point. I hope he is.

MACKENZIE: Somewhat facetiously, I'd say that the genius of that decision about compiler code only was one that had a very profound effect. Namely, it took the hardware engineering people in a sense out of the system business. Before that it didn't... A great deal of their domain, the machine language instruction set, the interaction even with customers on the machine language instruction set. So, in a sense, the higher-level system, if you will, became a domain of programming people rather than hardware engineering people. That was one of the more profound things that happened with that.

KREUDER: It got rid of one of the world's largest diameter headaches, namely, the design of the control console with all those flashing lights, and all that kind of crap, because that was the thing that every vice-president thought he knew something about. [Laughter]

MACDONALD: From the chairman of the board on down! I remember a red panic button that was there for Eppert.

DAHM: I think one of the things that was very revolutionary was the whole idea that the all of the software could also be written in a high-level language. I remember when Bob Barton told me that we were going to have do this ALGOL compiler in ALGOL, and I said, "That's crazy. You can't possibly do that. He said, "Why not?" I really could never give him a good answer why not, so we had to do it. [Laughter]

BARTON: May I comment on that? In the interest of history, I think that that wasn't a really revolutionary idea at that time. There was a group in IBM that I think Bob Bemer led that attempted to do that with--maybe it was IAL, I'm not sure. I don't think that was a revolutionary idea at all.

ROSIN: In a product, Bob? There were several people in the world who were using high level languages to process...

BARTON: I remember that my only capacity was as a consultant, so there are a lot of things I cannot comment on because I simply was not informed. But I'm just trying to say some of the things I know that are external to the whole thing. I think I am qualified to say something about those facts.

GALLER: Getting back to the decision making process. We heard a very strong statement that this was an ALGOL machine and then COBOL was brought in, etc. Some of you know that Paul Colen, who isn't here, sent me a statement about some of his ideas about what happened, etc. There's a phrase that he keeps quoting in there. He says a lot of the decisions were made because, "They wouldn't play in Peoria." Now, was that a phrase that was used at that time? And what did it mean?

LONERGAN: Never heard it before. I've heard the phrase, but not in that context.

KREUDER: It wasn't used then. He was referring to the idea that a delay-line memory wouldn't be salable. I think that's the context.

DENT: Many of the things that were talked about didn't play at all for a long time. [Laughter.] Trying to sell a machine where the operator or the user or the programmer didn't even know where his program was in memory was very difficult.

ROSIN: Let me follow up on that line of thinking, Ben. There was an early decision made, a positive decision, that this was going to be the ALGOL machine, as I understand from what you all have said. There was a decision made sometime afterward that there would be a strong effort als o to accommodate COBOL. FORTRAN was clearly excluded. But at some time later in the history of this machine, there was a decision made that there would be a FORTRAN processor of some kind or another. Why was that decision made? And would the architecture of the B 5000 have been significantly different had FORTRAN been included from the beginning?

KREUDER: I can tell you what the decision was made and that was in Paul Leebrick's hotel room at an NCC - Brad was there. Leebrick gathered us all together, and he said, "Men, we've got to have a FORTRAN compiler." I was full of about 50 jillion reasons why it couldn't be done and Brad interrupted me and said, "It's no biggie. [Laughter.] How soon do you have to have it?" [Laughter.] Then later, typical Brad says, "By the way, Paul, which particular FORTRAN are you interested in?" [Laughter.]

MACKENZIE: The FORTRAN translator had apparently originally been planned in the project to be written in ALGOL to translate FORTRAN to ALGOL. Fran Crowder was working on it in Product Planning. Lloyd, your section had it if you remember. the event that Norm's talking about was probably 1965, about then. And Joe Hootman and people this afternoon will tell you there was a tremendous frustration in the marketing organization because of the lack of FORTRAN. I think in a sense it was a conscious decision on the part of people not to have FORTRAN because of what the implications of having it would mean in terms of the way it would be sold and potentially compromising the system. I know if I said such a thing as Norm was saying, it's because Lloyd Turner told me at that time that it could be done. [Laughter.] No, seriously. And it turned out that it was not a big or a difficult thing to do at that point in time. I know in a strictly personal sense--right, wrong, or indifferent--I never wanted to see it on there because to me it had implications of trying to sell a machine head-on in FORTRAN shops, which I personally thought would be a terrible mistake and not the kind of thing to be trying to do with this wonderful machine called the 5000, which had a lot of other objectives.

TURNER: Richard and I had just come from the field and we had not done a FORTRAN compiler. In fact, my arithmetic algorithm that I used was yours, Bernie.

GALLER: Thank you.

TURNER: And so, of course, we knew all the ridiculous storage allocation constraint that one does in a FORTRAN, in implementing FORTRAN with the static storage allocation. And that was diametrically opposed to one of the basic design criteria of the 5000. It was not going to run well, no matter what you did to it. No matter how much you tried to optimize it, it was not going to run well. We tried our best to discourage writing a compiler, so we had to say, "Well, we'll give you something that will get you by and give you a translator." It worked reasonably well for a translator with all its inefficiencies. And then later on, we actually did the compiler after I left. I went away to the wonders of Product Management, but I have to say that FORTRAN, even on the 6000 series which followed, the FORTRAN performance was always, and always had to be, was going to be crappy. Because it just was built to do something different. In fact, it wasn't that it wasn't designed to do that; the design was at odds with running FORTRAN. So, there is a difference.

CREECH: I know for myself, and I believe for the people I associated with there was a sincere--although in hindsight naive--belief that ALGOL was such a superior language, it would take the world by storm.

OLIPHINT: Oh, absolutely.

KING: Yes.

CREECH: We believed that. We were wrong.

ROSIN: But I want to ask a question about ALGOL. There's constant reference to ALGOL in this discussion. Wasn't there a significant different between the Burroughs 5000 ALGOL and ALGOL 60, which was accepted to be the standard?

MACKENZIE: No. [General consensus in background.]

DAHM: Basically, it was a superset. ALGOL was a superset of the ALGOL 60 report.

ROSIN: What were the differences? Which ones were important?

CREECH: We had I/O! [Laughter]

ROSIN: What about generating...? There clearly had to be a way to generate machine instructions in this language and as I recall ALGOL 60...

TURNER: No. Absolutely not.

OLIPHINT: There was a string procedure added to handle the character mode kind of operation generally associated with I/O, I think, if I recall.

DENT: Format.

OLIPHINT: But there was no way to generate other than that. No way to generate machine instructions. That was one of the basic...

ROSIN: Excuse me. I didn't mean to generate machine instructions included in the source program. I meant to generate a machine language version of the original program required the ability in ALGOL, I would assume, to manipulate fields within a word.

DAHM: Yes. That was an extension we made. The capability of referring to a partial field of a word.

TURNER: It's called Extended ALGOL in that sense.

GALLER: Yes.

WAYCHOFF: We went to great lengths to implement ALGOL 60 as published. We did make a lot of extensions. I remember Lloyd resisted practically every extension unless there was a very good reason for it, and I learned something from him on that point. But, even to the extent of the evaluation of a "FOR" statement, the control variables, if the expressions in a "FOR" statement were expressions that had side effects, we would get exactly the answers that were intended by published ALGOL 60 because of our evaluation sequence.

GALLER: What about ESPOL? What made that different from ALGOL? What was it?

TURNER: Well, after we finished like our third version of the compiler, we had it so that it would compile any program any size into a memory module, by the way. We had a disk file coming which was a better backup device than the drums, and so we had to do another operating system. So we embarked on the disk file MCP. In three weeks, we took the ALGOL compiler that we had, which was extended ALGOL, and made an implementation language out of it by taking out the storage allocation. Because there was no operating system to do storage allocation. There weren't many additions to make. The whole thing got done in three weeks.

DAHM: Probably the most significant addition to ALGOL to make ESPOL was the ability to address all of memory as an array, which was necessary in order for the operating system to be able to do storage allocation. But other than that, it was just basically just the same as the extended ALGOL that we used for writing the compilers in, for example.

MACKENZIE: One thing to remember, the drum MCP, that Clark did was not written in a higher-level language. It was written in a thing called OSIL which was a pretty primitive assembler.

GALLER: I was about to ask about OSIL. We were always told there was no assembler on this machine.

OLIPHINT: There was.

MACKENZIE: We tried hard to keep it a secret. [Laughter]

CREECH: The assembler ran on the 220. It did not run on this machine. There wasn't an assembler on this machine.

GALLER: OSIL ran on the 220?

OLIPHINT: Yes.

ROSIN: Were later versions of MCP then written in ALGOL?

TURNER: Yes, and ESPOL.

KING: The disk filer in MCP was written in ESPOL.

TURNER: After that, there was no more assembler.

DENT: May I suggest that you're skipping ahead a little bit. You ought to hear a little bit about the original drum MCP, which wasn't a particularly successful project, but I think we learned a lot doing it. And that's what really brought the 5500 into being. But I think you can establish what was wrong with the drum MCP, what was wrong with the machine, and what things we had to fix; we did that as a second pass, and it was renamed for commercial reasons.

GALLER: All right. We might as well do that now. We would get to that later in time, but fine, let's do that.

DENT: Clark ought to do that.

OLIPHINT: As we've said earlier, there was always the idea from the beginning that there would be an operating system to run the machine, that it wouldn't be the stand-alone kind of thing that depended on operators. I can remember in Product Planning, in the early days when we were starting to decide how to do things and whether to do something in hardware or software, a saying that it got pretty well widespread: "Oh, the MCP will take care of that." [Laughter.] As we thought of more and more things that needed to be done--how to do this, how to do that--that was the answer to many, many things.

MACKENZIE: You used to even say that before you were doing the MCP!

OLIPHINT: That's right. Little did I know...

DENT: You knew you were going to have to do it!

OLIPHINT: Little did I know that when the MCP was going to do it, I was going to have to figure out a way. Then I guess the time really came when we had to start sitting down and figure out how the MCP was going to do those things that it had to do. That's all a long time ago; it's hard to remember how that all happened.

GALLER: What began to go wrong?

HALE: Nothing worked.

GALLER: Well, they worked in the sense, I presume, that the code was checked out but the performance wasn't.

OLIPHINT: But one of the things that happened was, we fell behind in doing the MCP. The tape assignment was one of the things the MCP was going to do. You didn't have to decide which unit a tape got mounted on. You put a tape on, and the system recognized automatically what the label was, and so on. But in the early days, we didn't have all that working, and so you had to put a particular thing on tape unit E, I seem to recall. You had to have a tape unit E...

MACDONALD: Which typically didn't work.

OLIPHINT: Yes, right. So we had to make compromises in doing a sort of a interim version of the MCP that didn't do multiprogramming, that didn't do many of the things it was supposed to do because time was getting close and there just were a lot of things not working. I remember being horrified in the context we were in, where we had all talked about and agreed that this system wouldn't work without software, to suddenly find out one day that a 5000 was going to be shipped. I couldn't believe it! I don't know how, when, or where that happened, but the stuff wasn't ready. The software wasn't ready, there was no way...

MACKENZIE: Some things never change. [Laughter.]

OLIPHINT: There was no way that anybody would be happy with the product. And I really couldn't believe, I remember at the time, that we were really going to ship one without the software because all our advertising, everything anybody knew was that it wouldn't work without software. I believe it was something on the order of 6 months to a year after first shipping of product, before we got an MCP that would do a lot of the things we talked about: multiprogramming, automatic tape assignment. Am I wrong on the dates there?

MACKENZIE: You're right in the general sense of things easing in, but the first shipment was on April Fool's Day, 1963, as Clark points out. The ALGOL compiler was operational and there was obviously enough and perhaps a lot of errors on it but enough MCP to support the ALGOL compiler operation. It turned out that the second system that went to NASA was, to my knowledge, the only system that did not have software requirements written into the contract. It was the standard contract of the type I suppose that the government was issuing. John, it seems to me that the first release of the COBOL compiler was September or October of 1963.

HALE: Yes, I think it was October of 1963.

MACKENZIE: During this time more and more MCP capability was becoming available. Sometime during that interim, and I think it was much earlier than October, some kind of multiprocessing, or multiprogramming capability as we called it then, began to operate. By the time the Marathon system got installed in October or November, it was a reasonably good working version of COBOL. It was improved tremendously after that, but that release had been made in October. So things were gradually picking up. By the end of the year, there were several systems that were on rental, perhaps customers were a little bit unhappy about paying rent and stuff like that. NASA had been the first one that actually went on rental.

DENT: I think there were several things that caused this. The first was, our tools were really quite poor. We were doing things in assembly language. In spite of all the other wonderful things we'd heard about high-level language, the operating system people were doing things in assembly language on a previous machine to help the bootstrap process, and nobody had spent a lot of time with the assembler in the first place. The job of the operating system was very badly underestimated, and that's partly because of this tendency to say, "Oh well, the MCP will do it." My remembrance when I got there was that I could not find a list of things that the MCP was supposed to do. Everybody had their own idea of what was supposed to happen, but we didn't even know how big the job was because we didn't have a list.

GALLER: There was no formal spec all for the MCP?

DENT: I don't remember there being one. I remember being very envious of the compiler group, because just by the definition of the ALGOL language, they had a spec for what they had to do. You could almost go through the pages and count the syntactic definitions and see how many subroutines you had to have. We had no such thing at all. There were some other problems. Most of the people working on it had not learned to deal with the concept of multiprogramming, or the concept of loss of control. So there were lots of bugs where somebody would save off a variable, go off to do something, like perhaps get a little more storage, not realizing that that might amount to loss of control. In order to get storage, if you were to guess somebody would have to do an overlay, and while that I/O was going on, some other job would come in and you'd do something, and then come back and store the variable. You have to learn to think that way before you can write code that has a hope of working. And we weren't thinking that way. There were some other rather basic problems with the machine. Of course, we already alluded to the underestimation of the memory size required. I think the original spec was the operating system was supposed to run in 512 words or something, which turned out to be wrong by...

MACKENZIE: Well, the whole thing was supposed to go in 2000 words, compilers and operating system. The other half could be used for programs.

OLIPHINT: The compilers all got too big. [Laughter.]

DENT: By the time we finally got all this automatic overlaying thing to work, it became obvious that there wasn't enough room on secondary storage to overlay things. I mean we literally had some job situations that ran out of memory--where the machine stopped, it said, "I'm sorry; I'm out of memory, and I don't know what to do about it." There were lots of things in memory that could be overlayed, but no place to put it. We really didn't have a good mass storage device. Brad alluded to the business of this mass file being a new concept, like you could leave your programs on the machine. I can remember before that if you were a programmer and you went to use the machine, you got the card deck out of your desk, you went in, you ran on the machine, and when you were finished, you took it back lest it be thrown into the garbage can. The concept of leaving anything in the machine for the next user that came along... But on this first drum MCP, we stored programs on library tapes, even parts of programs and things that had been compiled, jobs that until we ran them were stored on tape, which is not a very good device to do that sort of thing. In fact, we wore out a few tape units. We had to keep the directories on the tapes themselves, and if you remember the IBM format, you couldn't overwrite. So it was a little tricky trying to update the directory on the tape. There were these kinds of problems that I think we realized and we fixed, and it became to some extent part of the definition of the 5500. Not the least of which was saying, "Look, we've got to apply some of this compiler technology to help ourselves write the operating system."

TAPE 2/SIDE 1

GALLER: We have put some of the planning into context. I'd like to get more now to the specific innovations and ideas that went into this machine, the background of some of those ideas, and how they came together. Let me read a quick list I put together of some of the innovative things. We may want to add to them, just for the record, but then we might start back as to how it all got going here and how it happened to become what it was. Items, for example, that I think were on the edge of what was going on, the innovations, however you want to describe them. Of course we talk about the stack concept, descriptors of segmentation, the idea of high-level language only, all of the things that we've talked about with the MCP, modularity, and the more than one processor aspect, etc.

TURNER: And no SYSGEN!

MACKENZIE: You bet.

GALLER: No SYSGEN; fine. Are there any other strong aspects that ought to be on a list of innovations? There were many others, I know.

??   Dynamic storage allocation.

TURNER: Compilation technique and recursive descent.

CREECH: Programming development technology.

GALLER: When I said MCP, of course, I swept a lot of things in there, clearly.

MACKENZIE: Well, I think programming system technology is probably a better term, then.

GALLER: Let's start to get at how some of these ideas began to come in.

TURNER: There's one other that I think ought to be entered in addition. To I think specifically that something ought to be said about the road map that we did.

WAYCHOFF: The chant for meta-languages.

TURNER: The syntax chart for converting back to ?, which they're still using today in some form, even in PASCAL.

ROSIN: What inspired that, and who did it?

TURNER: The flowcharts that we did of each one of the algorithms to take each one of the syntax elements. We took them back to ?, and we would draw those little flow diagrams. We had several of those, and, of course, the decision ones were elliptical just like the old flowchart conventions that we learned using IBM machines. When there was a choice there was an ellipse into the square whenever you wanted to go to a description of something. So you put all of that together and that's what you get. And Richard and I worked on that in 1961.

WAYCHOFF: I remember that quite well. Lloyd and I were working together, Bob was spending some time with us also at the very beginning of the ALGOL project, and I was thumbing back and forth through my ALGOL, published ALGOL, from the @Communications of the ACM, time after time and wearing out copy after copy of it looking at the meta-linguistic definitions. I said, "What we need here is some thing like a flowchart to describe this." At the time, Bob Barton said, "Well, this is a recursive language." I remember he drew a square on the blackboard with lines leading out and then back in again saying that's all it would show. So I dropped it, and a couple of days later I brought up the same thing again. Then Lloyd immediately saw what I was trying to get at, and he came up with the idea of two different shapes; we called them pickles and boxes where the meta-linguistic variable is defined here or the meta-linguistic variable is only used here and defined somewhere else. That, I think, was one of the most significant contributions that we made on this. Because by staring at that, then the notion of recursive descent compilation becomes quite obvious.

GALLER: What about prior work in this area?

KING: Well, Ned Irons had been out to visit us and he had a recursive ascent that went up to the leaves and headed back to the trunk, and that seemed like a terribly pedestrian way to do things to me. We had all these nice flowcharts to do it the other way and it was more natural, it seemed to me, to start at the trunk and make the decision and then go on down from there. We didn't call it that. We didn't know what the hell to call it. We knew that the idea that we had built into recursive procedures and that the language was recursive, that there was certainly a connection there that just couldn't be overlooked. That's sort of how we decided to do it that way. And that's exactly the way the compiler was built. If you look at that funny road map of the extended ALGOL language, that was very close to an overall flowchart of the entire syntax of the compiler.

GALLER: I'm glad you mentioned Irons because I think at the time he was one of the first to begin to formalize the process of doing it. You disagreed with how he did it, but...

TURNER: Yes. His compilation technique turned out to be quite different from ours. But, certainly in terms of writing the compiler and looking at how he did things, it helped us make some decisions on how to implement things. We had a lot easier job than he had; trying to implement all of the procedures that are in ALGOL without any (stacks?) would have been very difficult. Most of the people in Europe that wrote the compilers had to simulate the stack.

GALLER: Let's talk about the stack a little bit. How did that come in?

KING: When we started out, Bob Barton had this idea of doing a system oriented around Polish notation, using a very condensed form of addressing and a rather elaborate indirect addressing scheme. When you use Polish notation, it implies a stack. So we started out with that. We also then had a second stack. I was very interested in a system called PERM that had a means of subroutine control that was recursive and being able to pass the parameters and control the passing of parameters recursively. So we ended up putting in a second stack for that. So we had two stacks, one for a data stack and one for this control stack. That went on for a while; it was sort of clumsy. Let me back up a little bit. The first stack, as originally proposed, was going to be a set of registers. But that created certain problems because if you have a set of registers, when you change context, you've got to dump all those registers. We worked on that, and came up in one session with what we called the deep stack. The deep stack said, "To hell with all those registers; we'll have two registers that are going to be the arithmetic registers, and then everything else is in memory." So we solved what we called the deep stack problem. We still had the two stacks, however. We still had the data stack and the control stack. It was Jack Merner who brought those together and said, "Why have two stacks? Why don't you just use them together." And that's really what came about in the long run.

ROSIN: Can we go back a little further on both of those stacks? I guess we'd like to know more about the concept of a stack for arithmetic processing and also this PERM system that you mentioned, which I think we ought to know more about. Where did they come from?

KING: I don't know where Bob got the Polish notation. Did it derive from the BALGOL compiler?

?? It came out of their language.

DAHM: No, it preceded the BALGOL compiler.

ROSIN: Bob, do you want to recollect?

BARTON: Somehow I don't think it's appropriate for me to say anything at this point. I want to know what other people think happened. I was just a consultant, and I can't say what in the world some of the people that were participating with line responsibility were doing in their room within a room within a box. I mean, there was a lot of stuff going on that I knew nothing whatever about. As a matter of fact, it didn't take very long before I was not terribly welcome most of the time. I had to go away fairly frequently because I was overstepping my bounds as a consultant. I tended to do that. Bill will remember when one fellow came into his office and said, "Can Bob Barton fire me?" And Bill, I think, laughed and said, "No, he's a consultant." And this guy knew. So, I don't know what they did with whatever my contributions were. I don't even know what my contributions were anymore. I only have a few things I can hang onto. They sure did pay me well. They really did pay me well. I think some of the people here would be very annoyed if they knew just how much I was paid. [Laughter.] And as an unknown guy who came from...

LONERGAN: I think you were a bargain.

GALLER: Bob, maybe if you don't feel comfortable, Dave could say something, but make sure that what's said is correct from your point of view.

BARTON: Well, I'm not sure that I can say anything along those lines except to tell in retrospect what I learned in later years about the genesis of some of the ideas. I can trace every idea back to--not quite to Babbage, but close. And that's a fact.

GALLER: Sure. I think most ideas can be traced to him.

BARTON: But I don't know the times about the genesis of ideas because I was, for instance, very poorly informed about what was going on in Europe. I was pretty well-informed about what was going on in the United States. For instance, I was telling a couple of people last night that Bernie would be surprised, but he had a lot of funny indirect influence on what happened. You may remember my early visits to Michigan while you and Bruce Arden were working on what became MAD. I was learning a lot on those visits about what was possible. Not necessarily how to do it, but what was possible. My view of what happened is so different from other people's reality that I really do have to disqualify myself as a reasonable witness. You asked me to try to set the record straight if possible. I tried that once before. I disagreed with the 5000 programming effort as being the first instance of an attempt to develop systems in a higher-level language. I mean, that's a fact. I could say a lot more about why I think it was successful, whereas the other effort was not successful. I could quote the late Ascher Opler's opinion on that. There's a lot of that kind of thing, but we haven't got several weeks. I would just love to stay as much in the background as possible and find out what other people thought happened. Because I really don't know anymore.

DAHM: I think I remember pretty clearly. In the summer of 1958, before Bob Barton went to Burroughs, he worked at Shell, and I worked for him that summer; that was my first involvement with computers. I remember at that time, him describing to me this really neat little algorithm he had for turning an arithmetic expression into machine code and how it was all inspired by Lukaszewicz notation, which he had run across in reading a book on symbolic logic. He also described to me how one could possibly build a machine with a built-in stack to handle this in a more automatic fashion. Now, that was before we were ever at Burroughs. I remember that clearly in 1958.

BARTON: I think that's essentially correct. And, incidentally, it ties into Michigan again. (I saw Paul shake his head.)

KING: No, I absolutely agree with that.

BARTON: Oh, O.K. Tying it into Michigan, it was Kopi's book, I don't know which edition. I was getting ready to go to a Michigan summer conference, and I remember the little program said, "Some knowledge of symbolic logic is helpful though not essential." So I went over to our library, and I reached on the shelf and there was Kopi's symbolic logic. I often read books in funny order; I sometimes turn to the index. In the index my eye fell on Polish notation, I thought "What in the world is that?" I turned to that page and there was a description relative to logic, and in the next 30 seconds, I saw it was a simple way of translating arithmetic expressions. I'd seen the article by Backus, and I thought it was unintelligible, by the way; too involved. I saw, I think next, the idea of handling iterations that way, which, by the way, was not done in that machine. I saw the use for interrupts, which we were kind of conscious of in our context at Shell. By the way, of course, the machine could have been used for interrupts that way, but wasn't; everybody overlooked it. Bob Creech remembers learning about this in the early days of the 6500. There was something there that people didn't fully appreciate. Also for subroutines. The business of different kinds of stack linkages and combinations of stacks, I honestly don't know what the history of that was. I think there was a lot of doodling done by a lot of people. We were well acquainted with using link lists as a storage allocation means at Shell. We were kind of well informed there. And we... Well, who was the "we"? Well, there was this guy, Joel Erdwin, who was responsible for the BALGOL compiler, which I'd like to get on the record was a brilliant job of programming. Dave Dahm has already told you that he was involved in the beginning of that during the thinking stage. It should be mentioned that Merner was involved in it all the way through. My opinion is that it was Erdwin's masterpiece.

I want to correct Dave Dahm's statement about my trying to get them to do FORTRAN. [Laughter] It's correct to a certain extent in that the job that I had taken, under generally misleading conditions, called for doing an impossible FORTRAN which would also include conversion of assembly language from the 7090, or whatever the machine was at the time, automatically. I knew it couldn't be done, but that was my responsibility. Erdwin would never have done a FORTRAN. I mean, he'd been going through this kind of educational experience at Shell, and he was not the sort of guy that would waste his time doing FORTRAN. He knew too much about language. It's sad to say, though, that after Erdwin did the BALGOL compiler, he went off to Computer Sciences and spent the whole rest of his working career either implementing FORTRANs or teaching other people how to do it. I think it was 20 years worth. So the punishment does fit the crime. [Laughter] Now, I'm talking too much. I've absolutely got to shut up. I can't do this.

GALLER: Let me interject an interesting thing here. Several people have donated documents here which I hope to circulate for your nostalgia and to copy for the archives. I'm looking here at an excerpt from the November 1962 Data Processing which reminds me that Burroughs built a truth function evaluating machine with some people from the University of Michigan before I joined them.

BARTON: That's right. That's something else I learned about later.

GALLER: It was a truth function evaluator with a built-in stack. In fact, just as a personal note, it was the article I read about that machine that got me into computing, so I appreciate it very much. But there was a history now in Burroughs of some of these ideas. That's very interesting.

KING: But that idea never crossed over from Paoli to Pasadena.

BARTON: That, unfortunately, was due to this one trip to Paoli. I just thought of this morning as a way of characterizing what I've been doing for 30-some years. I'm an industrial saboteur. I often do not discover whether or not the sabotage was effective until much later. It may be that I drop ideas carelessly. Under certain conditions, they fall on fertile ground. Now, during that whole period that I was consulting for Bill Lonergan's group, as best I can remember, I was given one direction. He asked me politely, I must say. It turned out that Erdwin was going back to Paoli, but due to some family difficulty, he couldn't go. Bill said to me, "Would you mind going back to Paoli?" I'd never been there. I was agreeable to going. I was asked to sit in on a meeting (about this number of people), and they were talking about this machine called the 825. Someone in that room said something that just drove me up the wall, and I made a wise remark. The guy defended himself by saying, "Well, how would you do it, smart guy?" I walked up to the blackboard, and I took the idea of Polish combined with a three-address machine, and gave him a "for instance." I learned somewhat to my annoyance later on, although it annoyed other people a lot more than it annoyed me, that the engineers went off and implemented this funny application of a kind of a three-address variant on Polish. Which, incidentally, was reinvented years later by the Intel people and the ill-famed 4-... whatever it is, Justin Retmer's thing. They used a very similar scheme. Anyway, that was a throwaway; I would call it a piece of industrial sabotage. One guy named Jim Anderson had a lot to do with that machine; he happened to be on vacation during this ill-fated meeting. So I'm afraid I do have to admit that that is the one effect I had on the 825. I'm not proud of it. I didn't mean it to happen. It was a piece of industrial sabotage. However, there's an interesting psychological aspect to it in Barton's view of the world. Before that time, I had not had much connection with the Product Planning people. I think once they invited me over. I was coming from the outside world, and maybe users of the machine would know something, so they wanted to know what I thought about the machine. I remember saying something about my ideas on machines at the time, and I remember generally a kind of devil's advocate response. I have never been able to work with devil's advocates. So the importance of the Paoli thing was this: When I came back from Paoli, I was reminded of the Polish business, and I came back and I remember - I think it's a fairly accurate memory - Paul King and probably a couple or three other people were in a room, and I think I told them about Polish and they were very responsive. I remember that. And I think, in a sense, that had a kind of a motivating influence on the whole thing. Earlier you commented on the state of the morale at the time. As perceived by me, morale was lousy. It followed a period of layoffs before I arrived on the scene. They had to cancel a couple of projects, one of which has been mentioned; there was another one that was a kind of a tape-driven printer -- possibly something else, I don't know.

MACKENZIE: Unfortunately, that didn't get canned. [Laughter.]

BARTON: What I'm saying is that there was catalytic effect that came about purely incidentally because of the accident of Erdwin's being sick and your asking me to go back to Paoli and my opening my big mouth at that meeting and then getting sort of enthused about the whole thing and coming back and perhaps offering something a little novel in the context of the time. Now, the history of Polish notation, which I learned a lot about later on; I know all sorts of people who invented it, all sorts of people who applied it. I think just one thing that can be said about it that's really significant is said by the mathematician Karl Menger. In a book of bits and pieces he had written -- miscellaneous collected papers -- he was struck, after the Hewlett-Packard Polish calculator came on the scene, by what a clever thing it was, and he remembered a trip he made to Poland in the 1920s when he learned about that notation. This is what he said about it: to the best of his knowledge, no European mathematician had ever seen such a notation. And he thought it was an incredible invention on Lukaszewicz's part in logic. And then he said, "But for arithmetic expressions, it's almost obvious, and in fact it's built into the major Western languages." Consider a phrase like, "Take the product of the sum of A and B and so on." So, as Menger looked at it, it was an obvious sort of invention for the way we used it. I was very interested in discovering that because it made me feel less smart when I read Kopi's book and saw the... But, of course, that was in logic, too. Maybe I'll get a little credit for that, but not much. Anyway, that's the application of an idea. It's got an interesting history. I think there's been an enormous amount of reconstruction after the fact. For instance, was this really an ALGOL machine? What I believe was that it turned out to be a machine that by several measures was very good for implementing ALGOL. I'm thinking of the studies that were done at Cambridge. I can't remember the guy's name. He did studies on a whole series of machines and ALGOL implementations at the time. And he's quite perceptive. For instance, one of the things he said was, "The Burroughs B 5500 is interesting because the code is so compact. Somebody must have had that in mind." Well, during that period, I don't think most people who talked about the machine to the public, and so on, knew that anyone had that in mind. But, in those days, it was a very natural thing to have in mind, particularly if, as was the case with me, I was interested in small machines. In my view, the machine got too big too soon. All the experience we had from working with very small machines, particularly the 650, only indirectly with 205s, was in the direction of simplicity and operating systems and engineering out as much as possible all the incidental red tape that was very common in those days. One of the guys that worked with us at Shell was named Bill Gant, who had an enormous influence on my ideas on operating systems. Bill was not satisfied until (?) could be met pushing no more than one button no more than once. And we had rather remarkable systems which he was responsible for on the 650. Similar systems were done for the 205 in another one of our locations that were really easy to use. Years later people would come by who had experience using those systems and went to other places with big machines and discovered that computing was hard. Just getting to machines was hard. So I think that one of the things that the Shell people brought in was a standard for operating systems which was not based on experience with the kind of operating systems that then existed, but was rather based on card systems of that time. And this crazy console with one button, that's Bill Gant. I don't know why people were so fascinated with that idea, but it was sort of a funny thing to have this console and then have the one button. It was very odd in those days.

There's something else about the disk. It could have made sense, if the machine were very small, to have only had a 32K drum on it. The pressures to make it a big machine tended to mean that the storage sizes moved up very rapidly. And then the drums really made no sense at all. Why were they used? They were totally out of balance. They were used because they'd already been designed and they happened to be there. I think they even had some drums in inventory.

KING: No.

BARTON: They didn't have them in inventory?

KING: It was the largest storage device available.

KREUDER: It was designed specifically for us.

KING: They were looking for larger storage and it didn't exist. There's disagreement here, Bob.

BARTON: All I will say is this; it made no sense to put those drums on the machine -- no sense at all. That's one of the reasons why the machine that was called the 5000 was not really a full implementation of the original concept. Two points: one, the machine needed the disk, they got a reasonable ratio of secondary to primary storage size. Second, it needed the operating system implemented in a higher-level language to fulfill that principle. There really was a principle which people bought rather readily - the idea of using a higher-level language for implementation. We made a stab at it a year before, but the notorious Erdwin refused. He didn't want to try it at the time. He was right. I'm always pushing people a little bit early. Clark had a very hard job to do with an assembly program. It was a violation of principle. There wasn't anybody in the United States at that time, as far as I know, that really believed you could do an operating system in a higher-level language. So, it's fair to say that for the ultimate disk MCP to be the success it was, somebody had to do something first and they had to learn a lot. I have a very big disagreement with Bill Lonergan. He's put it in print, and he said it again today, and I'm sure he believes it. He said, "There were a lot of smart people there at that time." My honest opinion, after thinking about it off and on and getting involved in similar things several times since, is that there weren't any smart people. They were just all ordinary people, just a cross section of ordinary people. Now, I would argue exceptions one at a time. I also think that almost everybody, perhaps everyone, really didn't know what they were doing. They didn't know how good it was, for instance. And when I say everyone, I'm including myself. I was scared stiff. How in the heck did I ever get involved in that kind of thing? I don't know. I can't even explain why I'm still involved in that kind of thing after all these years. As best I can remember, there was really no recognition of what was good about that machine for at least eight years. I'm talking about in the field in general. I'm sure there were users. I suspect Henri Berce was one of them because I know something about how he relates to the machines. He probably thought there were a lot of good things about it. I'm sure there were users who appreciated at least some aspect of it. But as far as the field was concerned, it wasn't ready. It was a little bit ahead of the computer science of the time, which really hadn't quite been defined. But does anybody deserve any credit for foresight and all that? I don't really think so. I mean, all you can say is that everybody did the best they could possibly do. I was the best possible Bob Barton. Paul King was the best possible Paul King (we didn't get along by the way, not at all). Bill Lonergan was the best possible Bill Lonergan. He perceived that I was unmanageable, he put that in writing, too. He said, "Probably this is the best way for him to relate to the world. He's not a good employee type." That was true. It wasn't a fun-type experience. I hated the whole thing. I hated to come to this meeting, I hated to see some of you again, although this turned not to be as bad as I expected [some laughter], I still think you can't do what you're trying to do. I think our memories are faulty, our perceptions are enormously varied.

KREUDER: That's never stopped us before, why should it stop us now? [Laughter.]

BARTON: Of course, there are several B 5000s. My B 5000 exists in a platonic universe where there are also perfect tetrahedrons and absolutely smooth stairs. Norm's B 5000 exists in a world of hardware. Bill Lonergan's B 5000 exists in the world of talking a bunch of old-fashioned executives into doing a crazy thing --and it was a crazy thing. Now, we could play a game where we take each person involved out and imagine if it could still have happened. I want to stop talking, but I do want to say one thing, and this is the main reason I came. The unsung heroes of this thing who really haven't ever gotten much recognition is the group of amateurs who worked on the ALGOL implementation language -- substitute for assembly language (which of course it was) -- and the final MCP, when they had all that was necessary to do a good MCP: implementation language, the experience of the first group, the big disk, all of those things. A very small group of people -- I don't know how many at average, maybe five --over short periods of time, did some of the most innovative programming ever done. Now, they were all amateurs, with almost no qualifications at all. Some of them had some training at Shell, but I've already told you that was restricted to a mundane punched-card 650. Without the experience, without the professional qualifications, they did two of the best systems ever done. The BALGOL which was done Erdwin, Merner, with Dahm's help in the planning stage, was another one of the best systems ever done. They were way ahead of their time, and I think one of the things that makes them so great is that these guys were not particularly qualified to do it. My particular hero for that whole thing, who made me believe (because I wasn't really a believer at the end of that period that the 5000 would come to anything) that it was good for something, was Lloyd Turner. Lloyd Turner, he's not the smartest guy in the world, but he's one of the most tenacious people around. He enslaved the people that worked for him. He absolutely enslaved them and he got away with it. [Laughter] Families were broken up; it was not only the time spent, but the R&R that went along with it -- the heavy drinking, and all bad. It was a terrible experience, but if there are to be any recognized people in this whole thing that need recognition, it's the guys that wrote the program. Now, they might not have been successful if the machine hadn't helped them somewhat. That's what Ascher Opler thought. But this is the one thing that can be accomplished by this meeting I think: is to put the credit where it belongs, and that's where the credit belongs. Ideas are cheap, they're a dime a dozen. Anybody need any ideas? I've got thousands of them, supercomputers, everything, the works. But somehow, things have to become real. And it's partly a matter of iron and it's partly a matter of scribe work. There's the iron man, there's the scribe. Chief scribe. Brainy people. Dave Dahm's a smart mathematician, he's a failed mathematician, of course, he's been a coder for years. [Laughter] Merner is smart in lots of ways and he's a failed physicist. We're all some sort of failures.

GALLER: I guess I'll disagree with you just a bit, Bob, and then maybe we should give some other people a chance.

BARTON: You'll never convince me, though.

GALLER: I think that the people in these groups were qualified as well as anyone was in those days. I think they were smart, or are smart, as well as anybody involved in those days. I think there was a lot of innovation, and I hope that this is the process of giving credit to the group that's here and to the people who aren't here who were in these groups. That's what we're here for, at least in part.

BARTON: The people who helped that didn't know they were helping.

GALLER: Sure.

BARTON: We should mention Ted Glaser. Ted Glaser somehow had a great deal to do with getting me interested in the subject way back in IBM days, and Ted's an expert on visualization. I learned from Ted that if you've got some new ideas, don't pin them down too fast. Keep them sort of loose in your head. The minute you touch paper, they start turning into specifications. Specifications can turn into iron, and if there's a mistake conceptually, it will go all the way, and then it will become something you have to be compatible with later on. Land of Polaroid said that you should never treat new ideas with a devil's advocate method. You should nourish them, you should coddle them. Then later on you can come in and shoot holes in them. But while they're new and fragile you must not do that. That was my big disagreement with those guys in those days.

LONERGAN: Why don't we get on back with the meeting.

BARTON: Oh, isn't that the meeting? I've been quiet all morning. So far I haven't said more than Bill Lonergan said all together.

LONERGAN: You had a list of things you wanted to go over, I thought.

ROSIN: I want to get back to the control stack, which we haven't quite finished. Paul mentioned that there was something called PERM, and it would be helpful if we knew more about that, I think.

KING: In getting ready for this, I went back and sort of did a time line on what happened in the time from the death of the 2111 until the announcement of the B 5000 and what happened at various points in time. I mentioned a B 3000. There was a small group of us working on the thing called the B 3000 after the 2111 died. It was a word machine and a character machine. It had built-in subroutine control of the PERM. But it still hadn't taken form as to what it was going to be in terms of its machine language. The 2111 was a syllable machine. It was a three-address syllable machine that was sort of unusual. We were looking at that, and at about that time we were putting in some of the concepts that had developed on modularity out of the meetings that we had with MacDonald and Glaser and Lonergan. Then I went off to a conference at UCLA and was exposed to a rather lengthy presentation on the Atlas, Stretch, LARC, etc., etc., the G-20. There were a number of ideas that came out of that then played into what became the 4000. Because then Barton got involved, and the ideas that he brought on Polish notation and stack blended in very well with the ideas that we'd had. Let me name some of the things that came into the 5000 out of that UCLA conference: the paging of the Atlas; the single number form came out of the G-20; one thing we got from Stretch was the name of the master control program --that's where it came from. That was the first instance of MCP; we stole the name. Then one of the things about the paging of the Atlas is that it was implemented using associative memory, which at that time was very expensive. I forget how much a bit in the associative memory cost, but it was beyond what we could afford. So there was always a thing of how in the heck could we fit that concept into what we were planning. At that time we had the things that became descriptors, we called them array words.

BARTON: Where did that come from, Paul? Do you remember where?

KING: The array words?

BARTON: You used the term descriptor.

KING: I don't remember it being called descriptor at that time. It was an array word and there were various control words. There were control words that came over from the 2111, the I/O control words, and then there was this array control word, and I've got some of your early papers that refer to it.

BARTON: You're way ahead of me. I don't have them.

KING: Well, I do. The thing that happened, somewhere in there, and I'm not exactly sure where and who actually came up with the idea, was, "Hey, let's take that array control word, and let's put a bit in it to indicate presence." That was a real breakthrough. There was another thing. In order to implement multiprogramming, we had to have storage protection. The original way it was done, is we had everything laid out in areas. Here was a program area, here was a data area, here was the control area. These were separate areas of memory bounded by registers. That was a very clumsy way. Then we came up with this idea of using a control in the descriptor in this array control word that allowed us to do presence checking and gave us storage protection, except for character mode. [Laughter.]

One other thing that came over from that conference, the flag bit. That came from the Stretch and that was one of the mistakes. But you can't get everything good. So what happened is then we went back and there were a number of very interactive sessions that we used to have in the conference room in product planning. I'm not exactly sure when Clark got involved.

OLIPHINT: Somewhere. I don't remember exactly either.

KING: And Merner was involved and there's Paul Colen, Dick Smith, myself, and Bob used to sit in from time to time. He'd appear once or twice a week in these sessions. Gradually this thing evolved. One document I have is the functional specification of the 4000/4400 as of August 1960.

MACKENZIE: Have you got that still?

KING: It's one of the things I saved. It's interesting because it shows a transition, it still contains the two stacks at that point in time -- in August. Now they disappeared by September or October. It was Merner that was involved, so I'm sure that he was in there at that time. We already had all the string operators -- not all of them laid out, but a number of them laid out.

BARTON: Who are you going to credit with the string operator? I'm curious about that. [Laughter.] You don't have to name names but at least eliminate present people if possible.

KING: My view of where it came from, and I know Paul Colen disagrees with me, but I believe it came from a string language that was implemented in the 220. There was a string language that was used I thought with an assembler or something for the 220.

OLIPHINT: Yes, that's right. It ran Cardatron. It produced format strings for the Cardatron.

KING: This is what programmers would like to have. They designed a language for handling strings. So let's try to build that into hardware. And that's my view of where the origin of that set of operations came from.

MACKENZIE: Out of the format bands on the Cardatron?

KING: No, not the format bands itself. It's the language you used in creating the format bands. The format bands were...

OLIPHINT: They were a bunch of bits.

KING: Yes. The format bits. But there was a language that you used to create the format bands. It was that language that was used as the source of what we called the character mode. Then, during that time, there was then...

TAPE 2/SIDE 2

...process between what was coming out in specifications and people starting to think how we're going to implement that software. And changes kept happening. I remember we started out with only three syllable types in the machine. We had an operator, we had a literal, and we had an address. And there was another where all we did was use it as a base plus an index to get at memory. That was not figured to be very useful, but we kept it because we didn't have a good alternative. I believe it was Merner that came up finally with the solution. People kept suggesting things: "Let's use it for this, let's use it for that." We had to be very careful because we only had four syllable types at that time. So, he finally came up with the operand call descriptor call --the call by value if you will, and the call by name. So that's what ended up going into the specs. I think that the specifications were pretty much frozen by October or November, somewhere around there of 1960.

DAHM: Who invented the idea of the accidental entry? Where did that come from?

KING: Accidental entry?

TURNER: Merner.

ROSIN: What is the idea of the accidental entry?

DENT: Call by value, call by name.

KING: Accidental entry was by Merner, I'm pretty sure.

DAHM: Part of the idea was that when you fetch something, the processor wouldn't know until it hit the word it was fetching whether there was going to be data there or whether it was going to be an indirect reference to code which would be executed to furnish the data. It made it a very clean way of implementing ALGOL name call expressions because the basic code inside the procedure wouldn't have to do anything special to access a call-by-name parameter. With access just like a call-by-value parameter.

MACKENZIE: Dave, there was a very crude precursor of that even in the 205. When you were reading in paper tape and the tape controlled the machine when it would come across a control word.

KING: Yes, but that was undoubtedly Merner.

GALLER: All right. But now I remember people like Ingerman talking about thunks and so forth. Were those ideas in the air at that time or was this something that...

DENT: Oh, it was required to do ALGOL.

OLIPHINT: I remember going to a meeting where Ingerman talked about that, and I remember thinking that it's too bad that people have to do that clumsy kind of a thing when if they just had what we've got, they wouldn't have to worry about all that.

GALLER: So you're saying, in fact, this was an innovation in a sense of Burroughs; it wasn't influenced by other people, it came out of a language assignment.

OLIPHINT: That's my recollection.

WAYCHOFF: Ingerman gave a talk in October of 1960 at the National Compiler Symposium in Washington, D.C., where he described thunks and he coined the word thunks, but as Ben said, it's simply required do ALGOL 60.

KING: One other thing. We talked about the disk storage. I can remember during that time -- and before that -- participating with engineering and going around looking at mass storage devices to try and find something that we could use as mass storage. We went down to Aeronutronics; they had a disk, and Bryant. The best we could find at that time was some Bryant drums. They were 1 megabit drums and that was the largest we could get that ran at a reasonable speed. There were some drums around that were very, very slow, but that wouldn't be very good for backup storage. So we opted for those specs that matched to the Bryant drum, and they ended up getting built internally at the Pasadena plant.

DAHM: One of the things I remember with respect to the disk file MCP. We didn't really have any idea when we started how we ought to manage that disk. Now today, one wouldn't consider managing a disk in any other way than having a disk directory on it and a variety of different files stored on it, but that wasn't very obvious to us in 1964 when we were designing the disk file MCP. I remember protracted arguments in which we were trying to decide whether to take that approach or whether to essentially let the user deal with it however he best could. In the end we decided to have the operating system do the disk allocation and to store files on the disk. It was obviously the right decision in hindsight, but it certainly wasn't very clear at the time we were arguing about it.

KING: One of the things, too, is that we did not specify software at that time, like hardware was specified. There were product specs on hardware, but you never did a product spec on a compiler or on a control program, things like that. That came later.

MACKENZIE: I would like to state for the record that one of the principle frustrations of our existence was the lack of any good product specifications in a hardware sense -- just so that it's not misunderstood. [Laughter] I want to put things in context properly. We were begging for those things. My recollection is it was the fall of 1962 before there was really anything that was really intelligible in a product spec. This doesn't mean, mind you, that the engineering people didn't know what they were doing. That isn't the point at all. We're talking about the documentation of the thing and a specification form that programming people could work with to understand the machine. I used to get that shoved at me so much that I had to make that speech.

?? Feel better, Brad?

MACKENZIE: Yes, sir. [Laughter.]

BARTON: May I make some comments and this probably will exhaust me. I listened very carefully to what Paul said, and I think it's an accurate perception. Paul saw what was happening at the time, and I suspect that there's probably pretty good documentation that that's exactly what happened. But is it what happened? Now, let me give you another version that's rather curious. I've always told the story, and I wrote it down in a couple of little papers which essentially might have been titled "How Bob Barton Thinks in Case Anybody's Interested" and how I saw the (descriptors?) and segmentation. Now, in fact, I did see it this way. I don't think if you're polite that you won't deny it, just as I'm not denying how Paul saw it. I said earlier today I think you can run inventions back as far as you want to. I did not know about PERM; I did not go to that UCLA course, but I want to tell you some funny things about it. I'm sure that that had considerable influence on Paul, and I'm sure that the work on, I guess we'd call it the 3000, the paper planning machine, had a good deal of influence on how he heard what he heard at UCLA. I have no idea whether he ever heard what I said. He may not have. I'm not sure when he went to UCLA, either. The UCLA course may have prepared him to hear what I said; I may have prepared him to hear what was said at the UCLA course; I don't know. I just want to tell you what I thought happened and my motivation. It goes back to Shell again and a lot of thinking we did about whether we wanted to put tapes on our machine, and we concluded we didn't. We wanted to have a disk; we couldn't afford to in that situation, but we thought some about what we would do if we did have a disk. About that time we were beginning to be influenced by the ALGOL movement. The 1958 report was out, and we were fooling around with experimental small compiler languages like Perlis's, another influence. One of the things that struck me at the time in a fuzzy, sort of ill-formed way, was that part of what the IAL was suggesting was dealing with procedures as entities and arrays as entities. It seemed kind of natural to imagine a scheme of tying together two levels of memory (and in those days that would be core and disk) on the basis of a natural division of the program in terms of procedures and arrays. You could use some other terminology; if you were commercially oriented, you could say tables of data, perhaps file buffer stores, that kind of thing. But anyway, I was motivated not at all by the idea of paging. I did know about paging, not from the UCLA course, but by a short note that appeared in what I think was the Journal of the IEEE that told about Atlas. I remember reading about paging and having had this idea about using natural units. I objected to the notion of just chopping up memory in blocks and having things flop over boundaries, and I also objected to something else: the theme that's run through all my contacts in this field. They talked about an operating system that would learn. They had the artificial intelligence plan. Years later I had a guy working for me in Australia that was supposed to write that system, by the way, and another guy that was an associate of Berker and had learned Berker's programming techniques. So I know how they thought at the time. All I'm saying is, when I read the short description, I immediately disagreed with it. Now, it seemed to me -- in the way I think about things, usually pictorially and usually in Gestalt, not in systematic analysis -- that I had to picture something like this: ...word that would sit in the main store of the machine and contain a base address, bounds, and a bit to make a distinction whether the addresses referred to the disk or the main store or secondary store. I also distinctly remember having pulled out the term descriptor. The term came from the guy who was a theorist in information retrieval, a library science type named Mortimer Taub who had written some little books which I found in the Shell Library. I grabbed on that term descriptor and hated it ever afterward because I had a gift for picking bad terms. I don't like to pin things down with words any more than I like to pin them down with fixed pictures before I'm satisfied with them. And here this term descriptor got latched onto and used, and it was borrowed by the GE people, and it became almost legitimate computer science jargon later on. Mind you, I'm saying that this is just part of the description of what I remember happened. Now, Paul is describing what he saw; at some time I'm sure he heard what I said. He may have ignored it, and he may have thought about it from a totally different point of view. It quite is quite possible that you could prove that everything that was done there had already been represented in Atlas, PERM, or whatever. I'll tell you another thing, and that's the Rice computer. The Rice computer was located very close to where we were at Shell, and I heard a presentation on the Rice computer. Now, who's the guy?

?? John Iliffe.

BARTON: Iliffe wrote a book after it. And that book was essentially reconstruction after the fact. Gee, we had all these ideas on the Rice computer. And while he was a gentleman -- I don't think he ever claimed we ever stole them from him -- I suspect he may have wondered. Now, when I heard the description of the Rice computer one day, I thought they had a very elaborate indexing and indirect addressing scheme. As far as I'm concerned, it did not influence the view of segmentation that I just described. It was completely independent of that. Now, so what? Am I interested in credit for that collection of ideas? No. Beside the point. All I'm trying to illustrate is that two stories and two people that I like to assume were honest and objective are both correct. But, I was a consultant. Paul wrote the specs for the machine. The machine defines what we're talking about, and all I can say about the idea of descriptors and segmentation is I consistently made that part of my equipment later on for working on other machines. I'm still trying to understand really what that kind of thing meant.

There may be still another version of the story of the descriptor and segmentation. There's a sort of after-the-fact view that it was an invention of virtual memory. Well, who defined virtual memory? Did IBM define it effectively? We certainly didn't call it virtual memory at the time, but a lot of hay was made in the trade press about IBM 10 years later reinventing virtual memory. I don't think that's what happened. I mean, in ma and pa language, I'd say that I was interested in the problem of how to tie two levels of store together in such a way that it fit the program structure. I tried to convey that idea in the form it was in my mind at the time, and I don't know whether it influenced what was done. Because there are other ways, for instance, the Iliffe way of saying well, we had all that stuff in the Rice machine -- all those concepts. I believe later on ICL did pick up from Iliffe a set of ideas having to do with descriptors and indexing and all that. There are lots of funny might have beens in connection with this. You didn't ask the question, why didn't the machine have index registers? Well, if you asked me that question, you'd have to put it, of course, in this form: as a consultant to that project, why did you recommend against index registers or didn't you care, or who was it that left them out in your opinion? Well, I think that what happened really was something like this. We had Iverson as a consultant and I think Duncan MacDonald must have had a lot to do with arranging that. He was a consultant in engineering at the time, but that time when I was a Burroughs employee, I was asked to contribute some money from my budget. I agreed to doing that, and I remember going to a series of talks by Iverson, and my reaction was, "Wow!" He was still at Harvard at the time; he had an unpolished system (it was many years before the book was published), but I sat in that lecture thinking you can do the kind of things he's talking about in a machine with very little more hardware than it takes for three index regis ters. I used to argue against index registers on the grounds that ALGOL indexing was more general, but in the back of my mind I was hoping that the machine could have vector operations. I wasn't very good at convincing anybody of that; in fact, I failed again with the 6700 program. I failed. One of the customers convinced Burroughs to do a retrofit on vector operations at that time -- I don't think a very good job, either. But that's a perception I have that has nothing to do with the record. I bet Paul cannot produce a single item that will show that there was ever a suggestion that vector operations be built into the machine. Am I right?

KING: Sure.

BARTON: That's true. Does anybody else here remember being harangued at any time about vector operation?

ROSIN: There are several hands up.

BARTON: And so on. Many different perceptions. I was well paid for my small contributions.

GALLER: I'd like to point out, Bob, that I personally think the word descriptor was a very fine word because to me it suggested -- maybe beyond what you had in mind for it -- that a great deal of information of various kinds would be associated with that.

BARTON: That's true. I have to admit that that is a good point. A very, very, good point in favor of the word.

When you consider, you consider typing as full elaborations and we were just touching on typing at that time, typing program or data. That was a neat little thing.

GALLER: OK. Now, Paul and Bill, you decide between you who should talk next.

LONERGAN: Well, I haven't talked for a while, I'd like to talk. I'd like to reiterate a little bit of what I said in the letter I wrote to the IEEE Computer. I agree with Bob's earlier comments about there being very little that's new. But I don't think that's unusual. The idea of evolution was all around when Darwin did what he did; most ideas are that way. They're not a wild leap out of the blue, so I don't think we should be startled by this . The question is: which ones to use and which ones not to use, and how to try to put a good machine together that's got some internal consistency, and balance, and all those nice things. So if you ask me how did it happen, I said in that article (and I really believe it from practice) is that the machine evolved from, let's say, April or May until October or November. And if you ask me where did it evolve, it all evolved on Paul's blackboard, unquestionably. Various things would be a big problem, we can't do so and so whatever, and it would be thrashed at for several days. A number of people would make suggestions. It's very hard in retrospect to say who actually was the person that came up with a solution to this. But, very typically, after several days of thrashing and people going in and out of Paul's office and thrashing all day with the blackboard, he then would have a session with me and he would tell me what it is that had been changed and why or what it is that he had rejected that was proposed to be changed, or why he rejected it. So the machine kind of evolved over time. Now, it's very difficult to say X did so and so, because sometimes there's three people in the room and somebody says, "what about so and so?" That influences another party to say, "Well, that would be nice, but why don't you do this way instead?" So the thing is a multiple thing. There are lots of previous machines. There's another machine, a German machine called the ER56, which I said in the article I was also influenced by, which happened to be a modular machine with separate I/O, and multiple processors, and a cross-bar switch in it. I really liked that architecture, and that's a long time ago. That's like 1957 or something or other. In fact, the Germans had some pretty damn good machines in those years. So, the fact that the subroutine thing came out of PERM is fine, as far as I'm concerned. So there's lots of plagiarism. In fact, designing a really good machine amounts to plagiarizing the best things that various people have done. And it evolves over some time. I'd say over that 6-month period the machine evolved. A lot of good things were put in, some things were thrown out. It's impossible, now, to go and say that the following six guys influenced this thing, or whatever. That's really where the reality is there. But I think that's fine; that's not a thing to be upset about or something. That's the best way to do it; small group of highly competent people. I would like to reiterate my earlier comment. I had Don Knuth in that group, and this is not an average group of people at all in the computer field.

BARTON: Don Knuth is a very good example of somebody who has a fine, established reputation. He certainly made some important contributions. One that I was sort of aware of at that time was his serving as a human logic checker for engineering. He wasn't a logic designer, but he learned the notation quickly and was able to follow it quickly and, in his usual thorough manner, made a big contribution. I don't think, however...

KING: It was a lot faster than building the machine to check out the -- that had been traditional. [Laughter.]

BARTON: I think, though, that Don himself, who apparently decided he shouldn't come to this meeting, would probably say that the kind of creative effort that went into the thing --not the sort of creative effort that he's known for (I think he'd say that; I've heard secondhand quotes to that effect, and I don't know whether I ever heard anything first hand from him) -- one thing that bothers me a great deal...

GALLER: Paul was waiting to say something.

KING: Yes. I wanted to go back to that UCLA conference and the Atlas and such. I can recall coming back from that conference; at that time, Bob had just gotten involved with the group.

BARTON: Do you remember the month that was, Paul?

LONERGAN: It was May of 1960.

KING: Late May, early June. At that time I remember spending several hours discussing the Atlas paging system with Bob and explaining it to him, and at the time, he didn't show any awareness of it. Then I loaned him my Atlas programming manual, that he took and went off and studied for a while. So, that's just a small point I wanted to clarify. But, mentioning Knuth, I used to use Don as the auditor of all changes to the processor. We had this processor spec, and there would be controversy. Clark and Lloyd would be arguing about where was something was going to get done, how it should get done, the influence on the instruction set, and there would be two differences of opinion. I'd often turn to Don and say, "Hey, what's your view on this?" I also used him as the auditor of the specs as they got done.

DAHM: I'm not clear exactly who's responsible for what details in the instruction set. I wasn't there when the most important decisions were made. But my view of what we thought we were doing during the development of the programming system: I personally felt that what we were trying to make real a vision that Bob Barton had been talking about for years, going back to the days at Shell, of what a computer system could be. It was a lot more than was current in the industry in those days. It included things like multiprogramming, multiprocessing, virtual memory. These terms didn't even exist in those days, but he could describe the ideas in very graphic terms. To me, what we were to do in the development of the programming system, was make real in a product those ideas, following that vision. Ultimately we were successful at doing that. It was not a simple task, and I think most major corporations today probably would not undertake such a thing with the vague, foggy ideas we had of exactly how we were going to make those things work. We didn't know when we embarked on this adventure, how we would in fact make them work. There were a lot of inventions that had to be made along the way to make them work. But from my point of view, what we were trying to do was instantiate (sic) the vision that Bob Barton had of what a computer system could be.

BARTON: I think it's still going on. I wish the devil I could stop participating in such things, but I think I'm going to be doing it for several more years. It's still going on, and all ideas, whether they come out of some abstraction and you're just trying to apply them (we're dealing with abstractions), have an enormously complex history, and with struggle we're trying to implement them. There was a period in Burroughs years later when I had a lab (I don't know how long, six years or so), and we built various things. It was a show-and-tell lab. What we were doing was taking ideas that were kind of mundane, they were all within reach of available hardware, but they were the kinds of things that when you talked to people about them their eyebrows raised. So you had to build a demonstrator of some sort. There was a period of time during which fuzzy ideas like that were being translated directly into hardware by some smart engineers who had been selected because they hadn't worked on computers before (they worked on graphic systems or something else), so you didn't have to argue quite as much with them about not doing it the conventional way. I don't think that was worth an awful lot to Burroughs because, as MacDonald said (I hated the man after he said it; it took me some time to get over it), "Some of these things are 10 years ahead of where we are." I thought, if that's the case, there's no point in ever working on them. And yet, those were half-baked ideas that we were trying to pull closer to reality. The systematic view of what happened in the B 5000 is well worth recording. The comments I make are not the systematic view; they're quite a different thing. Quite a different kind of person. Paul and I are absolutely different. Let me give you another example from that conference. I didn't know that a conference preceded this. I think about two years later, after it was all over with, maybe even longer than that, I learned that the G-20 had a peculiar kind of arithmetic built in. I did not know at the time. But I remember very clearly inventing that out of whole cloth one day and having it accepted readily.

KING: That's not true!

BARTON: Oh, it's not?

KING: It's absolutely not true! I came up with that single number form. I did it on my blackboard with nobody else in the office. You ignored it. It went over your head; you didn't even listen.

BARTON: I rediscovered it. I rediscovered it.

KING: That's absolutely not true.

BARTON: Oh, that's fantastic, Paul. You're denying my memories. I'm not denying yours. You can say you did that. You may very well have done it. I don't want any credit for it; it's not really worth very much.

KING: I don't care. But you're saying that you did it. You're saying now that you put that in, and it's not true.

BARTON: I didn't put it in.

KING: Absolutely not true.

BARTON: I didn't put anything in!

GALLER: I was invited once to talk to a group of historians on the problems of doing contemporary history. One of the problems of contemporary history is, in fact, that different people do have different recollections of what happened. I think we have now on the record a little bit this disagreement. I'm not sure it would be very productive to explore it much further.

LONERGAN: I would like to sustain the fact that it came out of the UCLA conference because when Paul came back is when he introduced the idea. It was stolen from the G-20.

GALLER: As I say, I think we have the elements of the disagreement now on the tape and maybe that's enough.

BARTON: I think I need one rebuttal on that. May I have one rebuttal?

GALLER: No. I should have mentioned, by the way, that at least those parts that we will put in the *Annals*, we will invite anyone to rebut.

BARTON: Well, I'm not going to do it after today. This is my last contact with this whole thing. May I have one more thing to say since I brought that up? I'm glad I brought it up. I didn't know it would push Paul's trigger or pull his chain. I'm telling you what I remember. I am not denying it was in the G-20; I am not denying that it was in your head at the time of the UCLA conference or after you heard it. What I distinctly remember, though, is inventing it out of whole cloth, and your sort of agreeing with it. Now, you had a style as a professional devil's advocate of perhaps wanting to see how somebody else felt about something that you had already made up your mind about one way or another.

KING: Do I get a rebuttal? [Laughter.]

GALLER: No. No.

BARTON: We don't really have any disagreement, do we?

GALLER: Bob, I'm sorry.

ROSIN: There's a problem here.

BARTON: What is the disagreement?

GALLER: I think we've explored that topic enough here. I'd like to ask if anybody would have any final comments on how you feel about the B 5000 project.

BARTON: May I ask one question, which is for the record? These guys have a lot of documentation. I want to bring up the issue of how patent attributions were made?

GALLER: I think that's a separate topic.

BARTON: It is separate? You mean it's not going to be part of this meeting?

GALLER: Well, maybe we can bring it up this afternoon in terms of the management functions, but I'm not sure that that's something we will discuss here.

KREUDER: Let me point out that when it came time to make it work, King was gone, Lonergan was gone, and Barton was gone, and so it was up to my guys and Brad's guys to make the thing go. [Laughter.] You guys were all gone by that time!

GALLER: I think that's very important.

HALE: I want to make a point about that, too, thank you Norm. You know, it seems like we've spent most of the day talking about this machine, and how it's ideas came about, and how it came to be specified, which are certainly important. But I think equally important is us poor bastards who had to make it run.

-- That's right.

DAHM: It's too bad we didn't spend a little more time talking about the implementation aspects.

KING: The 5500 was the success, but I don't think the 5500 would have happened if the 5000 hadn't happened.

KREUDER: That's for sure. Of course.

ROSIN: Why is that? We heard earlier that the difference between them was just the disk file.

KING: Oh, no. There were some other... It was learning how to do an MCP.

DAHM: There were some minor differences other than the disk file, but the real major difference between the 5500 and the 5000 was the disk file. That was the biggie. There were a lot of other smaller things, but they were less important.

KING: The 5000 never could have succeeded if it hadn't been changed.

TURNER: It's interesting to note that the disk file operating system ran fully on the 5000 processor without the 5500 changes.

MACKENZIE: Yes.

KREUDER: Sure.

GALLER: I think it's clear that the B 5000 was a very important machine. There were some things about it that made it not a success as it was implemented. The B 5500 certainly was an evolution from it which changed some of those shortcomings and, in fact, led to a very successful family which followed. There's no doubt about that. Brad, you look like you're bursting to say something.

MACKENZIE: You ought to view the 5500 as merely a worthwhile and planned recycle of an engineering design at a propitious point in time. It was seized upon from a marketing point of view to be a rejuvenated machine, and that's basically what it was. It was an important thing to have done, but it wasn't, in any important sense, as major as the differences from the 5000.

ROSIN: Was the 5500 program a success in the judgment of the people here?

MACKENZIE: Yes, there were 220 of those machines produced ultimately, which was, for us, a good size run at the time.

KING: And recognize that that thing was on the production line longer than any computer in history. You take a 360, for example; it had a three-year manufacturing cycle. I think the 5500 had a ten-year cycle.

MACKENZIE: The last 5500s were manufactured in the spring and summer of 1970. There were 220 or 221.

DENT: There was one hiccup -- a kind of major disruption in program compatibility and all the rest -- but except for that one hiccup, it's been going for however many years it is now: close to 25 with no foreseeable end. The architecture has lived longer than any other architecture I know.

ROSIN: You're referring now to the 67...

DENT: The A-Series, and all the rest of that. They're really very similar.

KING: It kept Burroughs in the computer business. There was a teetering point there of whether they should stay in the computer business or not. And when they realized the money they were making out of the 5500 program, they stayed in.

GALLER: All right. Bill?

LONERGAN: I wanted to make a comment about that. I went back to work for Burroughs for, let's say two years or so. Subsequently when I was having an interview session with Ray MacDonald in the Detroit Athletic Club, he said to me, "Two ideas really made Burroughs successful over my span of time." (I left when Ray Eppert was the president, I might not have left if MacDonald had been by then). But, in any event, he said two ideas made it for them. One was going to an electronic bookkeeping machine, namely replacing the mechanical Sensomatic-class of machines with electronic things probably three or four years before NCR did it, which really gave them a tremendous thing. I'd say my outside view is that they failed to follow up that nice advantage that they had. The second one, he said, was the systems that came out of the group in Pasadena. Then, of course, he said to me, "Now my question is, what's going to do it for us next? And that's why I'd like to have you come back here." Even so, my point is, there's a good recognition that those two fundamental things really put Burroughs onto the map. Another final comment I might make about the machine structure, as I said earlier, is the notion of having multiple processors and all of that in a grow-able kind of system and wanting it to be fail-safe, or graceful degradation, or whatever you want to call it. Years later, when I was back at Burroughs, there was discussion in product management about saying, "Let's give up this multiple processor thing because customers don't want it, or whatever -- having the ability to do that." I argued, unsuccessfully, "Rather than give it up, why don't you carry it further and really make it such that you can pull out cards and it doesn't fail and everything else?" Of course if they had, they would have done what Tandem and a lot of other people are now trying to do. So rather than retreat, I still felt I convinced even 15 years later that they should pursue it rather than retreat it.

GALLER: I'd like to make just a couple of closing remarks here. What I see is at a very interesting point of time, there were several different kinds of things that sparked in Burroughs. There were people with ideas -- lots of people with ideas, not just Barton. There were people with ideas. Bob happens to be sort of specializing in that. There were people with all kinds of ideas, and there were people who were pragmatic and had the responsibility and knew how to finally make decisions on which ideas to use and how to do them and to put together iron and code and produce something. We've talked about evolution, but there really were some major steps here in terms of the state of the art at that time. It certainly is not to anybody's discredit that the first major version of this thing had some things that needed to be changed. I think IBM, for example, is still talking about how their 360 architecture has lasted 20 years. We've heard that this basic architecture and strategy has lasted already 25 years. That's very interesting. One of the problems with all of this is that it's a very well kept secret. Now, I know that Bob Rosin and I, as academics, did appreciate what you guys were doing back then. We watched, we learned, and you can believe that we used this material in our courses and tried to educate people because there were some good things happening. We didn't buy your machine. That's a separate story. [Laughter.] We didn't buy your machine at Michigan. We were ready to pay for a machine at that time. We didn't buy it, in our case, because they said, "There's a MCP." When we said, "Can we get at the source code because we might want to change something?" (that was our business; operating systems was our business at the time), they said, "Nobody's going to get at that MCP." We said, "Fine. Goodbye." That was a different relationship then, but we appreciate it more...

DAHM: You know, that's very interesting. Because, in fact, for every customer to whom we delivered the machine, we also delivered the symbolic, and an awful lot of customers modified their operating systems.

GALLER: Well, there were some miscommunications. In any case, you were appreciated by some people and I think you can be very, very proud of what you did. Thank you.

SESSION 2

TAPE 3/SIDE 1

GALLER: In the second session of our discussion with the Burroughs people, we are going to take some time to talk about the realization of the system into a real product, some of the effects on the organization in doing that, and the opportunities and problems that some people ran into. Then we'll have another session where we'll talk about the marketing aspects as we had originally planned. I guess I'd like to start off with the question of what happened in the process of bringing it out. In particular, we know that there were some organizational changes, and it appears that in some of the structural questions like the role of Product Planning, Automatic Programming, and so forth, there was a shift at a later point in engineering, etc. I'd like to know what this did to the whole project and, again, what problems were encountered along the way. Brad, maybe you're the one to start with some of the organizational questions.

MACKENZIE: You're talking about the administrative organization and the way the company was organized to do it?

GALLER: Yes. I think that is one aspect we'd like to bring up.

MACKENZIE: Originally the programming systems (for the people who were covering this morning) were the two compilers, ALGOL, COBOL, and the MCP. And the original proposal, or plan if you will, was that the MCP would be specified, designed, and implemented by Product Planning. The COBOL compiler would be specified and designed by Product Planning and implemented by Automatic Programming, and the ALGOL compiler would be designed by Product Planning and implemented by Automatic Programming. That was the plan, and at the time there weren't very many people, but they were in Product Planning. There had originally been, I think, ten or twelve people in Bob Barton's Applied Programming, or Automatic Programming Organization, as it was called at that time, and they were split roughly 50-50 with people going with Bob to Product Planning in the spring of 1960. The other people were left in Automatic Programming, and I became manager of Automatic Programming at the time. Subsequently, we started to try to work on this plan when Richard and Lloyd came in from Dallas and actually did the work on the ALGOL compiler. Richard came in, if I remember, in September or October of that year and, Lloyd, you were there right at the end of the year. We announced the machine in February of 1961. At about the same time it became very clear to Bill and me both that the split, if you will, organizationally, wasn't really going to work and that we had to do something in terms of having one jurisdiction and responsibility for all the programming systems. We subsequently decided that Automatic Programming should take the people and do all of the work on the entire programming system. That decision was made in early 1961, and during the rest of the summer and so on the people were transferred to Automatic Programming. Clark came over, Al Litton I remember came over, Frank Crowder, there were six or seven people all together that were transferred into what was then Automatic Programming. Over the next couple of years, we built up the Automatic Programming organization to perhaps 25 people. At the time we delivered the system, we had 35 people in Automatic Programming, and the group was never any larger than that. There were probably a total of 75 man-years or something in the development of the programming system at the time it was completed. So that was the general framework, organizationally, that the programming system worked and was done in that time. I think Engineering was basically organized for the project, and the engineering organization stayed pretty much the same through the entire implementation of the 5000 until later on when Automatic Programming became part of Engineering and then Norm was named program manager for both the hardware and the software parts of the programming system at that time. I went to work for Norm then, with all of these fellows. During this period we had four sections in Automatic Programming. Lloyd Turner was running the Scientific Systems Development Group; John ran the Commercial Systems Development Group; Clark became the manager of the Control Systems Development Group; a fellow we haven't mentioned today so far, Warren Taylor, was the manager of the Systems Improvement Group, which you can think of, if you use today's terms, as a continuation engineering and a support group for the organization. That was the basic organization at that time.

GALLER: Did it work?

MACKENZIE: Yes. It worked fine. It took a while, extremely dedicated work from the people involved. It really was quite an experience. It was the thrill of a lifetime in that sense for almost everybody involved, although they were all pretty tired when it was finished, I think.

GALLER: At lunch today, we were hearing various stories about Lloyd Turner's management style. [Laughter.]

TURNER: Or the lack thereof.

GALLER: Well, what was it like? Either from Lloyd or someone who worked for him?

CREECH: Well, it was hard work. He demanded total dedication, total performance. We worked lots of long hours -- nights and weekends and so forth. At times during the project, we hated him. I had a problem: he wouldn't let us go eat; I was always thinking of a way to go eat. But at the end, when we got done with this thing, I think all of us realized that he had drawn the best we could do out of us, and we were proud of what we had done -- what we had accomplished. Left to our own devices, we probably would not have accomplished what we did.

GALLER: Certain parts of this discussion sound an awful lot like the book, *The Soul of a New Machine*. Would anyone care to comment on how similar your experiences were to that description or how different they were?

OLIPHINT: Yes, I read that. And I was struck at the time with how well I thought the author had captured the spirit -- the kinds of things that go on, except for the fact that it was hardware oriented and my experience was, of course, software oriented. I thought a lot of the spirit and the feeling for what was going on was fairly close to my experience.

GALLER: In that book (I'll just pursue it for a moment), the ending was, to me, a little sad in the way the group just sort of dissipated. How did this group end?

ROSIN: How did you know when you were finished?

OLIPHINT: I don't know. I wasn't there.

TURNER: For my project, we did the compiler, and when we finished with the compiler, we did it over again. In fact, we actually did it over twice. We did it over, and then we did an improvement. After it was the way we wanted it, then we fooled around with it, just a little here and a little there for a short time before we started working on the operating system. The disk was coming along at about that time, and I couldn't wait to get my hands on that part of it. I suspect Clark was kind of tired by that time, and I was all fired up and ready to get on with the operating system. But I think that from my standpoint, the thing was so important to us, and there was so much riding on our getting our job done on time, that we had to do what we did or else everything would have slipped. I was not about to let us get on a critical path past where I had promised. That's one of the things in the early days that I absolutely would not tolerate; missing our targets.

GALLER: You mentioned four groups. I guess the fourth one didn't actually produce a product, but did they come in on time with their schedules?

HALE: No. Mine was horribly late.

OLIPHINT: You heard already from me this morning that mine was late.

MACKENZIE: Well, John, it's interesting in this sense. Late, yes, in terms of the requirement. Late, particularly in terms of our expectations when the pro-duct was formed, probably not. I've got quite a bit of schedule stuff where the estimate was that the COBOL compiler would take 10 months from the time the 5000 was available...

HALE: I didn't make that estimate. [Laughter.]

MACKENZIE: No, this was before John's time. The problem was, as time went on and things slipped out in time, that the first customer shipment dates didn't slip out, correspondingly, with changes in the program because of revenue importance and things like that. We had originally anticipated, I think, that we would have 6 to 10 months of time on the machine to develop that. We planned to have two 5000s so we'd have time on the engineering system, (which they gave us time on; in addition to having a system of our own). When it was all over and said and done, we got our own system two weeks before the first customer shipment, and we had about four months of time on the engineering system.

KREUDER: But the first few months on the engineering system were pretty flaky because we were using the software people's use of the machine as a way of getting the last of the bugs out. In fact, it's a very effective tool for getting the bugs out, but it makes it relatively useless as far as checking out programs is concerned.

MACKENZIE: At the time of the first customer shipment, we had something on the order of 1200 - 1400 hours of good, productive time on the system over the 5 -month period. The first couple of months or so, we averaged maybe 6 or 7 hours during a day of good time, and over maybe a 4 - month period, we started building up to 10 to 12. We were definitely short of resources; that's why the people were working seven days a week around the clock.

TURNER: Except in our area. I didn't feel that I was short of resources.

CREECH: I did! [Laughter.]

OLIPHINT: We could have used another machine, though.

MACKENZIE: That's how you guys finished your compiler.

HALE: You must have because I remember arguing with you and David about who got what machine done.

TURNER: No, I meant people. I had plenty of people.

GALLER: Lloyd's resources are people. He had resources. The people's resources was machine time, and that's where there was a bind, apparently.

MACKENZIE: There were definitely problems with machine time. It was a struggle in that sense. I think finally, probably the late summer or early fall of 1963, we did have two machines for our own use at that point. From that point on, we had plenty of machine resources, as you remember. There were start-up problems with the program -- getting it going. The biggest hurdle was getting our own machine to work on because we were interlopers in a sense in the engineering lab, and the system 101 was, by definition, the earliest model of the system available. There were additional subsystems being put on it. It was very commonly, as Norm was saying, not available for the work. The fellows always had rather interesting ways of standing by, but there was somebody usually standing by seven days a week, 24 hours a day, to get on the machine when they could. Now, standing by might include being down at Norm's having a few beers while you were waiting for the machine to come up.

TURNER: One of the things that we did is that I insisted that all four of my guys be there always. That is, we wouldn't just have one guy and they'd have to call him. If they weren't all there, there was hell to pay.

ROSIN: Twenty-four hours a day, 7 days a week?

TURNER: Just about: 15 hours a day, 7 days a week, for 5 months without a day off. Not even one.

CREECH: It's interesting to note, though, that the actual implementation phase of the compiler and later the ? was a very small proportion of the time of the total project, which sometimes gets lost in the views of what it takes to do a development project. The design was done at design time, and not at implementation time. The discipline to not start writing code until it was time to start writing code was extremely important. There was a very small amount of implementation time on that.

ROSIN: What would you imagine the ratio was, design to implementation time?

TURNER: The design starting in early 1961 until June of 1962, and we got in a simulator. So it was about a year and a half of design. Then it only took us from June until November to get the thing implemented, but we spent some 6 weeks to 2 months on the simulator. Is that right?

DAHM: About that.

TURNER: And then maybe three months on the machine.

MACKENZIE: They had a machine simulator on a Philco S2000 in San Jose --

GALLER: To clarify: when you say June to November to implement, does that mean debug or code?

TURNER: Code and test --to finish the coding. It was working enough for the COBOL people to use it in November.

WAYCHOFF: I wanted to mention the simulator phase. We had a simulator running on the Philco S2000 at General Electric in San Jose. We ran on that for several months, and during that time the operators were supposed to take off at midnight, but Lloyd could always talk them into staying for one more run after midnight. Then we'd leave there and drive over to Stanford to reassemble the compiler because at that stage, we were just riding the bootstrap thing in OSIL -- the assembler -- and I think we were averaging about 3 hours of sleep a night during that time. I remember that we were told that the B 5000 prototype was working, and we could forget the simulator and go back to Pasadena; I was so happy. But we got back to Pasadena and the machine wasn't working well enough for us to use, so I had to turn around and go back to San Jose again. That was an agonizing trek back.

GALLER: John Hale, we haven't heard mu ch about the COBOL effort. What language was the COBOL processor written in?

HALE: ALGOL. It turned out to be our savior. We had so many problems. When I came to Pasadena in January of 1962, and I didn't even know what COBOL meant. There was a compiler designed -- in existence -- that, I'd have to say in retrospect, apparently was pretty bad. Unfortunately, we didn't know how bad it was. I didn't know how bad it was; I don't think anybody else did either, until we got on a B 5000. We did practically no simulation.

GALLER: What about design?

HALE: I don't know. The design was done before I got there.

GALLER: Who did it? Do you recall, Brad?

MACKENZIE: Oh, it would have been Berman Lippit. Lippit had left in the fall of 1961, and then Cole must have talked you into moving.

KREUDER: What form was the design in? Was it flowcharted or was it English text?

HALE: I think there was some sort of chart analogous to the ALGOL syntactical chart, but I don't think it was very appropriate or accurate. We did hardly any simulation for a couple of reasons. The MCP had to run before ALGOL could run. Obviously ALGOL had to run before we could run, so we were third man on the totem pole. There wasn't that much simulation time available. So I had no reason to know how much trouble we were in. From that experience, I have had an everlasting aversion to PERT charts. [Laughter.] Brad had this elaborate PERT chart with all the details of the compiler spelled out.

MACKENZIE: For what it's worth, I have an aversion to those, too.

HALE: It had everybody's estimate of how far along they were on it and everything looked great. When we finally got on the machine, the world fell apart. Nothing worked -- absolutely nothing. One of the reasons we alluded to the 5500 before: basically, the fundamental difference in the 5500 processor over the 5000 was the doubling of the PRT size. The primary reason for that was COBOL was so much bigger than what had originally been anticipated -- took so much more memory. We didn't finally get on a B 5000 until about October of 1962.

MACKENZIE: Yes, that's correct. The first limited exposure to it by our people was in the early weeks of October. The first really usage schedule really started about the end of the month.

HALE: If I remember correctly, the delivery of COBOL was supposed to be something like 3 months after ALGOL. Our first, what you could call really workable system, was really a year later, October of 1963. So we were much later than anybody else, much bigger.

ROSIN: Was the COBOL compiler the first major program ever compiled by the ALGOL compiler?

DAHM: No, the second.

ROSIN: Oh, the ALGOL compiler was. [Many individuals speaking.]

HALE: Well, the ALGOL people were compiling what they knew, obviously. COBOL was something that they didn't know. That was the first real test of the ALGOL claim, in my view. We had so many problems that if we had written that thing in assembly language, we'd probably be trying to get it to work today. One of the goals, which I don't recall hearing mentioned this morning was to have fast compilation. In those days compilations were horribly slow in any language. With sufficient memory, we could get up to something like 800 cards a minute in COBOL, which was just fantastic. Unfortunately, we forgot about the need for execution speed, object code. [Laughter.] It wasn't until users started running programs and comparing them to object code execution on IBM machines that I realized that somebody had made a terrible omission in decimal arithmetic on that machine. It was just terrible.

ROSIN: Was it ever rectified?

HALE: On a different machine, not on the 5500 or its successors today. It's still slow.

MACKENZIE: It was an oddball machine.

DENT: On the notion of decimal arithmetic, there are some decimal-to-octal conversion routine both ways, but there's no decimal arithmetic even today, I don't believe.

ROSIN: Can we shift over to hardware for a couple of minutes? I heard mentioned earlier that this was one of the first systems, maybe the first system, in which hardware design was strongly directed by the requirements of the software system. What was it like to bring that kind of system to fruition? It had never been done before, I imagine.

KREUDER: Well, getting it into spec form, of course, was a lot of work, but it had to be done. Once it was in specification form, then we could separate our efforts and go ahead and implement and then come back together later when we actually had a machine. Luckily for us, in engineering, at about that time our design automation system began to work very well, so that when we found a bug it was relatively easy to sweep the change in and get it incorporated into the formal documentation so that we didn't have really raggedy paperwork. So the very first one, and all after, were built with automated paperwork such that the real output was a deck of cards to be fed to the Gardner Denver wire wrap machine and it would turn the crank and produce the panels. So that saved us an immense amount of trouble. When working with Brad's guys, when we'd find a bug, it was very easy to get it fixed, and also easy to produce the paperwork to fix machines in the field. We were still finding bugs after 15 or 16 of them had been delivered. It isn't enough to fix the factory ones; you've got to get out there and incorporate that change in the field machines because they've got to be consistent among each other. There's just no such thing as a non-retroactive change; it affects the logic.

ROSIN: What was it like working on a system that was so radical? I always had the impression that hardware people, particularly in those days, took a great deal of delight in conjuring up new operation codes that might be useful. And here a bunch of people came from the other side.

KREUDER: I did my best to discourage my guys from doing that kind of stuff. We had a spec to implement and that's what we had to do. We wanted to make it economical and fast and get it done right.

ROSIN: Was there a lot of close cooperation and communication when necessary between software and hardware?

KREUDER: Oh, absolutely. As soon as we got right back together and began trying to run, we were in each other's hair all the time.

ROSIN: But before the reorganization, that was...?

CREECH: We didn't know each other.

KREUDER: Yes, that's right. Until we began...

MACKENZIE: For a long time, Product Planning, in a sense, was an intermediary between them.

CREECH: From my viewpoint, we were prohibited from knowing and talking to the hardware engineers until the organization changed. When we started running on the prototype and finding problems, we knew them as guys who came in and fixed the machines.

GALLER: There was a policy of separating the two groups?

ROSIN: No, he's saying he sees it that way.

KREUDER: Yes, there was no special effort to bring them together. It turned out...

MACKENZIE: You've got to remember that basically at that time, the programming system wasn't really being done as an engineering or even a company product, in most cases. Historically, people were putting out trace routines, assemblers were just starting to come in, and that kind of thing, but they were thought of as marketing aids rather than as an engineering product. This, in a sociological sense, was going on during that period, and there wasn't that much contact between the actual software implementors and those doing the hardware. Of course, there should have been.

CREECH: In my viewpoint, it like the two teams digging the tunnel, and they're so surprised and delighted when they meet. [Laughter.]

MACDONALD: That's only a half-mile off!

KREUDER: Well, we worked alike. It was such a treat when we could all work together on the same machine because we taught each other a lot of new things. It was a lot of work, but it was a lot of fun, too: making the whole thing perform.

GALLER: Was there good morale at that time? Did you feel like it was making progress?

KREUDER: About that time we all began to feel a whole lot better when we began to see those tapes move and hear the card reader click.

MACKENZIE: Well, the ALGOL certainly all had excellent morale and...

TURNER: Well, they were not allowed to not have good morale! [Laughter.]

MACKENZIE: I said something earlier today that I think bears repeating. Different groups of people perceive things differently, but there was certainly probably a more idealistic approach, to what the machine represented and what people wanted to achieve, mainly in the software group in a collective sense, than there would have been in the hardware people or the Product Planning people. The program people were really tied up in the thing in a systems sense. What they wanted it to do, and, in some cases, there were different notions maybe than was even intended.

HAUCK: It took a long time to, let me say, define specification, as was alluded to earlier this morning. The programmers didn't know what the machine was; the hardware people didn't know what the machine was; all of the operators had to be flowcharted on in detail before those specifications could be formulated. Another significant thing that happened on the 5500 system or the 5000 was that the entire machine was simulated. That was really a first for Burroughs; I think in the last 20 years or so, they never built a machine that was simulated to that extent again. That relates to the comment that Norm made about the design automation tools. Most of those things were generated right on the fly in line with the development of the system, the specifications of the system, so the design automation system was developed, the simulator was developed.

KREUDER: We pretty well had the logic bugs out of it before we ever put anything together. There are a lot of kinds of timing--race circumstances--that are almost impossible to find by simulation, but you can save those for later. We had a very solid, technology base at the electronic end. The circuitry had been pretty well proven on the B 200, and we knew what parts to specify. That sort of stuff was no real hassle to us at all.

HAUCK: Well, we got some surprises about midstream, as you remember. The logic designers (I think there were about seven of them on the processor) all went off in their corner and designed their logic; they would all use the same J count or driver's signal. Then when they got together and decided, "Well, now we're all going to merge our equations together," you know what we found out: it was three or four thousand loads on the signal. [Laughter.]

KREUDER: The point is we didn't have to build anything to find that out. In past incarnations, we would build it first and then wonder why it would suffer so.

HAUCK: But it delayed the program about 3 months, wasn't it? And we had to go back and develop some new circuitry to handle the added load requirements.

GALLER: Do I hear that the hardware technology was not pushing the state of the art? The architecture was what was new.

KREUDER: That's right. We had pretty well proven all the electronic aspects, at least as far as the logical circuitry. Memory was another matter, but all the stuff in the CPU was specifiable and anticipatable, and there were no surprises in there for us at all.

GALLER: One thing I realized during our lunch conversation, was that the disk we keep hearing about was in fact the head-per-track disk.

KREUDER: Yes, that's right.

GALLER: That there was not what we think of as the ordinary disk at any stage in this particular system. But that also it was not designed because of the need of their system. It was something Burroughs was doing, anyway, and was coopted in a sense.

KREUDER: To give author credit, that project had been started in Paoli as a research program, and they were about to drop it. Mac, or Bob, or Jack, perhaps, went back there and rescued it because we thought it would be "a good thing." We didn't really know what we were going to hook it onto, but we knew we ought to have one of the things.

MACKENZIE: Well, we needed it for the parallel access and virtual addressing.

KREUDER: Did you know that at the time?

MACKENZIE: I don't know. I know it now. [Laughter.]

KREUDER: I just thought it was a neat idea, and it would be a shame to let that project go down the tubes.

MACDONALD: I don't remember knowing much of anything at the time.

TURNER: I think is was sheer luck that that turned out to be such a fine backup storage.

MACDONALD: I think we knew at the time. Didn't we go through an iteration where we were looking at disk packs, with movable heads? And we started out with that big drum from Tireman [Avenue plant in Detroit] and we had to have head-per-track for low access time. And so I think it's legit.

CREECH: I thought that came out of the B 300 program.

TURNER: From a data standpoint, but we didn't know that as a system disk that having a head-per-track would be...

HAUCK: I think the first unit went out to the fishbowl; the first functioning disk unit went to the fishbowl.

TURNER: By the time we started working on our operating system, the thing had already been specified.

MACDONALD: That's correct.

TURNER: So we had no idea what could be...

HAUCK: That could be; my memory doesn't serve me.

GALLER: Erv, could you explain the word fishbowl?

HAUCK: The fishbowl was the room in the front of the building where the computer equipment would be installed--the showplace.

KREUDER: It had glass walls so that you could show off.

GALLER: But it was an internal Burroughs system.

HAUCK: Yes. In fact, the system that the disk file operating system was developed on was up there in the fishbowl.

MACKENZIE: Our system 102, I think, was installed in there at that time.

GALLER: As I recall, as a potential customer, it seems to me that the head-per-track disk was a very interesting device but seemed to be very expensive. And... So what I'm wondering...

TURNER: It was, and it was beautiful.

GALLER: And you found a good use for it in this system. Was it successful otherwise?

TURNER: Oh, yes.

MACDONALD: We built two manufacturing plants just to make it.

GALLER: What happened to it? Are you still building them?

HAUCK: There were big on-line systems delivered to the banks in England, which had floors with those head-per-track disks.

MACDONALD: Randolph Air Force Base in San Antonio, had all the Air Force personnel records on those things. It was a room that huge--200 feet square.

TURNER: We sold OEM (?); AT&T bought them.

KREUDER: General Electric bought them, too.

MACKENZIE: They probably got done in a technological sense and I think people could fabricate...

DAHM: I can relate to you an interesting incident about the sale of the head-per-track disks to General Electric. When I finished working on the disk file MCP in July of 1965, I left Burroughs and went to work for General Electric. When I got there, I kept trying to tell them about how great the head-per-track disk was, and I got told, well, it was either the world's most expensive disk or the world's slowest drum, and obviously it was of no interest to anybody. Then one day John Calure, who was the engineering manager there, walked in all smiles and said, "Dave, great news! Burroughs has just agreed to sell us head-per-track disks."

ROSIN: Let me ask a question about compromises. Oftentimes, I guess, when one takes a system from concept, to design, to implementation, to ship, compromises have to be made--often, I guess, for cost-performance reasons.

There are people here who represent various levels in the system, lowest-level hardware, or upper-level hardware, MCP, ALGOL compiler, on up. Were there compromises made at various levels that affected people at the next level, or did things sort of go along smoothly?

KREUDER: Well, neither of those is true. I don't think we made very many compromises. I think our idea at the time was that we could do it right, and it would probably be the less expensive way to do it. But that doesn't mean everything went smoothly.

ROSIN: Did you have to redesign hardware in order to meet the original design level?

KREUDER: Well, for instance, when we put that disk drive on there, then of course every one of those I/O channels, even the ones in the field, had to be retrofitted to take it. And Burroughs did that at their cost. They didn't try to pass it on to the customer.

TURNER: We didn't have very many systems out there, so it wasn't that big a deal.

KREUDER: Well, that's true.

TURNER: By then.

KREUDER: I suppose there was 13 or 14 weren't they?

MACKENZIE: There were probably 20 to 25 because I remember it was July to September 1964. When was the 5500 out, Norm? Do you remember when the first deliveries of that were?

KREUDER: No, I can't remember that.

MACKENZIE: It was in the summer of 1964, wasn't it?

TURNER: I can tell you when the first disk system was delivered, and I can tell you when it worked. That was the first week in January 1965. We went to NASA and installed our new operating system.

GALLER: I've been hearing references all day long to the fact that the upper management of Burroughs, back in Detroit, didn't really appreciate what was going on here. You guys were doing your own thing.

KREUDER: You better ask them that. Why ask us that?

GALLER: I should ask you that or I shouldn't?

KREUDER: No, you should ask them.

GALLER: Well, let me get your perception, and we'll ask it again in the other group. Did you perceive that as far as you were concerned, you had carte blanche to go ahead and do these things?

HAUCK: That was the view I had because we didn't have corporate groups coming in and overlooking the projects and trying to audit what was going on; everybody rolled their sleeves up and went to work.

TURNER: Well, within the marketing organization. It's my perception that Dick Baily and Jim Ford were solidly behind the program. And if they weren't, then they sure as hell didn't let us know they weren't. We were charging on, and from our standpoint, we got a clear signal that it was go. It was all the time go. And that's what we did. So, as far as the support that was required to do what had to be done...

KREUDER: We had all we needed.

TURNER: They were nervous and getting pressures from other places in the company, perhaps, but...

MACKENZIE: We didn't have the benefit of a strong corporate planning function resident in Detroit, for example, that would come out and review things or anything like that.

TURNER: Which was really good, actually.

MACKENZIE: There were decisions made at a surprisingly low operating management level, for example, and when we look back on it and think about the fact that we junked the ALGOL compiler, which was a very fine compiler, and decided to do another one, and when we did the disk file operating system, certainly BEAM was involved in that, but there was no big corporate decision about investing additional moneys for 6 months and doing a disk file operating system or anything like that. So, in that sense, the management decisions were made at a rather low level in Pasadena. Nowadays, those kinds of things, of course, would take appropriations, and there's a much more sophisticated system in place to prevent those sorts of abuses from happening again. [Laughter.]

KREUDER: The product line shows it, right?

ROSIN: Were there ever any incidents of new or modified requirements coming in from corporate at all? Affecting the program?

TURNER: From the Marketing organization.

ROSIN: Was that early on? When did those things occur?

TURNER: Well, the first recollection that I have is that people were complaining about the I/O language. It was very early in the program.

ROSIN: Before release.

TURNER: Yes.

MACKENZIE: It was our field marketing people.

TURNER: Yes, field marketing people would come in and tell us, as they got more involved in selling it and trying to figure out how everybody was going to program the crazy thing.

CREECH: I remember going in, for example, on a Saturday morning and meeting with a marketing guy from the field who had been complaining about the I/O facilities and asked, "OK, what do you want to do?" And working out of a syntax and saying, "Is that acceptable?" He said, "Yeah." I keypunched the changes to the compiler and put that construct in the language that morning before noon. And I was hung over, too. [Laughter.]

MACKENZIE: That's when you did your best work, Bob.

CREECH: I'm not exactly proud of that, but that was the kind of thing that we could do without anybody worrying about it.

MACDONALD: To make an organizational distinction, everything was approved but it was also highly delegated. That's the best way to put it.

OLIPHINT: We had pretty direct contact with our customers and our field marketing people. I was constantly out fixing bugs or talking to somebody. There were phone calls all the time, back and forth and...

MACKENZIE: One of the things, certainly, to remember, is that most of the people in Automatic Programming at the time of (?) had had field experience as users. Lloyd had, John Hale had, Bob Creech was a user, and the company was small. In that sense, there was a lot of contact with the technical sales effort. The thing that they're talking hearing from the field, they were really hearing from the technical sales people. We had some real shouting matches on those kinds of things, but it was good input. In fact, several of those people ultimately came in and worked in our group as very successful members. I remember Byron came in that way and...

OLIPHINT: Jim Ambler.

MACKENZIE: Jim Ambler. So, our growth in Automatic Programming was not by and large hiring people off the street. It was bringing people in from the field work.

DAHM: Which you might expect, since the organization was part of Marketing.

MACKENZIE: Yes, it was part of Marketing at the time.

DAHM: I never really thought about that before, but it probably has a lot to do with the strength of the group that there was a lot of field experience. Whereas today, most of our programming staff or people that we hire out of schools really have had no field experience.

MACKENZIE: It was not an inexperienced group in a programming sense. As we were saying earlier this morning, we lacked formal programming-system experience, and the advice we would get was by and large that we weren't going to be successful. So, consequently, we never really tried to recruit people that had programming-system credentials as such. But the people were, by and large, experienced programmers.

TURNER: Also, we had to be very careful because people that had their own ideas about how to do things, we had to retrain. I mean we did have some of that problem because they'd say, "Oh well, no. We're going to do it like we did it before," and there just wasn't much room for that.

GALLER: You were past that stage.

MACKENZIE: These guys had experience. Lloyd and Richard had done the Cardatron compiler on the 205. John Hale had been their manager. It sounded crazy, but they had done it and they had made their estimates. And this was at a time when people didn't make estimates come true in the programming-system business. It was kind of funny: there were three of them, and they had estimated they were going to finish the compiler in 17 man-months or something like that, and they finished it in 14 because one of the guys quit 3 months before the scheduled delivery date. [Laughter.] So they had experience in meeting their obligations, but it was the kind of experience that was really parochial to Burroughs.

CREECH: My biggest handicap was 5 years of programming experience in conventional systems.

MACKENZIE: He had to work it off.

CREECH: I remember when I realized that my problem, my struggles and learning what I was going to have to do in this job--work on this machine, do an ALGOL compiler--I had to forget everything I knew about computers and start from scratch. When I made that breakthrough, it got a lot better. I had to struggle.

MACKENZIE: For example, you had quite a number of years of experience as a programmer before you came into our group.

CREECH: That's right. I had 5 years and it damn near killed me.

[Many individuals talking at once.]

MACKENZIE: It wasn't just getting a bunch of people together to program the thing. The people had good programming backgrounds.

WAYCHOFF: Ken Meyers was in charge of the I/O system for the B 5000, and he was an experienced programmer (he may have had 10 years of experience). When he got on the B 5000 he thought, "My god, I've chosen the wrong field and I've wasted all this time because I can't understand this thing." He was really worried about selecting a new career.

ROSIN: Was the Automatic Programming organization always under Marketing? When and how did that change?

MACKENZIE: There was an Automatic Programming organization; Bob Barton had been the manager. My recollection was Bob started in the middle of 1959 or so when he joined Burroughs. He was in that position, as I said before, until May of June of 1960 when he moved with Clark and a number of other people into the Product Planning function. They didn't change what they were doing, really, but organizationally they moved into the Product Planning function. You were talking before about how decisions get made in Burroughs. The way I ended up being manager of Automatic Programming was that I was sitting in my office one day, and Bob Barton and Bill Lonergan came over, and we got to talking, and they said, "Do you want to be manager of Automatic Programming?" I needed a job at the time, and I said two things. I said, "What is it?" and "Yes."

GALLER: Later, Automatic Programming moved into Engineering.

MACDONALD: Actually, (Case Byron?) was the third one to come in. Yes, it was about August of 1964.

GALLER: Product Planning was no longer at that time.

MACKENZIE: I can't remember. Yes, it was there. Gene Lamar was the manager of Product Planning at that time. But they were not operationally in the implementation part of the thing at that stage.

GALLER: Why was it moved to Engineering?

MACDONALD: Because it was pretty clear that the two should be together as all one function, if anything should be software driven. So it became clear that that's what should happen.

MACKENZIE: Product Planning was there. They were working on the 3500 work at that time.

ROSIN: Didn't the separation of the Automatic Programming group from Marketing undo some of these positive inputs you had from the field?

TURNER: By then, it was almost done.

MACKENZIE: Over time it did, but not immediately because the people didn't change. It was still the same people that went from programming. But certainly after...

TAPE 3/SIDE 2

MACKENZIE: ...the notion of having programming-systems work done in engineering became institutionalized, then the people would get hired out of college to become programming-systems people and became in a sense part of the engineering discipline.

CREECH: It sort of insulated the architects from the users.

DENT: I think you're quite right. I think it had that effect, though it didn't become apparent for a while. By then the people that were doing the system software knew what Marketing needed, and proceeded to tell them what they needed for the next 15 years.

MACDONALD: It didn't leak off until perhaps the mid-1970s, or something like that.

GALLER: What should I be asking this group before we close it off in terms of what was happening then? What do you wish I would ask you so you could tell me? You want to say something?

RALPH PEARSON: What about programming documentation? The bane of all existence to programming.

GALLER: What about documentation?

PEARSON: I'd like to hear some comments.

MACKENZIE: Why would anybody want that? [Laughter.]

ROSIN: Who wrote the manuals, for example? Who wrote the user manuals?

MACKENZIE: We wrote the manuals. Well, there weren't too many, but we wrote some of them and such as they were, we wrote them in Programming Systems work and some of them later on were written in Sales Technical Services.

ROSIN: But the early ones with the initial releases were done by the same people who were developing the software?

MACKENZIE: No.

CREECH: I got most stuff learning ALGOL from a proof-copy of McCracken's forthcoming text on ALGOL that he sent to Burroughs because we were the only ones doing anything with it. I used that sucker very productively. I was trying to read the ALGOL 60 report when ALGOL was tough.

MACKENZIE: We had a big bugaboo about flowcharts. Flowcharts were the big rage. People always want to see your flowchart, and of course, we didn't have any by and large, or none that we wanted to give to anybody. Of course, that was one of the advantages, in a sense, of higher-level-language programming, You had a document that, even though you didn't have formal documentation in one sense, on the other hand you had code that people could read.

CREECH: I would like to say that there is a big misconception. We did have flowcharts. We had one of the most rigorous methodologies for developing the ALGOL compiler you could think of. There was a book kept in a central place that we all had access to--one book. It had flowcharts. Those flowcharts, when they were exploded to break down into further charts, were put on the chalkboard and cussed and discussed into the night to make sure that we all understood how everything worked and that it did work.

MACKENZIE: But we didn't want to publish it.

CREECH: The question of publishing was a different subject. We didn't have that problem. We were interested in getting the thing done and doing it in a nice fashion. And the flowcharts were there. Somebody still has that book somewhere.

WAYCHOFF: I do.

CREECH: It was invaluable. We had a very rigorous project methodology that we adhered to in that project.

WAYCHOFF: We played computer for hours on the blackboard writing ALGOL constructs through those flowcharts.

DENT: Just for the ALGOL compiler. I don't think that was the case with COBOL, and I'm pretty sure it wasn't on the drum MCP.

??	That's right.

MACKENZIE: One of the things, when you are asking people what people might like, is: the thrill that the ALGOL people had when the compiler compiled itself for the first time must have been overwhelming. And it meant something in a technological sense because they went from--what did it used to take to assemble that damn thing?

WAYCHOFF: Nine hours.

MACKENZIE: Nine hours to assemble it on the 220 to 4 minutes... compiling itself. When they had gone through the cycle, the generated code was more compact than the code that they had originally hand translated themselves.

CREECH: We thought that would put to rest once and for all the argument between assembly language programming and higher-level language programming because the same guys, the same four guys, that wrote the compiler, we wrote a version in assembler language to bootstrap ourselves. The compiler that generated itself, as Brad just said, when we got it compiling itself, beat us in both space and speed over our hand-tuned assembly-language version.

TURNER: With 4400 words before and 4200 words after.

GALLER: Unfortunately, as we know, the microprocessor people had to relearn some of these lessons years later.

TURNER: I think it's interesting: the ALGOL compiler was 9000 words, and the disk file MCP with all the functions in it that we delivered was 11,000 words.

MACKENZIE: And I shudder to tell you what that mother is now. [Laughter.]

TURNER: It's called creeping excellence.

GALLER: That's right. We were discussing, I guess last night, why things have gotten so big, and one reason why some of these have grown is that there are a lot more statistics taken and measurement in some of them, as well as function and checking and so forth.

KREUDER: Well, that format of compile time was absolutely astounding to anybody we talked to. In many cases, universities that were familiar with what it took to... well, it was just hopeless on a 360 of any size. This flexibility that Lonergan was talking about this morning, what we call active spares; if you had a 5500, you could back it up with any other 5500 that didn't have to have the same configuration. But, if you had a 360 and if they'd been done with one little bit of memory different, or one different tape configuration, then you couldn't back it up with another one. I always thought that was the dumbest god damn thing...

GALLER: And much of the industry has not learned that yet.

KREUDER: Yes. And there was no way to redo another SYSGEN because by that time they'd have the thing fixed.

DAHM: I tend to forget that, but that was one of the things that was really nice about the MCP. It went out and interrogated its environment, figured out how much memory was there and how much disk, how many tape units, how many card readers. Just figured out its whole environment and then used it. It never bothered anybody about telling it these things.

OLIPHINT: Even a drum MCP had that, didn't it?

DAHM: That's right. I said MCP; I didn't say disk file MCP.

GALLER: I presume Burroughs current systems still do these things, right?

CREECH: Oh yes.

GALLER: Okay. Brad?

MACKENZIE: I think that there were a couple of things that were really profound. One of the things that the work of the people did really illustrated was the value of taking time to think something out before they started trying to write code and the value of producing tools to do the job that they needed to have done. In fact, there were some extremely large applicational systems done afterward where I'm sure if they'd put more effort up front, in terms of designing the programming system tools that they needed for them, they would have ended up with manageable systems where they failed because of the very reason that they weren't manageable. And these people really did that. The business of writing a programming system in higher-level language, not only important technologically, but in a management sense, it was probably one of the reasons that they were able to be successful. Because one person could have a much broader span of control; there were fewer interactions between people; all those were really important. You were asking about compromises earlier, and I'm surprised Richard or somebody didn't bring it up. We all had a view of this in the card system, was as being said earlier this morning. There was a superfast 2000-card-a-minute reader and everything was nice about it, except every once in a while, it had a habit of reaching in and pulling the whole center portion out of the card. [Laughter.] And so out of desperation, they finally decided that if they could get their deck through the machine once, by god it was going to be on tape after that. I think it was mostly Richard who did this fine thing. They were very, very disappointed with the fact that I wasn't really excited about that. It was because our view of the system was as a card system. And of course, the way the system works nowadays, and as a matter of fact from that point on, was in that sense of being able to capture it on the way into the system.

GALLER: We all learned that lesson at some time during those 5 or 6 years.

MACKENZIE: That was one of the compromises, if you will. And I think just in a strict anecdote sense, one of the things that always amazed me, because I was extremely naive about a lot of things: I used to see the COBOL people running around with this test deck. Remember the COBOL test deck that Landreth used to have? They'd run around and they'd want to try something out, and somebody would run that damn test deck again. And finally one day I said to John, "Why do we use that test deck all the time? I mean my view is the COBOL people could run a COBOL test case when they need to or something like that." John looked at me kind of funny, and he said, "You know, you've got to remember. Nobody here knows COBOL from a programming point of view." [Laughter.] He said, "We're all ALGOL programmers." I thought, "My God, he's right." And I was pretty scared when I realized that.

GALLER: That's a question for marketing later.

MACKENZIE: Well, I remember it was a revelation to me. I hadn't thought about that. Do you remember that, John?

HALE: Yes, I'd forgotten.

DAHM: I'd like to say a few words about the two MCPs. There have been a lot of references to it today, and I'd like to give just a few minutes of my view of what happened. There were two MCPs done. The first one was done in assembly language, and it did not really have an adequate backing store, in that it was restricted to using these two drums, of 32,000 words each. The second operating system was not done in assembly language; it was done in a higher-level language, and we had an adequate backing store. It was a much more successful operating system. We got it done a lot more rapidly. It was more robust when it was finished. A lot of that can be contributed to the fact that the first operating system, in some sense, served as a prototype.

TURNER: Model.

DAHM: It was a model. We could look at things that were done, and we'd say, "Gee, that really worked badly. We'd better figure out some other way of doing that." Or we could say, "Oh, that worked out pretty well. There weren't any real problems with doing it that way. Why don't we just do it that way and not think about trying to invent a new mechanism." That's a very valuable sort of thing.

TURNER: It took us one year.

DAHM: It took us one year to do the second operating system. The first operating system basically took about 3 years.

CREECH: Pioneers are those guys with the arrows on their back.

DAHM: Right. It was only the second operating system that really lived up to the original promises. I'm reminded of something that Dr. Bob Johnson, who was the VP of Engineering eventually at Burroughs, used to say to me (this was some years later). He used to say that Burroughs was too small a company to not ship our research. Basically, the first MCP that we did was a research project, but we had to ship it to our customers. We couldn't afford not to. And the second one was the real production.

GALLER: You said it was only the 5500 that lived up to it. I would turn it around and say, "The 5500 lived up to it." I think at this point, we should close this session and give our marketing friends who have been waiting so patiently a chance.

BARTON: May I make a comment?

GALLER: Very short, please.

BARTON: If it were long, believe me, it wouldn't be important. I just listened very carefully with no axe to grind. I had contact with a fair amount of what was going on at the time. You heard something about as close to the straight truth as you can possibly get. I didn't hear a single thing (except some little details of who and what at what time) that wasn't correct. It was as good as this morning's session was bad. You've got a good thing for the Annals in this session.

GALLER: I have no doubt about that. Thank you.

BARTON: And I bet every one of you believes the same thing. You heard each other talking, and you believed the guy's saying pretty much what happened. Is that right?

ALL: Yes.

BARTON: Do you all have a good feeling about this? I think the meeting's been worthwhile. I'm glad I didn't go home after lunch.

GALLER: Thank you.

SESSION 3

GALLER: This is the third session of our discussion with Burroughs. We now have, in addition to a few people from the first two sessions, some people who were more directly involved with marketing. I'm now going to ask them to introduce themselves and say something about their role at the time. Jim?

JAMES FORD: My name is Jim Ford. I was with Burroughs from 1946 to 1965. At the time of the B 5000, I was computer sales manager responsible for sales promotion to the commercial, financial, and scientific markets; for sales training; for sales technical support, which included at that time the Automatic Programming and Applied Programming group in Detroit and Applied Science Group in Pasadena, and a Management Science group in Detroit. Basically, I was responsible for the introduction and sale of the B 200 and B 5000 series.

GALLER: Okay. Paul King we've already heard from. Brad MacKenzie we've already heard from. John Hale we've already heard from. Henri Berce is next.

HENRI BERCE: My name is Henri Berce. I was with at that time, Ohio Oil Company which was later on renamed Marathon Oil Company. I was a member of a team of two people charged with the responsibility of investigating the possibility of getting the Burroughs machine on board for the research facilities at Marathon Oil (or Ohio Oil at the time) in the Denver area. At the present time, I'm a consultant working, to a good extent, for the Burroughs Corporation.

JOSEPH T. HOOTMAN: I'm Joe Hootman. I joined the ElectroData Division of Burroughs in the summer of 1957 and left Burroughs in the summer of 1966. During the period of time that we're talking about, I had several roles. At the beginning, I was what we called a district computer specialist, which was a polite euphemism for salesman, in San Francisco. I was involved in the sales of systems to Stanford University and Stanford Research, UTC, and was on the sales team that sold the system to the First National Bank of San Jose. I went into Detroit to be part of the sales training team that trained, I guess by end count, about 220 people through that program on computer systems. I then headed up the fire department for a while, which was an interesting assignment; we'll get into some stories about that. Then I went to Los Angeles as the district computer products sales manager, and then I left the company. Today, I am the head of the information systems management consulting practice of Cresap, McCormick and Paget, the general management consulting division of Towers, Perrin, Forester & Crosby.

KING: God, you can remember all of that?

HOOTMAN: Yes. I've said it a few times.

PEARSON: I'm Ralph Pearson. I joined Burroughs in 1949, fresh out of Northwestern University--maybe not so fresh, I'm not so sure. I stayed with them until I joined a small company called ElectroData in Chicago. Bill Lonergan gave me the itch regarding computers when he came into Chicago to sell the G printer. Do any of you remember the G printer?

SEVERAL: Yes.

PEARSON: I took some training at Illinois Institute of Technology under Ned Chapin (remember Ned Chapin?), who used the 205 as his model to teach about computers. And Dean Holdiman hired me in Chicago. One week after I joined, a terrible, terrible day happened in ElectroData called Black Friday. A lot of people were laid off and said goodbye. Then several months later, Burroughs acquired ElectroData, and I was back working for Burroughs again. The second stint took me up to becoming a member of the B 5000 task force, which was called operation Leap Frog. At that time, Dean Holdiman had moved into Detroit, working for Jim Ford to head up the technical programming aspects of it, and Brad reported to Dean. Gordon Lovelace had come in from a branch manager in the east to head up the sales task force, again reporting to Jim. A fellow by the name of Bud Offinger moved from the Detroit branch operation into Chicago to take Dean's place. It was at that point that I was notified that it would be a good idea if I planned to spend a month and a half or two out in Pasadena on this task force. It was a delightful experience. I left Burroughs in 1966 of February, went with Computer Sciences, and have had a checkered past since then, until my present job. For 10 years I've worked for a company called Mead Data Central, which is a subsidiary of the Mead Corporation of Dayton, Ohio; it used to be known as the Mead Paper Company. And they market a tremendously large data base, about 7 billion characters on-line right now, of both legal information and news information in full text so that researchers literally can go in there and use any words or phrases they want to do their research. Several people in this room have used this system with their work. It's been a delightful experience, also. A very delightful experience.

GEORGE A. COLLINS: I am Al Collins. I have had several stints with ElectroData-Burroughs Corporation. The original one I guess was in 1957; I started with ElectroData in Dallas. Many illustrious people have had successful careers from the Dallas office. I didn't. [Laughter.] I left and rejoined Burroughs on numerous occasions. At the time of the crime, so to speak, I was the operations manager at Stanford University. I had just left Pasadena, and I was the operations manager at Stanford University. We were just installing the Burroughs 220 computer and the famous ALGOL. We kept very, very close contacts with people in Pasadena, and I guess Stanford was one of the very, very early university owners of a B 5000. I think we had machine number 6 or something of that nature. I'm now self employed at Collins and Associates in San Diego.

GALLER: You said the famous ALGOL. Is that BALGOL?

COLLINS: BALGOL, correct.

GALLER: I might mention that in addition to the people we just introduced, Bob Rosin is with me and Mondy Dana. I guess what I want to know is from the marketing point of view, you people were out there learning that there was a new machine coming along the way. How was it perceived from the marketing point of view? What was it going to do to you? Was it a solution to the problems that you'd had, or did it add to the problem?

FORD: Well, a dying man, a man dying of thirst, will grasp for anything. And I say that against the background of your earlier question as to what was the mood in Burroughs. Was this product market driven? I would have to say that in general, in the 1959-1961 era, the mood was utter chaos, and I speak from the standpoint of the sales organization.

ROSIN: In Burroughs?

FORD: In Burroughs, yes, and to some extent in the field in general, the business in general. But we had in our product line an electronic sorter which could be attached to nothing. We had a B 251, which was a ledger processor, which was a numeric machine which moved a lot of paper and which would make a great boat anchor in due time. The third thing was a tube computer the 220, and if you think selling tubes against transistors is an easy job, just try it sometime. So that was the product line going into the time of the B 200 and the B 5000. Needless to say, we clutched for them with open arms. And fortunately, they were excellent products. The B 200 family eventually became a fine machine. We were thrilled with the B 5000 because it offered something new, and Burroughs had been in a catch-up mode for so long that to have a product which genuinely, we felt, leap-frogged, was a wonderful new experience. So we were very enthusiastic about it. As has been said earlier, we gave full support to it. Its newness was a two-edged sword because it wasn't until IBM blessed multiprocessing that we were really considered legitimate. A few adventuresome people bought our product before IBM said it was okay, but they were mostly users and people who had had a good experience with Burroughs for one reason or another. So the market was somewhat limited until it became a generally accepted mode of operation. You asked, how did the field organization feel about it? We were thrilled and anxious to have it, and there were a lot of interesting experiences I'm sure which these gentlemen can tell you about.

ROSIN: Jim? Could I just follow up for a second? When did Marketing formally get involved in the project?

FORD: That's an interesting point because Burroughs, like many other companies (and I go back to 1946, as you know), was a manufacturing, engineering-driven company. I was brought into the home office as a sales promotion representative in the financial field, and the way I would find out about new products would be to be invited into Engineering to see something and to see how wonderful it was. Well, very often it wasn't that wonderful. And so you had to say what you thought and then pretty soon you found out that was construed as being negative. [Laughter.] So in the interest of survival, you began to say, "Gee, that's great, and how soon can we have it?" [Laughter.] Les Clausen, who was the financial sales manager reporting to me, and I were invited down to Paoli in a big convoy including Eppert and MacDonald, and a whole flock of people from Detroit. We went down to see this machine and really couldn't say too much about it while we were there because they were all so enthused about it. So when we got back, Les and I said that what the market really needs is a low-cost, transistorized, alphanumeric system. We were told by Product Planning, "There is no such thing." Of course, 3 months later the 1400 was announced. It was an M&E-driven company. I think when Bill Lonergan and Paul King with field experience came into the picture, that marketing by proxy was better represented than it had been in the past. But up until that time, it was largely Engineering deciding what they could do and somebody deciding what the market wanted. Then the sales department was told, "This is what you have. Isn't it great?"

ROSIN: Did you have to prepare to deliver this to the market? There must have been work for you all to do.

FORD: Oh, an enormous amount of work. As I say, the work in connection with the B 200 and the B 5000 was a labor of love because we really wanted those products. The Burroughs organization, in general, was much more closely attuned to the B 200 series because it was a business data processing system. Of course, they grabbed hold of that quite well. You have to understand another thing about Burroughs at the time: it was financially strapped. So, in addition to facing Goliath, we did so with very meager resources. This is why the wonderful job that our Automatic Programming group did with a small number of people was so greatly appreciated. In anyplace else, that probably would have cost two or three times (or maybe more) what it cost in Burroughs. But, fortunately, they were able to do their job with a limited number of people and do it very well so that we were able to finally get the product out.

HOOTMAN: I think there's another thing, Jim, that should be part of the background of this which has to do with the ElectroData Division/Burroughs Division thing. Whereas the people in Pasadena were relatively isolated from some of the Detroit influences, the opposite turned out to be the case with the field marketing organization. And I think (Ralph, you alluded to this) it was something like May of 1960 that there was an announcement that the Burroughs Division, which was the accounting machine/adding machine division, and the ElectroData division, which had the 205 and 220, were to be merged. An effect of that was kind of interesting because many of the senior ElectroData people rather promptly left the company. I remember in San Francisco as a salesman I was reporting to Ray Eppert because three people between me and Ray had left the company. We lost a lot of good marketing people and, at the same time, these products were coming down the line. I remember at this CUBE meeting, sitting down with Paul King and Dick Smith and I said in so many very blunt words, "I'm going to walk out the door along with the rest of those guys. Tell me what's coming down the pipe." I put Paul and Dick in a very awkward spot, but they gave me some broad hints. I guess at that point in time we were talking about the 4400. I made an interesting decision; some of the rest of us did--I guess Ralph was aware of all this as well. Some of us decided to stay on the strength that the company was going to need people who did understand digital computers and could get out there and market them. Within a few months, I was under medical treatment for an ulcer because what Burroughs did to me is, they sent me up into a classroom and said, "We're going to begin with a hand-operated adding machine, and you are going to learn these products, and you are going to go out and sell them." Three months later I was given a territory and I was told, "You've got all the insurance companies in San Francisco. Go out and sell." And I went out and sold adding machines. If it hadn't been for this product stuff in the pipeline, I wouldn't have been there; I would have been off somewhere else. But I think that the impact, the difference between the Manufacturing and Engineering side and the Marketing side is part of the background of this, because when those products did come out, we were, in a marketing sense, shorthanded. We didn't have the field representation that we had with ElectroData, and we had a hell of a time getting organized and getting people in the field who could effectively present and represent the products. You said something, Bernie, this morning about the sales response. We suffered in some cases from people who did not have good sales and good technical ability in the field.

GALLER: I understand that Paul King ran a short course for field people. Maybe you could tell us about that, Paul, in terms of what backgrounds did they have?

KING: Well, I don't remember who made the selections. Was it Dean that picked the people? Somebody picked a group of people from throughout the field to come in to be the nucleus of preparing all the salesman literature, how we were going to sell this thing, everything. I think Ralph's the only here who was in it. Stan Buchman was in it, Dean Earnest.

PEARSON: I think I'm the only one in this room.

KING: Yes. There were a group of people that were brought out to Pasadena for a period of...

PEARSON: Almost 2 months.

KING: I think it you ended up in the middle of December and you came out in the middle of October.

PEARSON: Yes, the latter part of October.

KING: They were brought in, not sure of what they were going to do. They were going out to Pasadena to hear about this new product...

GALLER: Did they generally have computer backgrounds?

KING: Oh, yes. They were the cream of the crop from the field. It was made up of partly sales tech reps and partly salesmen. Both kinds. Because they broke into two teams. We took two classrooms and made in into one. They were in there, and this was my first attempt at trying to talk to people that didn't know anything about this thing or what was coming. I started talking about operating systems, MCP, ALGOL, Polish strings, stacks, etc., etc., etc. For the first two days, here were the best people out of the field, with completely blank looks in their face. They couldn't understand this. Then on the third day, you started seeing the lights go on. One by one they all got it, and then it was fine. But first it was distressing. At the end, they expressed the same thing: that they understood it, but they couldn't explain it. Now, after doing it a few times, you sort of get comfortable with it, but this was my first attempt. Then they went on and produced the literature that you see. The concept manual, which I hope you can get a copy of, the descriptor, and things like that. Then they broke up into teams and a number of those people, the sales tech reps, I believe stayed on. I'd say four or five of them stayed in Pasadena and moved into Automatic Programming.

GALLER: There was no machine yet at this time?

KING: No.

??   They wouldn't tell us that.

KING: So there was no machine, and the rest of the field didn't know about it. It was just this one group. The rest of the field didn't hear about it until two weeks before the announcement. Ralph made the announcement in New York. I remember we had a meeting that Bill Lonergan and I went to. It was two weeks before that where the rest of the field heard about it--not all of them, but a few people from every office came in. Some of them were sort of horrified about what we were going to do to them. That we'd never be able to sell this thing, but it was too late.

GALLER: How did you help those people learn enough to sell the machine later?

KING: We didn't.

HOOTMAN: Some of them never learned.

ROSIN: Is this the famous task force we're talking about, now? Do you want to tell us how that was formed? What kind of people were on it and what your assignment was?

PEARSON: I believe there were ten of us.

FORD: I selected the people that were on the task force.

PEARSON: I believe eight of them were tech reps, and two of us were sales reps. Welton and myself.

KING: And Gordon? Well, Gordon didn't really participate.

PEARSON: Gordon was kind of our leader, but he was running back and forth from Detroit.

KING: He was great.

PEARSON: Yes. And I take Bob Creech's point. Here I was a fairly successful sales rep in the field of serial processors, 205 and 220 (that's all I understood), and for two days I sat there in utter amazement listening to this Niagara of words come out of Paul King and trying to figure out, "My God, what the hell is he talking about?" But, as he says, about the third day dawn began to break, and then over the period of the next few weeks, we finally got to the point where in the last couple of weeks we were actually to do something productive and get something out.

KING: I remember you and Greg used to make sales presentations to each other.

PEARSON: That's right. Or anyone else who would listen. [Laughter.] There weren't very many people who would listen at that time; they thought we were bumbling idiots. I was in somewhat of a unique position. Leading up to this point, you must recognize that at one time, and for quite a period of time, maybe 3 or 4 years, within Burroughs there were at least two sales organizations. There was a huge sales organization which sold adding machines, calculators, bookkeeping machines, accounting machines, and so forth. I was of that genre. When I joined Burroughs, I spent about 8 years doing that and then switched over to this elite group of snobs who went out and marketed computers--these wonderful things. To a great degree, there was animosity between those two sales organizations. Now, when we come up to this B 5000 thing, many of the technically trained and competent computer salespeople had split, and the people who were supposed to end up selling this equipment, were the adding machine salespeople. Well, here I am with 3 or 4 years of experience in selling computers, having difficulty understanding the 5500. How are you supposed to impart that knowledge to a group of people who hardly know how to spell the word computer, much less what it is?

FORD: You know, Ralph, I have to differ with you with the fact that adding machine people were expected to sell this machine. I don't think that was ever the case. The Burroughs top-line salesmen were extremely knowledgeable in systems.

PEARSON: Oh, yes.

FORD: Very knowledgeable in banking systems, insurance systems, commercial systems, and they were simply learning to apply a new tool to what they already knew. So it wasn't adding machine salesmen. It was very knowledgeable systems people...

PEARSON: Systems people, yes. But they did not know...

FORD: ...who were learning a new tool. In many cases most of the customers that they were calling on (this was the saving grace) didn't know quite as much as we did. We maybe had read one more chapter than they had. That kept us alive because we knew the buzzwords, and of course we had a sales technical organization. It was team selling, very much. Everybody on the team was called in when the situation called for it. I mean, Lloyd was called in to selling jobs, and other men were. The president was called in, and anybody who could help us get an order. So it was very much team selling. We built on the systems knowledge of a large number of people; some of them obviously picked it up more quickly and were comfortable with it, anxious to learn because they knew this was the direction in which the technology was going. They looked upon it as an opportunity to rescue their careers from total decline. So, it wasn't all that bad. It just took time, and effort, and persistence, and a lot of loyalty, and a willingness to take a chance.

GALLER: Weren't a lot of your early customers of the B 5000 people who already had the 205 or the 220?

FORD: Sure.

GALLER: So they were computer oriented.

FORD: Yes, the people that worked with them. We had a pretty sizable field engineering organization. We were building on the base that Burroughs acquired from ElectroData and broadening it into the general products sales organization. Certain people emerged as being interested and capable of learning the new technology, and it didn't take them too long. Of course, you really learn when you make an installation. That's how you learn, really. The rest of it is theoretical; as soon as you start making some installations, you rapidly get pretty professional, the hard way.

GALLER: Speaking of installations, the machine was announced in early 1961. They weren't delivered for some period of time. You really didn't have one to show people for a long time. And even when they were delivered, you didn't have the software really ready to use for some time. What was that interim period like, when you were selling a machine that wasn't?

FORD: Pure hell. [Laughter.] That's all I can tell you. It was pure hell. And even after it was delivered it was pure hell because it didn't work too well. [Laughter.] Most of Burroughs products, and I think this is maybe still true, are delivered before they are ready. This is why the company had suits and all of that stuff. We talked about what drives a company. Well, basically it was urgency. Urgency. And when you're operating under that mode, you get some products out in the field sometimes before they're ready, and that's the way it was.

I have to tell you a story that tied in with this morning. I got a call from Dean Holdiman who was in Pasadena, and he said, "You'd better come out here." That's about all he would say. So I went out and I learned that the compilers would not fit on one drum. The machine had been sold as a one-drum machine. So I had the job of going back to Detroit and telling management that we were going to have to supply a second drum gratis. So I flew back to Detroit on a Friday night and called Bement Saturday morning, and told him the news that there would be about at least 20 drums that we were going to have to supply gratis because there was no way with the delivery delays, etc., that we were going to get people to pay for another drum. The dollars, as I recall, were about between $200,000 to 300,000, which is, let's say, worth about a million and a half today.

GALLER: That's the total, though.

FORD: Yes, that was the cost of these additional drums. Bear in mind that Burroughs was so financially strapped, they could barely cover their dividend. They had this enormous expense of putting rental machines in the field when they were starting up in the data processing business. The enormous expense of R&D and engineering and all of that not producing any revenue. It was a very difficult time financially. So, Bement said, "Well, let me get hold of Ray and come over this afternoon. We'll meet down in my activities room and talk about this in a quiet atmosphere." Which was brilliant psychology because we got over there, and to Mr. Eppert's everlasting credit (I had told him about this group of dedicated, bright, hardworking software people who were doing the very best they could), and he said, "Well, if that's what we have to do, that's what we have to do." This was a man that was under enormous pressure because of the profit position, the stock position, etc.--all the things that a CEO goes through. Of course, I half expected to be fired because I was responsible for this. In any event, he rose to the occasion. That also was the birthing of the B 5500. Because we knew we couldn't go on delivering drums for free. So what we did was to change the number, increase the specification to two drums, add a few other goodies, call it the 5500, and went on from there.

TAPE 4/SIDE 1

FORD: I bring up the point strictly to illustrate the sort of top-level support which the data processing program received from Mr. Eppert during his tenure. I think there was another turning point in the life of the B 5000 program that occurred later when the program was going to be discontinued. I guess the old saying is, "it's better to be lucky than smart." It was discovered when the idea of discontinuing the 5000 was under analysis that it would cost more to discontinue it then to keep the plant going. That single fact kept the B 5000 alive until IBM started to sell multiprocessing and the sales of the B 5000 started to reach a higher level. Saved by the bell!

ROSIN: What year was that decision?

FORD: That was about 1966.

HOOTMAN: Bernie, you asked a question about the timing of things, and I think it's important to look at how this thing evolved and kind of set a stage. We've talked a little bit about the task force which was a stage and the product of that. We have some materials here; there was a set of about 100 viewgraph cells, and these have an important place. The really difficult thing (Paul alluded to it) was teaching this thing to somebody. The visual aids that the task force produced were extraordinarily helpful in this regard. There was another terrible problem with this whole thing in that some of these inventions (and it came up this morning); Barton could describe what they were, but he didn't have a label for them--descriptor and that kind of thing. All of these semantics that were associated with the product. If you said to somebody "program invariants," it didn't mean anything to them. So the task force did an absolutely remarkable job of taking this thing and in a way translating it and packaging it into something that could be understood. This was an extremely critical phase of the thing. There was a phase that followed that that paralleled this period of software development, where a few of us were brought together and trained (some by the task force and some by the other), and were set out in the field. The product in that sense was not released to the whole field marketing organization. I guess there were some of us that were assigned to go out and work on it. We did team selling, as Jim said. I had a sales technical representative, a chap named Rick Truitt, who is now gone. We went out and hit the road. One of the interesting things that came out of this (there were other teams doing this elsewhere around the country), we went out with a set of viewgraph cells and we arranged for presentations. As a by-product of the story (Lloyd and I were talking just before the break), one of those was given to a group from Lockheed Missiles and Space. Several of those people subsequently joined the company on the strength of that. Another one of those presentations was given to the GE atomic power equipment people. And some of those people came, and out of that came the connection that provided the place for the simulation to be done. So it was not only good from the point of view of getting our light out from under the bushel basket, but also we did some pretty effective recruiting and got some very talented people as a result of that. Then in the middle of 1962, Jim [Ford] set up a sales training group under Harry Butts. I came in, and several others came in--people who had very strong backgrounds. Then we began to very aggressively train the balance of the field marketing organization in the product. I think we had those people for 2 weeks at a crack in Pasadena. The team was down there about 13 weeks altogether training all these people. Things that one hears about the Compilogram, which was a device to teach these guys what a compiler did. It was kind of a game that was put together. We exposed them. We did it in Pasadena because that allowed access to the development people. Jim made the comment (and it's really, really important here, it's one of the reasons that the guys that talked this morning were very much a part of the sales team). We took them out to prospective customers. Norm was asking me this morning, "Do you remember the time you took me up to UC Santa Barbara?" Lloyd and I made, God, I don't know how many calls on different places. I used to grab him every time he'd come up to Stanford: "Come on. Let's go make a call somewhere." They were very much a part of the team, and the marketing people began to understand these people were a resource. I think it presented a problem to Brad in that sometimes we asked more of their time than we should have, and they had jobs to do. But they were just as anxious to help us do the work. I think the key part [was when] then we moved on into the phase where we had a pretty good representative set of salesmen out there who were selling both the 200 series products and the 5500. Then it started to take off.

FORD: I think it's worth noting that the general products line salesmen, many of whom had excellent contacts in many large organizations in the country, served as bird dogs for the data processing sales teams. There were twelve districts in the Business Machines Group, and each district had a data processing district sales manager, a sales technical manager, and a field engineering manager. They had other people working for them. So, Named Account salesmen, who might have been calling on X bank for many years and had excellent contacts, could bring this team in in order to make the sale. And then, over time, more and more men were trained and could function on their own. But it was an evolutionary process and I think it was managed reasonably well. One of the interesting things was selling the B 200. The banks were controlling Burroughs to a very great extent because of its debt level, and we were obliged to sell 2-year lease contracts on the B 200. Now, not only did we have to sell against IBM, but also against their 1-year contract at a time when everything was, "Gee whiz; we'll probably be changing this thing tomorrow because of the rapidly changing technology." Getting somebody to sign a 2-year lease took some real selling.

GALLER: That's very interesting. Could we get some reactions now from our user force here? Henri, you were a Burroughs customer.

BERCE: You mean before that time?

GALLER: Before the B 5000.

BERCE: Yes, yes.

GALLER: And then they came in and told you about this new machine. What was your reaction to it?

BERCE: They didn't come and tell us about this new machine. We went and asked them about this machine, and we didn't get any answers. [Laughter.]

GALLER: That's first phase. Then?

BERCE: Then we had to do some pretty damn hard work to find out on our own, with some of those gentlemen's help here, to find out what the machine was all about.

ROSIN: What was your reaction to it when you did find out?

BERCE: Our initial reaction to it, as soon as we found out the slightest idea what might be in the wind, was very favorable. If it wasn't favorable, we wouldn't have pursued the idea any further than that.

GALLER: Your group was doing scientific work on the machine.

BERCE: Yes. Our research facility is not concerned with anything else but technical work.

GALLER: You were already using BALGOL.

BERCE: This may sound funny to you, but Brad can back me up on that. We were using toward the very end, toward the last months of the life of the 205, we were using ALGOL on the 205. The compiler was written by Don Knuth. It was given to Brad MacKenzie. Brad, when Don Danar and I were in your office once, you reached up to the shelf and said, "Here is something funny that you may want to play around with." I said, "What is it?" "It's an ALGOL compiler for the 205. And it was paper tape reels. By the way, it took us better than 2 days to get it ready for the machine. Not because it took so long but because you have to read it in one full swoop and if there was an error in the read some place, you had to repeat it, you had to start from scratch. So it took us I don't know how many tries until we finally got the damn thing to read all the way into the machine. We had reasonably successfully played around with it. It wasn't a production tool, but it was an experimental tool. But at that time we knew already that if the machine turned out to be anything that was remotely usable to our requirements we would probably dive into the 5000 thing. So, we said we might just as well go and get some experience in ALGOL since we're going to be using

ALGOL on the new machine. That was our first experience with ALGOL.

GALLER: Was there higher-level language available at all on the 205 before then?

BERCE: No, no. That was machine coded.

MACKENZIE: At the same time, there was a Cardatron compiler that was mentioned earlier this morning as well. I can't recall, I don't think there was any assembler.

BERCE: No, no.

GALLER: But then you were told about the machine. It looked good to you, right? You were ready to order one?

BERCE: The machine looked good to us before the specs were announced. We had means of getting inside information from a number of people, including Bob. We used to sit around at night trying to figure out whether it was really likely to do anything remotely like what the people were saying about it originally. We finally figured out that yes, indeed, it's just got to do it, it can't fail. Of course, you may say that there was a lot of trust on our part in that, but I think history bore us out.

ROSIN: How much time was there between the time you first learned something about the system that you felt was meaningful and you actually received delivery of the system?

BERCE: About 14-15 months. Is that about right?

MACKENZIE: If I remember, you got the machine in October or something like that.

GALLER: And how long was it from that October delivery until it was doing useful work for you?

BERCE: How do you want to define useful work? One hundred percent or 10 percent? 10%?

GALLER: I'm sure 10 percent came before 100 percent.

BERCE: Some useful work almost immediately--after the machine was installed. There were a lot of problems that we had. There were a lot of things that we were told were going to be available but were not--for example, a three-stage backup from primary memory to secondary memory to tape, which of course didn't work. We made the discovery pretty soon that even the two drums were not sufficient to do a lot of things that we wanted to do and things like that. After we became aware of the limitations of it, we just said, "Well, that's the way it is and that's the way we've got to (adjust our research?) and to reset our constraints."

GALLER: Did you in fact pay rent as soon as you got it, or was there some period where because it wasn't working that well...

BERCE: I think (and I don't know; maybe Brad knows about that) it may have been a month or two.

MACKENZIE: It was certainly on rental before the end of the year.

BERCE: Yes. It may have been a month or two. I know we got it by the end of October.

GALLER: One other question. Do you regard your installation and your reaction as reasonably typical of, at least, scientific installations or all installations? You don't sound terribly unhappy with what happened. You seem to have had great patience.

BERCE: No. What happened with the 5000: we were not terribly unhappy because we were pretty well aware of the potential possibility that things were not going to turn out as rosy as they were represented by the Burroughs organization. We were, on the other hand, very, very happy to get out from under the terrible constraints of the 205. We had to do work with the machine. And since that was the machine and since with all its initial deficiencies, it was still immensely better than what we had with the 205. Naturally in that respect we were very happy with it.

GALLER: How about something like the IBM 7090. Did you compare machines at that time?

BERCE: Oh yes, indeed. We didn't want a 7090 primarily because we wanted a machine that was going to be as liberal, as broad, as far as the user community is concerned, as possible because the research facility, as it was constituted, did not contain a programming staff. Every scientist, every mathematician, whatever have you, had to do his own programming one way or another. Or, alternatively, con somebody else into doing some programming for him on his own. The answer to that kind of a situation, that kind of an environment, was a machine that was going to be as liberal, as forgiving, if you wish, as possible. We could not possibly see the 7090 in that light.

GALLER: Joe, you wanted to say something?

HOOTMAN: Yes. It's a generalization to say that most of the early 5000 customers were already Burroughs users. But I think there's something more interesting behind it than that. The Burroughs product had generally been sold to (if I can say a user has a personality type, I think that's a way to get at it) organizations that were reasonably self-confident and that were not capable really of being intimidated (and Henri kind of articulated some of that) by support problems or by technical problems. We found out so clearly in making those early rounds of presentations, that the very hard-core, IBM-dependent (you know, the having their diaper changed kind of thing) people just absolutely had no interest at all. So the community of Burroughs users was then, at the 5000 time, and I think has continued to be, people who are, in a sense, relatively strong, relatively self-sufficient; they know the game, and they can play it within the constraints of some of these things. The tolerance that was shown by the early users, in that first year or so was remarkable. They saw through the goals; the kind of objectives that were articulated this morning were things they could understand. If you talk about competing with the 7090, one of the interesting things that was going on out here on the West coast was a lot of aerospace. There were a hell of a lot of 7090s around. This brings up the FAP [FORTRAN Assembly Program] issue again. We were trying our damnedest to get into those places, but a lot of the 7090 applications were brute-force speed kinds of things, and they had a lot of data reduction stuff in there that had been done with stuff that we couldn't even get at. This is what perpetrated a lot of that thing. Interestingly enough, after I'd been to Detroit and I came back to Los Angeles, I simply put out an edict, and Ralph probably remembers, that we're not going to sell to the aerospace industry. I said, flat out, any salesman who goes out and calls on aerospace companies is going to get fired. There was a hell of a brouhaha over that. I said, "You guys go out and sell to people who know Burroughs [back to Jim's point], where we have the contacts and connections. Get into the savings and loans, get into the banks, and knock off this stuff." We couldn't compete benchmark-wise against the 7094 where they wanted to run brute force and we couldn't convert their programs. A lot of these guys--you could talk about multiprocessing, and they didn't, they frankly didn't give a rat's ass. That wasn't what they wanted. So, we learned fairly quickly, I think, to drop those people. The emphasis then shifted to those kinds of environments where it was really good to be able to do several things at once and where people could understand the facility of doing that with this machine.

BERCE: I think, to answer your initial question as to the comparison between the 7090 and the 5000 on a technical level, the overriding consideration, in our case, at least, was the fact that it had a 48 bit word, which the 7090 could not offer. We thought at the time, and later proved ourselves right in that thinking, that we couldn't possibly live with any less than 48 bits.

ROSIN: For floating-point precision, or integer precision?

BERCE: Floating point; more than 90 percent of our work was floating point.

MACKENZIE: As I recall it, the dynamic storage allocation made a lot of difference to people.

BERCE: Yes, it did.

ROSIN: I want to ask one question of Henri and Brad. Brad, you were in the development organization, and Henri keeps referring to you almost as though you were his direct salesman.

MACKENZIE: Oh, no. I wasn't a salesman, but I'll tell you we became you might almost say intimately acquainted with all of the users during that first year, even to the point of going to special CUBE [Cooperating Users of Burroughs Equipment] meetings which were held to discuss whether there was every going to be things. We got to know them all pretty well. Many of these early customers were as caught up in the spirit of what the 5000 was trying to do as we were ourselves. They had their jobs on the line also. A lot of these people did - there was just no question about that. In fact unfortunately, some of those supporters got hurt in the process.

HOOTMAN: But remember what was said this morning: so many of the people in this Pasadena group had prior field experience. To be very blunt about it, the field marketing organization had a lot of respect for those people, and we trusted them. We didn't try and isolate... these were not skeletons that were kept in the back closet, where we were afraid that we could take somebody out and they'd blow a sale for us. We didn't. We had the opposite kind of relationship. So there was no problem. In fact, if a prospect said, "Gee, we want to talk to somebody in engineering," we said, "What do you want to talk about? We'll get you the name!"

GALLER: Also, it didn't hurt that Automatic Programming was in Marketing.

HOOTMAN: I don't think that would have made any difference.

GALLER: Al Collins? How about what it looked like from Stanford's point of view?

COLLINS: Well, we had a very special arrangement or something--one of these cultural things, like I could call up Creech and speak Texan with him. [Laughter.] Let's see, John Hale had hired me, and one of you guys down there fired me. I don't care which one it was. [Laughter.] It was very interesting. At Stanford at that time, you know, they mumbled something about a simulation machine. In some ways, we had the most interesting simulation of the B 5000 whatsoever in that we had a 220 at the time running the BALGOL compiler. I wrote a very small operating system for the 220, a few hundred words, and with all respect to the University of Michigan and MAD and your 7090, we were probably outproducing you by a factor of 4 with a machine that was a factor of 100 smaller and slower. It was probably one of the most successful job shops ever around. We had very close contacts with these guys. They would come up and we'd go eat Mexican food. Dave Dahm came up and gave a class on how the compiler really worked. George Forsythe--this whole sort of technical numerical staff at Stanford were very, very taken with the idea of ALGOL, and it was phenomenally successful. Versus the 7090--a 7090 got sort of "drug in" over our screaming, kicking objections. I hated the thing. Nobody on the staff wanted it, but, somebody gave it to the university. I had very careful records about compile times and execution times on the 220 versus the 7090. We could document without a doubt that the 7090 with FORTRAN, and optimization, and the whole thing, could not compile fast enough per day to handle half the workload that we were doing on the roughly 150-microsecond 10K 220. We had no object code. We had no binary code. Everybody walked around with their deck and ran it through the machine; essentially the machine ran at card-read speed all day long. We had considerable experience in that every time Brad would say, "We've got a new compiler coming up," we didn't really have to worry about, "Well, we've got to convert some code or whatever." Anything they brought up, we'd hang it on the line, and get on the phone to Pasadena, and tell him a lot of things about it--literally within a matter of hours. The shop was totally self-adapting. It just kept moving up, moving up, moving up, and moving up. Now, that was all on the 220, and I'm not sure when we found out about the 5000. We might have found out about it considerably earlier than some of the people in Burroughs. Their security was terrible, you know. We knew a great deal about it through personal friendships and professional relationships. There's an interesting aspect: I don't think anybody was really trying to keep anything secret at that time. I mean, very, very early in the project, people were more than happy to talk about almost anything about it. The BALGOL compiler itself had stacks in it and it did Polish notation, all of that sort of thing, so we had an extremely good feel for what this machine would do when you got around to building it in hardware rather than building it in software. So, I think it was a case that we were as excited about the machine as Burroughs was and had very good relations and some fairly stormy relationships with people, but there was no question whatsoever in our minds that that was the machine we had to have. We had the 220; we got a 7090. I made an aborted attempt to get a KDF 9 from English Electric, which was the other interesting stack machine at the time. But there was just absolutely no question in our minds that it was going to be out come hell or high water. There was a lot of both, but we got it and stuck the thing in, went through the normal teething pains with it, and it was very interesting to see the thing sitting side by side with Mr. Watson's special. It was just no question, absolutely no question as to what the performance was.

GALLER: And yet, the machine didn't sell very well for a number of years.

COLLINS: I think, with all due respect to the marketing people here (and I was president of the users group at the time and going off to CUBE meetings), there were a lot more Burroughs machines bought than sold. Considerably more Burroughs machines were bought than sold. It was a great idea, and I think the people that knew were quite willing to go with it, but it wasn't painted blue. You looked at the mechanical specifications--the memory, etc.--and they looked terrible. It never came up this morning about the idea of the memory and the I/O channels being interleaved so you had about a six-way access to a relatively slow core memory. In effect, it by far outperformed machines with much, much faster hardware. So why it didn't take over the world commercially is hard to say, but it...

FORD: Can I answer that?

COLLINS: Sure.

FORD: It took courageous people to buy the B 5000 for a couple of reasons. One, it was an advanced concept which IBM hadn't blessed, and also Burroughs's financial condition was such that people wondered (and IBM sold the idea) if it might not survive. It was expected to be a casualty. People don't like to buy machines that might be orphaned. So that was one of the pervasive things in the marketplace that deterred some people who might have liked to have bought the B 5000 on the basis of its merits from doing so. There was a very pervasive question as to whether or not Burroughs was going to be a survivor. RCA had just come on--huge corporation; GE had come on, Honeywell had come on, and here was Burroughs. Well, you've heard of the seven dwarfs, so I don't position it in that group.

ROSIN: I want to ask Bernie Galler a question. Bernie, you were a potential customer. As a matter of fact, I recall a visit from Burroughs Detroit to the University of Michigan at one point, I think well after introduction, suggesting that the University of Michigan ought to get a B 5000, 5500 in those days, for no other reason than that Burroughs is a Michigan corporation and you were the state university. You and Bob Bartels (then the director of the Computing Center) declined. Can you recall any of the thinking? Any of the reaction?

GALLER: Several times in the last 20 years, it's been strongly suggested to us that we should be getting a Burroughs computer because we're a Michigan university. On the other hand, as a public university we've always had to take the position that we cannot get a machine because it's a friendly company. It has to be price performance and everything else. We have never been in that position, according to our analyses. As I said before, the B 5000 was one which we were very interested in for all the same reasons that we've talked about. It was a very exciting machine, but we were in the operating system business, and we were told we couldn't get at the operating system. We were not users of computers, we were people who ran computers, and you wouldn't talk to us.

ROSIN: Do you think geography interfered with that? I get the impression from listening to Al and Henri that because they were in California and had a relationship with Brad and his people, they had a feeling about the system and an optimism about the system.

GALLER: Were you in California?

BERCE: No.

COLLINS: He was in Colorado.

BERCE: I was in Denver.

ROSIN: Oh, I'm sorry. Excuse me. I certainly never had the impression that Michigan had the close technical relationship with the people in Pasadena at all, that these other folks were hearing from.

HOOTMAN: I'd say your salesman let you down. That link wasn't established between you and the people as Dave said this morning. A lot of people took that thing and screwed around with it, and you were not handled properly.

MACKENZIE: Ironically, you didn't ask the right question, Bernie. You could have played around with the software.

FORD: You can imagine how hard and how far over backwards we bent to make a sale. If this was the only problem, it wasn't communicated clearly, because that would have been taken care of in 5 seconds.

GALLER: Somehow that link was not made because that really was a problem.

FORD: No, it wasn't.

BERCE: I'm surprised to hear you say that you were told that you had to no access to the operating system because we were told exactly the opposite from the very beginning. We indeed wouldn't want something where we couldn't have access to the operating system, so in some manner of speaking we were in a similar situation with the operating system business like you were.

PEARSON: For the first year after the official announcement of the B 5000, maybe up to a year and a half until Joe's training organization really got under steam in Detroit, the field sales organization was very, very thin of those people who had the ability to go out and convincingly tell other people what this piece of equipment was. Now, the few people who had the capability, then, would run across probably the single biggest point that would knock us in the head: that we did not have a FORTRAN compiler on that thing. The only thing that was available was higher-level language, and, that being ALGOL, and who knew what ALGOL was? Maybe Dijkstra and all those people can tell me how wonderful it is and it's the future of the world, but I've got $5 million dollars invested in FORTRAN programs here, and what am I going to do with them?

GALLER: I understand that.

PEARSON: So it was a very practical thing.

GALLER: At Michigan, that wasn't the problem because we had already done our own version of ALGOL 58 and we knew all about it. We would have been happy to have a machine with ALGOL. So something happened.

ROSIN: What about cost performance? I also have the impression, looking at the literature that came out of the task force (the Descriptor, for example), that a lot of the presentation of the system to customers was on the basis of technical innovation and technical excellence, and yet I hear from Al Collins that, in fact, cost performance was really the selling point in their installation.

COLLINS: No, not true. We bought the machine (well, we drug it out of Burroughs, I don't want to say we were sold the machine)...

HOOTMAN: I want it recognized that I was the account manager on that! [Laughter.]

COLLINS: You never took me to dinner! Anyway, we got to obtain the machine because we were very heavily involved in the ALGOL world, and it was an act of faith--the reason we got the machine. In fact, as with most universities you don't really look at "cost performance" except so that you can go brag at the other universities: "we're doing this, we're doing that." In fact, if you sat down and looked at the cost performance, the thing was an order of magnitude over the 7090. Because I was also responsible for running the 7090 shop. I sat there and I had a 7090, 16 tape drives on it, three 1401s to stage the jobs on and off, an army of operators. We finally wound up at the point at Stanford with no operators. With the 5000, we'd put the card reader in the hall and the printer at the other end of the hall, and any researcher walked by, dropped his deck in the card reader, and by the time he could walk to the printer, his job was finished.

ROSIN: That sure wasn't the case at Michigan.

COLLINS: On the 7090, we had an army of operators hanging and unhanging tapes around the clock. But cost-performance-wise, it was orders of magnitude. No question about it.

HOOTMAN: I took a senior consultant from one of the major accounting firms down there one time, and we watched the students just walking up and putting their decks in and walking over and getting them off. He looked at me, and he says, "That's the scariest thing I've ever seen in my life." Actually, the statistics that Al produced, though, came later in the game when things had stabilized. You know, those kinds of statistics we didn't have at that point.

COLLINS: But we knew those were coming, though, because of the statistics we had on the 220. We knew those were coming, and we knew basically the same people were doing the job a second time around. So, there was no question in our mind.

HOOTMAN: I think we need to comment about George Forsythe in that sense. George was a proponent of teaching people applications, and he felt very strongly that that university was not a FORTRAN trade school and fought bitterly. I saw him attacked one time at an ACM meeting by an IBM technical representative, publicly, for his decision on this thing. George just came back and he said, "Damn it, Stanford University is not a trade school for FORTRAN programmers." He had a very strong view of what the purpose of a machine was, whether it was Burroughs or somebody else, and he stuck to his guns.

COLLINS: There were some very interesting side effects of that. I don't know how many are familiar with them today. We were right next door to Hewlett-Packard, and a lot of guys went to school there and came over and bought machine time from us. The first commercial machine that Hewlett-Packard put out, its basic operating system was ALGOL, and if you want to know where the B 5000 is today, go up to an HP3000 and scratch the skin real carefully. The B 5000 is still alive and doing well, thanks. You can pick up the code and read it in the MCP, and it's almost identical.

BERCE: Cost performance. I have here an internal report from Marathon Oil, dated February 1962 (that's about 8-9 months before we got the machine). It says, "The lease costs are right (that's the B 5000) at $50,000 and capabilities, incidentally, are nearly compatible with the $60,000 IBM 7090. So, the IBM 7090, according to this report, supposedly offered a little bit more capability and in fact, cost 20 percent more.

COLLINS: And operationally, it was twice the machine.

BERCE: And operationally, it was terribly expensive in our way of looking at it because we would have to have additional personnel to run the damn thing. We at the whole comp center were eleven people in those days.

HOOTMAN: One assumes, of course, that there are rational aspects in the marketplace, and that isn't so. One of the problems of the B 5000/5500 is that if somebody didn't want it, there were 18 knock-offs that they could find anywhere along the line [to justify not buying it]. It isn't compatible with FORTRAN, etc. And the fact of the matter is that that marketplace out there was not then, and I don't believe is today, rational in the sense that people really do follow cost-performance justification. I remember one time going back on a deal that we lost. The man gave me the comparative analysis, and it was absolutely a piece of trash. This man was a Harvard MBA and a very bright and confident guy, and I just raised hell with him. He turned around, and he gave me a great quote which I've never forgotten. He said, "Mr. Hootman, never forget that irrational decisions must always be justified on a rational basis." So much for cost performance, even if we could prove it. You could use it if you wanted to sell the case, but it didn't always work. The thing that I wanted to bring up, that Jim and I had talked about, and that is an absolute tragedy, was that Ray MacDonald, in his infinite wisdom, running the International Division of Burroughs, refused to release the 5000 to the international market and particularly to the Europeans.

TURNER: That's absolutely not true.

HOOTMAN: False?

TURNER: False. And I asked him two weeks ago, and he said that the reason that he didn't introduce it is that Ray Eppert wouldn't let him introduce it. He would not allow him to do that.

GALLER: Do you want to comment on that, Joe?

HOOTMAN: What can I say? The point of whether MacDonald did or didn't, I think, is that it wasn't released into what was primarily an ALGOL market.

MACKENZIE: At that time.

KING: Did he say why Eppert wouldn't allow him to release it? What was Eppert's reasoning?

TURNER: Apparently, Eppert's reasoning was that he would have to train a sales force and have to get new service people, and that they wanted him to make hay in the U.S. first and release it in Europe maybe later. That's the essence of the thing. It didn't make any real sense. After that he said he got to be executive vice-president and started running the place, as soon as he saw the sales cost, the price-to-cost ratio, that it was so good that he insisted immediately that it be given to international.   And that did happen in fact.

MACKENZIE: Subsequently, I remember one time counting at least 25 systems that were operating internationally. But they were, as Joe pointed out, later in the program.

HOOTMAN: Except for the Japanese.

COLLINS: A few of them went down to Australia and New Zealand.

GALLER: It's interesting, though, that Europe in particular, which accepted ALGOL much faster than the United States did, would have been a prime area for marketing.

BERCE: There, again, that points out the lack of understanding of the machine on all sides, but some isolated users, including Burroughs top management. They didn't understand what the hell they had.

FORD: Why did Mr. MacDonald want to kill the project if he thought it was such a good machine, and he did? A group was put together (financial men; I happen to know one extremely well), who participated...

GALLER: Was this after the period we just heard about, when he realized that it was a good idea?

FORD: It was 1966 or 1967. It was after Eppert. It was when MacDonald took over, whenever that was. I don't try to second-guess any decisions, because you don't know the pressures that they were under at the time, but certainly, the company was looking to be as profitable as possible as soon as possible. The B 5000 at that point in time was not a winner. It was natural to consider whether or not it should be continued. I am told by my friend who was the financial representative at the meeting that the reason it was continued instead of aborted was because at that point in time, it would have cost more to discontinue it than to continue it--in other words, to shut down the ElectroData plant. That was the basis on which it was continued. Fortunately, by the time the systems were run out and so on, the sales began to pick up, and it became a profit-maker.

ROSIN: Can I interrupt you? In the question of it costing more to discontinue it, was there a viable alternative?

FORD: Yes, I guess that's because there would have been so much money spent to balance parts and all of that. Whereas, they could keep the thing going and see if they couldn't make it happen.

BERCE: My guess also is that there was another consideration there on the part of MacDonald's decision. That is, there was nothing really to plug the void, the vacuum, that discontinuance of the 5000 would cause.

GALLER: That's what I wondered. One would almost wonder why they even considered stopping it at that point if there was not replacement machine.

HALE: There was a new machine under development at that time.

ROSIN: That was never released?

TURNER: Oh yes, the 6500.

TAPE 4/SIDE 2

HALE: No, the 3500.

TURNER: The 6500 had already started by then, too.

KREUDER: Yes, we were working on the 6500 by that time.

ROSIN: Maybe I... This might be a reasonable point in our discussion where we could ask about some of these successors. Was the 3500 at all related to the 5000/5500? Was the 6500 at all related?

DAHM: The 3500 was related in one philosophical sense. It was a machine that was designed with the idea in mind that it would be a COBOL execution machine.

COLLINS: But other than that, it was a standard von Neumann register machine.

BARTON: There may have been another reason, I think I can offer as an observer. By the time the 3500 was out, the disk MCP was known to be successful and they used a smaller disk, with the 3500, I think largely on the strength of that success, basing some of their systems on that disk. They were not tape dependent in the way competitive IBM systems were at that time. Would you guys that worked with that thing agree with that?

HALE: That's right. It was disk-oriented from the beginning, the 3500.

BARTON: Do you think that was helped by the 5500 success?

?? It was the same disk.

MACKENZIE: I thought it was a smaller one that was used for that.

KING: It was the same disk.

MACKENZIE: Ford got to choose the segment size.

PEARSON: Before we leave Ray MacDonald, I can remember at one time him calling meetings of several of the marketing organizations together to kind of pre-release or pre-announce his plan for the future. Because of the financial condition of the organization, for a period of time, they were going to have to return to marketing those products that they understood and knew better, which was more or less the standard line of products that Burroughs had been known for years. And they were going to have to deemphasize the marketing of the computer systems. At that time there were a number of us who were dedicated to computers who thought, "My God, it's been deemphasized so much that with any more deemphasis, it's going to disappear off the face of the earth." But I know that was one of his commentaries, and it probably was a very wise move for the continued success of the Burroughs Corporation because they did do that and their revenues did pick up, and they were then able to continue on and I think then began to put money back into the computer program as well.

?? Is that your feeling too, Jim? Do you recall that period of time?

FORD: Yes. I would say it's accurate.

GALLER: But that deemphasis is not the same thing you're talking about possibly...?

FORD: Well, deemphasis takes many forms. One form was to restrict the growth of the sales organization--actually cut it back, which was what you were talking about. I would restrict the areas into which salesman could go in order to cut down on expenses. So there was a general contraction during a period. As Ralph has said, it undoubtedly was necessary. It was painful for many, but it enabled the company to survive and then to resume its activity. So that was a phase which the company went through which was difficult but probably necessary.

HOOTMAN: There was also a major program undertaken and Jim has alluded several times to the financial problem. We had a program to convert the existing leases over to purchase, and it generated something like $30 million in cash flow, which was a critical thing to do at the time. MacDonald saw me years later and looked at me. He said, "Oh, I know you, Hootman. You're the guy that helped chew up my lease base." Guilty as charged, but that was a tough one.

FORD: It was hard to win an argument with MacDonald. [Laughter.]

HOOTMAN: Well, I did get him one time. Ralph mentioned this briefing. In a meeting in Los Angeles, he said that he had withheld (he didn't say at the time, that Eppert told him to) the 5000 from the international market. We had a Japanese distributor, I can't remember... Kohiki something or other.

?? Katakiu.

HOOTMAN: Yes. And they had sold, at that point in time, three of them. I was sitting next to Ray on a tabletop while he briefed the marketing group, and he said that he had refused to release the thing internationally. And I sotto voice said, "Ahh, soo." [Laughter.] As we walked out of the room (he's a fairly tall fellow), he put his arm around me like this and he said, "Hootman, I heard that." [Laughter.] That's the only time I ever got him.

TURNER: I was in Detroit in product management--in fact, in Group Three Product Management working for McCollough, from late 1965 until 1967. In 1965 and 1966, the machine's sales had tailed off considerably. In 1967 it started to pick up again. But in no sense, while I was in Product Management, did I get the impression that the corporation, at least in those years, was backing off from the thing. In fact, before I left Pasadena when I was still working for Kreuder, we were already starting the plans for the 6500. So in 1966 and 1967, we were trying to get the funds to do the 6500, but MacDonald was very much aware of that going on and, in fact, let it get started. During that time, since I was in the Product Management organization in Detroit... Certainly McCollough was for it. Maybe there was something that I didn't know about, but we were pretty close to those guys during the time.

BARTON: Of course, while Lloyd was there, did a certain amount of damage. He talked me into getting involved with Burroughs again.

MACKENZIE: More sabotage, huh?

BARTON: Yes. More sabotage. I remember walking in the door the first time, and I was met by one of the people--his name was Fred Kalback and he said the people working on the 6500 want to see you. I remember clearly what I said: "I don't think that's a very good idea because I've already heard that that machine is pretty much frozen, and if I go down there, you know what might happen." He said, "Oh, no. Don't worry about that. They just want to say hello." Some of the people here remember who were there at the time.

GALLER: Joe, you mentioned the fire department earlier. Would you care to explain?

HOOTMAN: Well, Jim, I guess you were responsible for doing this. There were two groups in 2nd Avenue in

Detroit, one consisting of technical people and another consisting of some marketing people, who were assigned to try to do what could be done to make sure that people accepted machines when the things were shipped and did not stop paying lease and did not throw them out. I did that for a while; [Len] Kreuter did it for a while. Then there was a technical team of a couple of guys; Sleeper was in there, Hebner was in there under Herb Hayde. The fire department was our own phrase for it. I think we had some very officious sounding title. But, basically, our charter was to help the field get people to pay on time and promptly and continue to keep the machines installed. We were the complaint department. And when something happened, you know, when the stuff hit the fan, it was Jim or Gordon, or phone calls would come in to Eppert or whoever. They ended up in that bailiwick. It was a burn-out job, as a matter of fact, because it was in the office on Monday morning, collect the phone calls, find out what's going on, get on the airplane, and get out there and see what you can do. The interface back to the technical groups in Pasadena was strong, and particularly in this case, through Hayde and the other technical people, as well as the field engineering people. We had some not insignificant problems with certain installations on field engineering support. Sometimes it was a function of the people at the site or whatever; typical kinds of management problems. I shouldn't, I guess, for the archives name the client but it was an eastern B 5000 installation that made carpets and...

HALE: Mohasco.

HOOTMAN: I wasn't going to say it! We had a hell of a time maintaining the machine for some reason. I don't know what all the problems were, but...

?? Cable connectors. [Laughter.]

?? We had problems with crashed heads and things of that nature. So there was a concerted effort by a small group of people to stay on top of those problems and try and make sure that the customers were satisfied. It was damned hard work, I remember that.

BARTON: Bernie, may I ask a question of the marketing people? I have an impression sometime in this period after the machine was out and going that one of the reasons why it had become profitable was that it was demonstrated, that it was, in fact, expansible; people were adding a lot onto it. Now, I don't know whether that's actually true; I'm just telling you what I heard in those days. If it's true, it's worth mentioning, because it tells you something else about the MCP and the role of that. I wonder if the marketing folk can throw any light on that. Was there a period after which 5500 users started expanding small systems to the benefit of the whole profit picture? Is there any knowledge about that?

FORD: I left in 1965, so I really couldn't comment. Specifically, it seems to me that it's historical as far as computer systems are concerned that they do nothing but grow. Certainly it's the secret of IBM's success, and I have no doubt that this occurred, but...

BARTON: In those days it often was thought to require a lot of awkward reprogramming and so on.

KREUDER: I can shed some light on that, I think. For a while in Pasadena, our shipments out of Manufacturing were on the order of 40-50 percent add-ons and 40-50 percent new products for the 5500 product family. The nice thing is that the gross margin on the add-ons was immense. An initial system usually got its manufacturing costs back in the space of perhaps 10-12 months, whereas, for instance, in an add-on CPU you've got the total manufacturing cost back in less than 2 months. Memories were not quite that good, but they were still awfully good. So the margin just gets a whole lot better when you begin doing the add-on, and of course the sales cost must be a lot smaller too, huh?

HOOTMAN: Yes.

KREUDER: Because the customer sort of phones it in, right?

FORD: Well, Al says that they did that anyhow, so I mean...

BARTON: So from the factory end, the rumor apparently was true. Presumably people then had really established that they could make the machine grow nicely. And also some shrank the machine, too, I remember. That worked as well in one case. Was it United Technology, Joe, that had to cut back in their budget, and sent part of the machine back, and still ran the same program?

HOOTMAN: Yes.

GALLER: I would guess, from the reaction of the users we hear and the ones I know from other places, that they've pretty well stayed with these machines and grown with them. Is it correct that you haven't really lost many customers over the years to other systems, or have you had the normal turnover that everybody has once in a while?

MACKENZIE: There's not really anybody here that can answer that I don't think.

BERCE: Of the 5000 period, or across the board?

GALLER: I was thinking until now, even.

COLLINS: Well, just sort of gut feel, of the 5000s that I knew of, most of them lived to an extremely ripe old age. There was a couple of them down at customs in San Diego. It was not unusual to see a 5000 in a shop for 10, 12, 14 years, and people still running production on the thing.

BARTON: Oh, by the way, the used market price for the machines stayed abnormally high during that period.

ROSIN: Oh, like BMWs.

COLLINS: Yes, yes. Everybody kept them at least 10 years. I can't think of a single one that went out in less than 10 years.

HOOTMAN: I think from what I've seen in some of our consulting work, there are some major Burroughs installations that have been lost over time. I suspect by the same token there are some new Burroughs installations cropping up that were not users. Probably some of each.

MACKENZIE: There tends to be a very solid user base from the point of going from one system to the other, but if you go back and look at the original 5000, the first 40 systems or something, I'd hazard a guess that maybe eight or ten of those customers still have Burroughs large systems. So there has been attrition, but they've been replaced by and large with new customers that have maybe bought two or three systems and moved on.

HOOTMAN: As an aside or something, I'd like to add something into the record, and I think it was fun. In the late 1960s Dave Dahm and I put a company together called Remote Computing, which was one of the early time-sharing companies. We got, because of Dave's expertise and my finagling, I guess, what I think was the first major software contract that Burroughs ever let outside. Dave and a team of people developed the time shared operating system for the 5500. Some of the things we talked about today just bore an immense amount of fruit in that thing. To watch Dave and his people compiling the time shared operating system, while we had paying customers on the machine doing the same thing, was absolutely one of the most mind-boggling things I've seen in a long time. Those guys would come in and just log on. I had to ask them, I guess, eventually, would you please (come in and work at night.) It wasn't that you were screwing up the rest of the customers, but you were using up too many cycles that we could have billed out in the daytime. I think it was a remarkable endorsement of the technical product of ESPOL [Executive System Problem Oriented Language; the compiler compiler] and the hardware capability that in a time sharing environment, those guys were maintaining and modifying and recompiling that system along with other jobs in the mix. Fabulous.

KING: Were you involved with the use of the Chinese fire drill?

HOOTMAN: Oh, God. Yes. Yes, I'm guilty. In the fishbowl?

ROSIN: You'd better explain that one.

HOOTMAN: Somebody touched upon the question, how did you guys, in those early days, demonstrate the capabilities of the machine? You've heard reference to the fishbowl; the 101 was behind the curtain. There was a big window, and there was a set of drapes in front of it, and we would actually put prospective customers on an airplane and take them to Pasadena. We had a little group (Ralph was down there, and Paul Leebrick for a while), and we would arrange to troop these people in and open the drapes, and, God, there in all its glory was the B 5000. People wanted to see the damn thing do something. So, we developed a demonstrating routine that exercised various parts of the system. It's pretty damn hard to demonstrate multiprocessing when you have a console that doesn't have a bunch of lights on it. So you just make the tapes go and the card reader go and the printer go, and the whole damn thing would just go like hell and the users would go, "Hmm, but what's it doing?" But we had the Chinese fire drill, and that's what we used for a long time as a demonstration. Now the sequel to that story is what happened in Cobo Hall in Detroit. Jim made a fabulously gutty decision, I think. What the hell was it, an ACM conference?

KING: Spring Joint Conference in 1963.

HOOTMAN: Here's Cobo Hall, a huge, huge thing and a little postage stamp space in the middle of it with a raised floor and cables running off out to the edge, and the intent was to install a 5000, I don't know which serial number, was it 103?

MACKENZIE: No, no, no. It was an early one, obviously. It was in the spring of 1963.

HOOTMAN: The idea was to install the thing and demonstrate it at the conference. I think it was an incredibly gutty decision, but it was responding to that same need-can you show people that this thing does, in fact, exist?

GALLER: Did it work?

FORD: It was exciting.

HOOTMAN: It was exciting.

?? He asked did it work?

KREUDER: There was a little problem technically with the machine. As I recall, there was an electrical problem getting the installation in.

HOOTMAN: Well, we had some problems getting a proper version of the system in and getting it up, but we did run the Chinese fire drill. Of course, that Paul is alluding to is that Ralph Butt, with whom Mark, I guess, had talked on the phone and couldn't be here, was up on the podium with a microphone and was in the middle of explaining the interrupt process and how the peripherals could interrupt, and so on and so forth, and the printer was out of paper and just going merrily along and the damn thing, of course, didn't interrupt at all. This guy from Prudential Insurance, who was a prospect, was standing and trying to get Ralph's attention and telling him to knock it off and go on. So, we did have some problems running the Chinese fire drill, but...

KING: What I was referring to was the stories I heard of demonstrations that were run on the system that was in Detroit, where you'd bring people in and run the Chinese fire drill and then shut off part of the system and run it again, and then shut off another part of the system and run it again.

KREUDER: It went the other way, I think, Paul. We started out with a minimum system and then you could add on other I/O channels just...

BERCE: You had to go the other way because you couldn't shut off the memory bank. You could add on to the memory bank...

KREUDER: The system would pick it up and, you know, understand it had another I/O channel and...

KING: People wouldn't believe what they were seeing. I heard the stories that they would come back the next day. The first day, they didn't understand what they were seeing, the next day they'd come back and see it and they wouldn't believe it. And then the third day they would come, they'd see it, and they'd believe it.

HOOTMAN: I think there was a lot of skepticism in those days that we were faking it--that the whole thing was a gimmick and it was a hoked-up deal. The fact of the matter was we were running a standard version of the operating system; we didn't have a hoked-up version. The damn thing didn't always work for us and we did have problems like the printer and so on, but we were running the real McCoy. The other Ford story I want to tell, of course, has to do with the button, Jim. That was the Ralph Butt invention and what did he do, Pearson? He solicited from each of us a contribution, didn't he? No, I guess he went to you, Ford, and Jim said he didn't think this was going to... I happen to have one on my shirt. It says, "I touched a B 5000," and that was from the computer conference in Detroit. Jim didn't want to do that, I guess. I don't know what your reasons were, but anyhow, Ralph decided that Jim didn't know what the hell he was talking about, so he collected from each of us and went out and bought the buttons.

GALLER: Ralph Butt?

HOOTMAN: Yes. Obviously, the things were an enormous hit, and Jim then grinned and said, "Well, all right. Buy some more, let's get on with it."

GALLER: I remember that button from that conference.

PEARSON: If any of you have never touched a B 5000, you're welcome to come over and touch part of the B 5000. [Laughter.]

GALLER: Well, gentlemen, it's I think time to begin to bring this to a close. Would anyone like to make a closing statement?

PEARSON: I think I'd like to say that from the viewpoint of a tactical field salesperson, I can't stress too much nor thank these guys too much who worked in Pasadena during that first year to year and a half, when that's where much of the marketing and the sales work was done, not only by everyone who's in this room, but by some people who weren't in this room that worked for Edgar Eichhorn in the scientific applications area. Because without some of those applications programs, some of this equipment wouldn't have been as attractive as it was at the time. But to a great degree, this was our sales organization and the unstinting effort that everyone of them gave, that guy over there and everyone who's in this place, to see that that thing got sold, and once sold got installed and kept running-unbelievable. You can't say thank you enough.

COLLINS: I'd like to make a statement and this is the deepest, best-kept secret in history. I was out with the purported father of the B 5000 one night. He became extremely intoxicated and he told me where he got all of his ideas. Ray Marshall drew the schematics on the back of a taco. [Laughter.]

BERCE: I'd like to comment on a number of mentions that were made this morning in a debate as to how good the people were (or really how no good the people were) in designing and engineering the machine. My impression was that what they didn't know, they explained to each other.

GALLER: I think we lived through a period back then--I working at the university, you people doing your work--which not many people will be privileged to live through again in our field. I think it was painful, but it was exhilarating, and we were lucky. I really appreciate your coming together today to do this. I know I had a great time, and several people here have commented to me that you've enjoyed this and I understand that.

HOOTMAN: I just wanted to say, Bernie, on behalf of myself and I'm sure the rest of us here, we appreciate the time and effort that you all have taken--and Mark. It's been a very fun and worthwhile experience, and thank you to your sponsor as well. They may not feel very comfortable reading some of the things that have been said, but nevertheless, thank you for the opportunity to be together.

GALLER: Thank you.

CREECH: Along those lines, I'd like to say that in the spirit of toast, I don't have anything to drink. It has to do with the fact that as far as credit, you are the one that started all this a couple of years ago with your letter and caused this eventually to come to be. That was the start of it. I happen to know that for a fact. I think it's safe to say that you're one of the few if not the only person who commands the respect enough of all the people in attendance here to pull this off. I think we owe you a debt of gratitude for that.

ROSIN: Hear, hear!

GALLER: Thank you. It was fun.

END OF INTERVIEW