GB: Probably you want [to get on tape] Larry Portner, Bob Puffer. Andy Knowles was important, particularly in the way we worked, there was enormous stress between marketing and engineering. The more you can try to center it into the different activities, the better off it is. Because it's a complex story, and you want to try to keep it that way and then talk about the interactions between the various forces.

There is a way of doing it a little bit chronologically, too. When Glenn Rifkin approached me about his book he had an outline. He wanted to come interview. I said, "No, don't do that. I'd like to think a lot more about this. Why don't you give me every question you want and I will answer the questions and think about them in that context." I did that. He had an outline and I gave him comments on it, how I would organize it. There's a lot of stuff in here, essentially a lot of stories. As we go through the questions I'll try to remember it. He came back after that several times. [BELL HANDS OVER 20+ PAGES OF NOTES HE WROTE FOR RIFKIN'S BOOK.]

I talked to Henry Burckhardt the other day, and I was out talking to Dave Cutler. Henry said, "Well, it turns out it takes eight years to get it out of your system. You're just able to be calm about it now." So it's been roughly eight years, and eight days now since I left. The other thing is the less I become a DEC stockholder the better I feel. If I had purged myself of DEC stock a long time ago I would have been much richer, but secondly, I might have just totally ignored the company. Maybe when I finally don't own any DEC stock I may be able to feel good about the company.

INT: I think maybe one different perspective from the Rifkin book is instead of a political history we're trying to recreate a little bit of climate.

GB: I think that's right, absolutely. Everybody I think that you would ask, "Would they do it again?" Everybody would say absolutely. No question about it. I was involved in another company Ardent, which became Stardent. Crazy company, but the greatest group of people I'd ever worked with, outside of the VAX team. But we accomplished an enormous amount and we feel very proud of it. Some idiosyncratic, crazy people running the goddamn thing, that you would rather not have to be there. That's the same way I feel about DEC sometimes. Virtually everybody feels enormously proud of what they accomplished in that environment. With that as an overview, plus the understanding that Ken really was able to manage that environment for a certain time -- he was really a great industrialist. You'll see that in this [GESTURING TO PAPER GIVEN TO INTERVIEWERS] an enormous amount of conflict we had about working. Maybe one needs that stress and conflict. You probably do need stress and conflict, and that was one of the things that determined the environment. Lots of it was unhealthy.

[STILL GOING THROUGH PAPERS TO GIVE TO INTERVIEWERS] There's this article in Computerworld that I wrote while I was at NSF in 1987. This is my current vita, which is how I regard all the things I've done, so it's long, which is everything including books and articles.

INT: We want to focus on engineering.

GB: Great.

INT: How would you describe DEC's historical approach to engineering?

GB: I hope you'll search the archives first. For example, we wrote many handbooks on doing engineering. Let me urge you to use them before you go to the oral histories.

I think the engineering approach varied over a long period of time. I kind of view the periods as almost associated with these major organizational changes. There was the modules, getting the company started, which

I really don't know much about. There was the period of when Ben [Gurley] came, built a prototype PDP-1. I came a few months after that, and started working on the -1, and worked that whole structure, and the -4, -5 and -6. That was until '66. I left then, but the main change was the organizational change which was critical.

The products and the organizational structure are highly correlated. That's why I think for your own sanity that you may want to break it into these periods. Because where there was great organizational turmoil, there was a change in the way things were done. I would break it, Pre-computers, Computers from '66 when there was an organizational change into a product line structure until '74, and then the product line structure was getting in the way. Then we went back to kind of a functional organization, but with the market product lines. That in a sense got obsoleted by VAX in the early '80s, when we had all these people painting VAXes different colors, and pricing them differently and they weren't focusing on the market. At that period the marketing groups got destroyed, and not replaced by the right form of marketing group. I view the period from '82 or so, called the VAX strategy, which was all momentum. Denny the Dunce could have run the company. It didn't matter. It was standard, turn-the-crank, evolutionary engineering, nothing very creative, just slug it out. Then the PC's coming in, not being able to be integrated at all. Then this period of chaos that the company's been in for the last, two or three years; recognizing there are these things called standards, and UNIX and open systems and oh my God, what are we going to do with our life? We need a new strategy. There was a period of searching going on. There's an interesting paper on the growth of companies, and it talks about periods of growth, periods of unsettled change and floundering around, and then another period of growth -- you can go up or down during these periods, of course. It's companies trying to find out how to deal with this new environment. Given all of that, that's how I divide the world. Clearly the future is in question.

Let's go back to the question. The computers during the early '60s when we created the -1, -4, -5 and -6 architectures, that was a model of "We'll get market by creating all of these products." It was quite an entrepreneurial environment. We had "Computer Special Systems," developing products like memory testers. I remember doing a one-page pricing sheet once for new projects, where you'd say how much is it going to cost, and then you'd kind of multiply that by 3.3 and you'd look at how many you thought you were going to sell, whether it was one or not, and put the engineering costs and price the option. That way of coupling engineers to markets and products was great. It was coupling projects to a real customer demand. Like the -4 was built to sell to Foxboro and Corning and a bunch of other people in process control. The -5 came out of a special controller for Chalk River. This is described in the book Computer Engineering. The -6 was really our first attempt to build big computers. None of us knew a damn thing about software. We could write programs, we could write compilers, etc. None of us had the foggiest idea of the issue of software, the cost of software, and the idea that there was a kind of a balance sheet associated with it, that this code was worth something, you want to accumulate it. There's a lot in this new book I and McNamara wrote. We discuss my confession -- 'Oh my God, why did I do a PDP-4 rather than just making some changes to the PDP-1, reimplementing it, because it was kind of an engineering thing.' I come down so hard on engineers from time to time about incremental improvements; the only reason I do it is because I know precisely what goes through their head. I've been there! They say, "Oh yeah, boy, if we do it this way we'll save six diodes, three flip-flops, and oh, we'll save $1,000 on every one we build. And oh, we'll get to rewrite the software, and it will only cost $3 million dollars!" "How many are you going to make?" "Well, ten. Or a hundred."

Engineering was initially very entrepreneurial, then got into the product line structure in 1966, which was again quite entrepreneurial. This is described in my notes to Rifkin.

JP: The PDP-6 was a real departure from the kinds of things that we were doing. Would you consider that the first real engineering risk the company took? Because that was a big investment for the company's size in relative terms.

GB: Absolutely. That was a big goddamn risk, in the sense that the PDP-1 was a copy of the TX-0, taking the modules and making that work. The -4 was crazy. The -5 was a real contribution; it was the first mini to be used as a component. [With] the PDP-6 we said, "We've doing these little timesharing systems (the PDP-1 specials) and let's make a real computer now." I don't remember why and how the PDP-6 got started exactly.

JP: But somehow the powers that be let it happen, right?

GB: Yeah, I know I proposed it, and set off and started thinking about it. Then got Allan Kotok got involved and then it started going. I think it was at around the time that the MIT CTSS was coming in, and we said, "Let's make a time-sharing computer," because we had made one around the PDP-1 that was too small, it couldn't do the kinds of things that we wanted to do. You needed a real calculator. The PDP-1 didn't have floating point. If you're going to share a system it needed to be able to execute Fortran at a reasonable rate. So the smaller linked machines just couldn't run fast enough. So we proposed it and got to do it.

Prior to '66, extremely entrepreneurial. We had a good special systems group doing a lot of stuff. Then this idea of using a computer to do other things became clear. I remember the PDP-5 came out of that. Ed deCastro, its product engineer, was an applications engineer assigned to build the Chalk River Special Front End. Using that to build special logic you'd use the computer and program it. That came out of a lot of our backgrounds, it was something that I learned at MIT when I was doing speech research. Special purpose systems aren't worth building, you just program a computer.

INT: If the -6 was a change was it driven by any customer need or just engineering?

GB: As I recall, we didn't have any customer that wanted it.

INT: Australia comes to mind.

GB: No, they were the first buyer! That's how bad it was! We couldn't find anybody who would buy it! Initially in the business plan there was an 18-bit computer and a 36-bit computer. In '62-'63 there was a time when we had proposed this PDP-3 to Cambridge Research, and lo and behold they ordered the goddamn thing. We said, "Holy shit. We can't build it!" Harlan Anderson and I went over to them, and we were driving down Route 2, AFCRL, and I said, "Andy, what we're going to do is we're going to sell them a 36-bit machine. It's called two PDP-1's. 18 plus 18, it's 36. No problem, we're going to change the contract and we're going to solve it." We had gotten the ITT order which ultimately resulted in 20 PDP-1's. I was the project engineer of that, and we were just sort of squirming to get that out. Here we get this whopping big order for a machine delivery in nine months. Of course the PDP-3 was just a souped up PDP-1. I didn't think it was particularly good. So we sat down with the guy with our new purchase orders and said, "Sorry, we made a slight change. We know we proposed this a couple of years ago when you set about getting the contract, and ordered it, but we're going to make a slight substitution." [LAUGHTER]

MAN: What was your role in going into the PDP-6?

GB: It was my project. I was the architect, chief

implementer and project engineer. My notebooks are around somewhere. I don't know exactly where they are. "Procedures for engineers using a notebook." This is 1963. Here, 6-16-65, spreadsheet in terms of time, original specs, time, present time...
The documentation for the birth of PDP-6. It looks like that's what it is.

We always tried to make DEC engineering highly entrepreneurial. There are a couple of slogans that I'm very proud of, one is "He who plans, does." This was coined by me in 1972 when I came back from CMU. Then I had a one-page memo on the make-buy, what you should be making versus buying.. a policy. Trying to make engineers be responsible for what they did. Totally responsible. Engineering in my view always had a lot more responsibility than it had control or ability to execute, so you had to do this otherwise you'd have all this tremendous fingerpointing.

But during that time there was a strong product-centered kind of responsibility, hierarchical thing, somebody being responsible for the new set of modules. Then when we did the -6, there were the circuits guys. Dick [Best] managed the circuits guys. I oversaw <u>all</u> the computers. I directly managed the PDP-6 project, did the architecture, the logic design of the processor, everything except the floating point. Alan [Kotok] did the floating point. Dave Brown did the memory controller. I'd say "Dave, how's it coming?" "It's not done yet." So I ended up doing the memory controller, and then there was somebody else doing the tape controller. I'd say, "How's it coming?" "Not very well." I ended up doing the tape controller! I designed virtually all the logic of the machine except the floating point unit.

Then we started putting it into production. I think Alan was responsible. Then also Bob Savell, I believe. I went down and started working on software. That's when Dit Morse, was trying to run the operating system. The notion of a operation system, per se, and then separate compilers and utilities had a structure and interface to it. We were trying to get this operating system up, and every night Dit would come up and change the calls. The next day I'd hear: "My compiler, I can't use my compiler, what the hell is this?" This happened over a period of time. Dit wasn't able to to manage his time. I worked on the data structures and lay out and what the calls were going to be and all that, and Dit was implementing it. I finally said, "Dit, I'm going to run this project, get out." He's the first, and only guy I ever fired. The irony was that Dit was a really smart guy. I've spoken to him since then. Then when I came back in 1972 and headed engineering, Larry Portner, who was running part of the software on the -6, came to me and said, "I'm going to hire Dit to do a file system." I said, "Hey, that's great. He's certainly a bright guy and he clearly can do that." Six months later, Larry comes in and says, "I fired him." And I said, "Well, maybe you learned something, that was yours, it isn't mine." But Dit had done a reasonable job. He had built the architecture for the file system, the PDP-11 file system, which is probably the base file system that we still have for everything. But it was funny.

MAN: So you had trial by fire in teaching yourself software?

GB: I had written a compiler when I was a Fulbright scholar in Australia. I'm not a software guy, but I occasionally program.

[END OF SIDE A -- BEGIN SIDE B]

GB: I'm not a classical engineering manager. I don't know whether I would work for me or not.

JP: That's an interesting point. Besides brains and ability were there other characteristics that you looked for when you were building your team?

GB: I guess in different times, while you're doing a project team you're weighing project kinds of people and you look for somebody that is good at that. Every month or so I'm involved in starting a project up somewhere, basically it's the same old stuff. Do these people really have a very strong technical base? Do they have a process? My view of how you engineer is pretty much embedded in the new book. The thing I think I have is the ability to take a lot of complexity and to structure it into smaller problems; at least that's how I see myself. Things have to fit together. I think I'm an architect. That's an intuitive thing. There are times I'm an engineer, if you have to look at waveforms, or go down deep and do something, I can do it. There are pieces I don't feel comfortable about doing in software now. But I mostly understand things from the electron level to the integrated spreadsheet stuff. So I'm interested in all computing, the chain and how you build it. Structuring that complexity. All the books I've written have had that flavor.

MAN: You evolved that role because you stepped in and you just did it.

GB: Right. When I went to Carnegie in 1966, there were a bunch of people in marketing leaving because it was an organizational change to the product line structure. That wasn't the issue, [though], nor was it the issue when I left in '83. I thought the company was in great shape [then]. [I left because] I had fulfilled the contracts that I had made with myself. In '83 it was clear, after my heart attack, that I wasn't able to deal with the stress. I think I just saw it wasn't working with Ken and Jack. I wanted total absolute control over engineering: that was it. There wasn't any negotiation. There wasn't any room to move. It was very simple. I pretty much knew what was going to happen when I left. The company would run fine for awhile and then Ken would screw it up -- perhaps beyond repair!

As an engineer there's only one way you can know something. You have to construct an experiment to prove it. Leaving, as far as I'm concerned, was the beginning of an experiment. It took a little while to execute. I wanted to be wrong. It was an experiment I wanted to fail, but in my gut I knew what was going to happen. Intuitively I knew the characters involved.

MAN: As the product lines developed, how much did you work with the cross-functional groups?

GB: Digital was an engineering environment, strongly entrepreneurial in the 1960's. People driving to build projects. Memory testers, special systems. Ed DeCastro was in the special systems group initially. Somebody would say, "I want a controller to do that," and they'd go off and build these various projects. We were all together up in Ottawa for the [PDP_-5] controller. On our way back we said, "That's going to be a computer. You go off and build that." That's when I went off and did the -6. Kotok and I did an architecture for PDP-5, and said, "Ed, go build it like that. Here's what we've learned about the -4 that goes into the -5. The historical approach was that.

The engineering committee, over a long period of time played an important role, as a bunch of different things. Initially Ken communicated with engineers. We all communicated about project status. Project status was dealt with [by the engineering committee.] It was a place for a consensus about what something was going to be, or how to solve a large problem. It might be interesting to get the engineering committee's minutes.

I think if you look, Ed DeCastro I would guess wrote one of the first engineering standards. We started embedding engineering standards. The combined knowledge of how you do engineering (the process), and what the product constraints are. Half the companies don't do that. It's really important to have those down in one place where you can look at it and somebody can say, here are the standards that we've agreed to: e.g., environmental standards. Things that could work with other things, because as you built bigger systems you had to have everything running in the same environment. Things like that were very important. The engineering committee was the place where engineers from every group met together. [You should ask] Allen Kent what his historical approach to engineering is; Allen's a key guy. Tom Hastings, too. Kent and Hastings were both very good scribes, and are keeping order in all of this. In my view, a lot about me versus a lot of this is knowing precisely what everybody can do, and how they operate and what their good or bad parts are. I characterize people in many dimensions. I look at them under a microscope. For example, when we did VAX, I hired Hastings to do programming; he maintained the VAX architectural strategy. Strecker is very good at that, very clean, very precise writing. Kent was like that in the physical area. I look for people like that at various times. You need them on every project, whether it's an editor or what. Know what you've got. I'd say it is an intuitive feeling of knowing what you need to have to make a project succeed. My book shows what I mean.

[GOING THROUGH BOOK] This is something I call a technology balance sheet, and this is what I look for in a company. When I go into an engineering organization I look at 12 dimensions. This is a refined view over time. This is a part of a whole theory in the book because I look at a company and I say that in a start-up there are 12 dimensions, and I look in engineering and there are 12 dimensions that are important. You've got a bunch of people here, the people dimension. I ask who's running the project, how do they work as a team and all of that. Then, whose vision is the product? Does it reside in one head, or can you show me the structure for the integrity of this project? Then I look for other things like, here's your process. What standards are you adhering to? What are your internal standards? What is your technology -- what set of skills do you have that you uniquely can execute that nobody would allow you to do things? So this is being very explicit about what I feel, how I think I've operated intuitively. I have this intuitively, and I just forced myself to put it down [on paper]. This book, in a sense, is a highly structured [view] of how you start a company. Here are 12 dimensions, this is how these dimensions should grow over time. Here are the states that the company should go through, very machinistic. I believe you can start a company just like you can write a new piece of software, with the same reliability. You do all the things by the numbers. That's absolutely totally non-intuitive. I spend my life being either one way, or being in both of those camps.

MAN: Were these criteria applied to the 11 and the VAX projects?....

GB: I think a lot of them were there. I think all of these had the character that everytime you engineer something, you're always stretching to do something that's better than the previous one. So in a sense the project is always going to be in some kind of trouble because you're always stretching. When I consult I'm very careful not to interact at a certain point, because I know I can make the project undoable. [LAUGHTER] There's some stuff going on at Intel now. I said, "OK, go run the test chips. I will not talk to you about the project until later...Let's see what the tests show." I risk not getting the tests. We call this "ntl"-ing a project.

There were strong product lines, and there was a lot of pressure on product lines. The -8 product line was there. The 18-bit product line and the 36-bit product line. Then there were these splinter application product lines, the Edu product line and others. You had two fundamentally cross-purposes product lines. You had the product product lines, and you had the application product lines. By the way, the '66 to '72 era was a hard era to get another machine formed. You had all the resources all tied up. Who's going to do something else? It's the classic problem of "Shit, how are you going to do something new? The 8-product line is going to make the next 8."

Out of that came the proposal for the 16-bit machine. That was the famous X. The whole X story would be an interesting story. I've probably got some notes somewhere when I talked with Henry Burckhart a few years back about the X. I'm not sure Henry told me everything. Henry implied that if Digital had made the X they would have probably stayed. Ken basically forced the formation of DG. Couldn't have done it better. Here it is, [READING A MEMO ALOUD] "Re the DG Formation: As I said before, it would be unclear why the folks left to form DG. At Carnegie, Ed DeCastro and Henry visited me to discuss the new X. I blessed the X and wanted to see it built but the group did a number of things to get it rejected, i.e., telling everyone that it was a more difficult project than the PDP-6, which is the last thing that everybody wanted to hear. The X group did experiments with large boards which Ken was also against, having just switched to even smaller boards for a wire wrap machine. This further alienated Ken et al. I don't know what the group was doing at the same time as the X vis a vis raising money or thinking about DG. I'm sure they weren't designing the machine, as the Nova looked nothing like the X. Ken and Ed DeCastro aren't great communicators, and Ed was moved organizationally into an untenable spot. Having been responsible for making the most money for the company in the 8 line, the X group was paid the ultimate insult. Ed was put to work under John Jones who worked for Stan, both of whom he had no respect for. John was a bright MBA student of General Doriot with a physics BS, who made a market selling PDP-4, 7, 9, etc. to physics for pulse height analyzers..." That's kind of how I think it happened. Whether the X would or wouldn't have happened. But once it did, once the DG was out there, then DCM had to form to compete with it. That was an aborted horrible effort under a guy by the name of John Cohen who didn't know anything about computers. He had a little team doing it. Finally it became such a disaster, that Roger [Cady] who was handling the PDP-8 at that point, came over to handle the PDP-11 project.

Then the PDP-11 formed out of that, just in time to start to stop DG. The -11 still wasn't quite good enough to stop DG. Things like the LSI-11 were critical to doing it, and then the 11/45 was critical to doing it, and then when the VAX hit, it just stopped them dead.

MAN: Did the -11's emerge to then consolidate and pull together the scatteredness?

GB: Yes, there is a memo that I have somewhere. The only regret I have by the way is not keeping every memo that I had ever received from Ken. You get a feeling of his many faces. I do have a file with about that many of them. Have you read a lot of them?

INT: We have seen some of his correspondence from the early days up through I would say, late '60s, early '70s.

GB: OK. I had proposed the formation of central engineering, in February '74. It was all pulled together, at a Woods meeting in Bermuda of the Operations Committee. It rained there and it was cheap and easy to get to.

JP: And that was to consolidate resources?

GB: I came back from Carnegie in June 1972 as a staff guy. I had memory engineering and power supplies, two of my favorite things. I was essentially VP of engineering but yet I had no resources. So I played staff guy pulling together, looking at things in various ways, proposing different things. Then in February of '74 I basically proposed that everybody report to me. In that sense, I think, some of the entrepreneurism got lost, but it was fake entrepreneurism. Building another machine, that's not very entrepreneurial, or doing another version of the next release of Fortran. I look at follow-on the next one of the series given the technology shift, that's not particularly entrepreneurial. This fake entrepreneurism produced almost 10 incompatible operating systems.

Doing really different things is... deciding that you need a terminal, or a priner, for example, was creative. Phil Laut and I proposed that we get into the terminals and graphics business in the early '70s. Phil was wandering through numbers. He used archaeological filing, piles of files. He was doing our numbers, he was my staff guy. The line guys were just rip-shit at him all the time, because he didn't care that much about the goddamn numbers, about how they were running. Basically as a numbers guy, he was very good. He got into rebirthing after he left DEC, and has written five or six books. He and I got along very well, because we were both intuitive. "We think there's a market for this based on some of the numbers." "Let's get in on the terminal business." He had a good sense of working from all kinds of things. When Central Engineering got going then you needed enormous methods. We measured every goddamn thing. You could say I'm very intuitive on one side, but now, go look at the engineering plans. How much stuff do you see today from DEC of this kind of stuff? I ran the whole damn place on these curves -- or rather the managers ran them by the curves! I ran engineering relentlessly. For example Ken and I had strong arguments about the issue of whether or not people were allowed to use semi-log graphs. And I ran the whole engineering department on semi-log stuff. What's the state of the art? Give me that point! I don't think any of it runs now that way, or has run that way for a long time. Certainly as measured by the output of the products, there's not much attention to that goodness.

MAN: Who protected engineers from the influence of marketing and manufacturing?

GB: When central engineering started, then the trick was to make a good coupling between engineering and marketing. We had all kinds of mechanisms for doing that. The marketing committee was a place where the projects got proposed, to a series of what we called "Pots" -- different places where strategy was determined for various price bands, market uses, things like that. This was the board of directors for an engineering group. Every engineering group had a board of directors of all the marketing groups that were going to market the product. That was the basic mechanism, or one of the mechanisms, that was used for determining strategy.

We plotted enormous numbers of different things, like the revenue that a group was bringing in on one axis, and the amount that they were spending on the other, and then shifted that in time to see where we were spending money. Virtually everything translated into numbers. I used to say, I think numbers people are different than I am, because they get something out of numbers, but then I concluded that they don't. They don't get anything out of the numbers. They only get obvious, dumb things. "This number is bigger than that number." Lots of insight there! You know, what's the price advantage of one versus the other, and that stuff doesn't come out of a spreadsheet of numbers. I forced a tremendous number of metrics. Stan [Pearson] drove that planning process. It just forced an enormous number of different numbers and ways of looking at product strategy and product direction. All of that was aimed at the allocation process, and convincing ourselves that the group knew what they were doing. It was self-management. Do you know what you're doing? You force people to analyze something from all angles.

MAN: That was a good way of developing product strategy.

GB: The trick was always to get new things in. As we got larger, the research group was important, and then the formation of advanced development within a group. Right now I'm helping Microsoft form a research organization. So I look at what they're doing, and I say "You guys don't need a research organization, you need an advanced development function first because if you do research it's going to come up with a bunch of ideas and there's no way to get the idea into the product groups."

[END OF TAPE 1]

MAN: Once product strategies were developed, product managers took them over?

GB: Product managers came sometime in the '70s. Engineering was getting so complex in terms of the groups that it had to interface with -- marketing, manufacturing support, other product groups. They were selling, and so the product management function took over that whole collective set of activities, dealing with product logistics. Those were very hard jobs. Then the engineers were then free to go work on the product per se. In the old days the engineers did a lot of that stuff, "Hey, I'm proposing this, here's where it sits, manufacturing is going to build it for this much..." But as those functional groups grew, the damn things got out of control in terms of communication.

JP: Besides consolidating resources, was it your goal to provide a buffer to the engineers so they can engineer?

GB: Sure. That was the fast thing; in fact there was no time for engineers to do these other intergroup communication things. So you had a product management function within VAX that was the buffer, so the guys doing the engineering didn't do anything but build it. Over time, I suspect that got too large so engineers never saw a customer. In the earliest days, a lot of engineers interfaced with the customers. Gradually that got buffered away. Then you clearly had to have metrics to measure where they were going.

The product line groups had engineering. When I was there during the '72 to '83 period, the product lines did special hardware, special software. Lots of conflict there, because Ken very often wanted me to be responsible for all of these things. I had had enough of experience trying to do it, that I knew that I couldn't make it work. "Gordon, are you doing all these dumb terminals?" There would be a list of 17 different terminals. I looked at them and I characterized them, and every product group had its own terminal. Three or four guys, a little team off doing this shit. Some of them worked and some of them didn't work. I'd go on and try to influence them, and they'd say, "Oh no, you don't understand the market. It's got to cost this much and it's got to be like that. So bug off." So after a while he said, "I'm not going to spend any time on that shit, because there's just no reward for it." There was no way to deal with it. We just didn't have a good way of working it. A lot of the product line engineering stuff was very ill-conceived; some of it was an attempt to get crazy margins and do product differentiation when there was no differentiation needed. What they should have been doing all of the time -- it's clear in retrospect -- was the software, to go into the various markets to do unique things. That was the kind of character of engineering. Some products were important. The MINC was an important thing to get out, then the industrial PDP-11s were important. Masscomp formed, with a bunch of good guys from the DEC lab group. They self-destructed but the product was very good. They should have been great.

Under the product line managers we had a dichotomy. I can look now at how bad VAX was for the company; it took a lot of entrepreneurial energy and made everybody sell VAXes. It streamlined the organization, but that gave them an enormous number of resources, and they just blew it! They didn't make investments. I can make a very strong case that VAX was very bad for the company, the winner and the loser. It took no brains at all to run the company! I look at the era of VAXes as when the organization atrophied, went to sleep, and just got enormously bureaucratic and nothing came out. The product lines should have been out there groveling for software, building applications, making deals, and burying itself deeply into customer relationships. A year ago, believe it or not, I tried to help sell a VAX to Cirrus Logic, which I'm on the board of. I said, "Cirrus, for God's sakes don't buy an IBM AS 400. That's the last refuge of the idiots. That's the worst possible machine that you could think of. Here we are selling PC components. I would bite the bullet and make my whole company run on PC's. It takes a lot of guts, but it can be done." PC servers, etc. Believe me, all that stuff is available. You can get it. It takes balls to do it, but I would. Then they started dealing with IBM, and IBM scared the shit out of them. So they went with IBM. DEC made an OK proposal of a VAX/VMS running ASK, and a UNIX machine that was going to do ORACLE. So they had three packages running on two computers and a network. And I said, "That's OK. They've got a lot of computational power there. You want to go distributed anyway. This forces you there." Actually the killer in that sense, was a thing that I would have had trouble with, too, which is you're running PC's, you're running UNIX on SUNs, you're running UNIX on DEC, and you're running VMS. That would have brought in four different systems. IBM says, "Just run this little rinky-dink AS 400, and three applications into your PC network." That's what they did. Now the terrible negative thing that that taught me is that DEC had a lot of power in the various semiconductor companies around Silicon Valley. But what it made clear to me was that what DEC should have been doing was putting in a very strong field marketing organization that knew semiconductor companies, dealing with the accounts, the floor, the line, the inventory. They should have buried themselves into a bunch of markets or not have been in there at all. They didn't. They had kind of a weak sales organization and the salesman was out there scraping up software off the walls, bathroom floors and everywhere else and trying to sell it. And IBM just beat them on that. So DEC didn't get those applications from third parties, make the deals vertically into these organizations and knock off.

My model is around SIC codes and applications. There's a very strong model of how you do market segmentation in the new book. They just didn't do their homework.

If I hadn't have left, I don't think DEC would have atrophied. I can look at all the things that that screwed up and say twouldn't have happened. I can guarantee you DEC wouldn't be firing people. It would have probably killed me to stay, though. The idiocy of DEC today is so stupendous that I couldn't have let it exist.

MAN: Open systems: What did it used to mean, and to what extent were the seeds of open systems ideas sown back in the '60s and '70s style at Digital?

GB: Open systems came out of UNIX. Then the fact that UNIX was then out there, and then people started porting it to different platforms and then said, "I don't have to write my own operating system, I have kind of a standard." When we formed Encore, it was way ahead of its time, but it conformed because of UNIX, we can go out and that's what the world wants. They want the standards. They want to be able to port software from place to place. I'm not sure it was Open-standards-equals-UNIX at that point. I define open systems at a level of you can buy a system or component from more than one source. At the PC level, I can buy it from multiple sources. I can buy a platform and hardware. In fact, I divide the world up into this [DRAWS], and then I can show the dilemma of DEC of the 1990s. You've got platforms, environments, and applications. You have to have three levels. Here's the DEC dilemma today. Fundamentally, there are a total of 6-7 different application platforms, all of which will change from 32 to 64 bits. No way can the sales department assimilate these, or customers understand them.

MAN: Well, this is a hell of a strategy, Gordon, but wasn't that the same case in PDP-11 land?

GB: It was nearly the same, that's why I said I was going to run engineering!! DOS, RT-11, RSTS, RSX-11A, M and then D. We had a lot of environments and they were all different.

MAN: Not to mention the 10/20's.

GB: At least these were separated in pricing. The 10 and 20 were over $500K and the -11's were $50K - 300K. Some of these were actually price segments. RT-11 was a single-user system. But look at this, there's a little segmentation here. This is bigger systems, this is desk-top stuff, and you could say these are PC's, workstations, mainframes down to workstations and stuff like that. But that was at a time when you had a strong product line structure where these guys had figured out how to use them. They were even using the operating systems to segment their markets. So you had people who knew what was going on there. Who at DEC knows anything about any of those environments other than the VMS environment?

MAN: I remember a quote I heard you say that one of the dumbest things that Digital ever did was not to separate the VAX from VMS.

GB: Because we would have been able to make the whole thing portable, and then ease this transition. You could in principle then write the portable software. The fact is these are really different environments. That's the problem. These are different machines. Just as these were similar machines, but finally we had emulators for M to run RT-11 programs.

MAN: In PDP-11 land you just said let's go with 11M plus for compatability. Toss the others.

GB: Yeah. I'm probably the greatest computer genocide guy going. There's a kind of paradox. It's one of the most brutal things that you can do to an engineer: kill their project. But the worst thing for an engineer is to face the market and have a product fail in the marketplace. The guy would say, "Oh shit, we didn't get to take it to the market, what a dumb organization." But once it goes into the market and no one buys it, it's very hard. So what's the kindest thing to do, and what's the most profitable? Take your losses early. I remember two of those. The 11/60 and the 11/74.

JP: Bob Everett and Jay Forrester said that that was a really critical thing to recognize and admit when you've got a dog or when something's not working. Just stop.

GB: Get out. Exactly. Roseanne [Giordano] and I were talking last night and I said, "Roseanne, do you finally understand about the PDP-10, that I didn't kill it?" And she says, "Yeah, I think I understand that." It's well-known that I'm the guy who did the PDP-10 in. And she said, "No, you really didn't. It was the group, they did themselves in." You could say, well, it was bad management. You didn't select the right people. That's not the way DEC engineering worked. Goddamn it, I have never "put" anybody on any project. Engineers selected what projects they worked on. You can use coercion, you can do everything, break their arm and knees, but you can't force them to work on something they don't want to. That's my philosophy, and that's the way DEC engineering always ran. I don't believe in interchangeable engineers. Engineers are solely motivated by their drive or willingness to want to work and accomplish something on a project. If you fall out of that for any reason, namely somebody's made to work on something, or assigned, all that gets in the way of good engineering. When PDP-10 got cancelled, it was an absolute vote of no confidence on my part, I was confident that those guys wouldn't be able to produce a computer. They had tried for a number of years. They had no management that was skilled in design, and they had no designer or architect. On my balance sheet, they came up zero. They had all the DEC processes and stuff like that, but they had no skills to speak of. They had a good circuit guy. They had no logic. They had no architect and they had no management. So what do you do? You shoot them. Get rid of them.

JP: That begs this other question which is one of yours.

Is there an engineering ethic?

GB: Yes, it's commitment to the project and team!!! I think this ethic has transcended DEC. It's on the west coast to a lesser degree. I always thought this ethic mostly existed only within Digital. If you make a contract when you start a project, that contract says you're going to finish that project and make that product work. If you sign onto a project, you finish the product and make it work. That's basically the agreement. I didn't let people transfer out of projects. So to me that's an ethic. There's an ethic of what's important. You make good products. You don't leave the project until it's done, or until it's stopped for some reason. There was an ethic that said they had a responsibility as to the efficacy of the product.

When I went out to Stardent, Allen [Michels] introduced me to somebody and said, "Here's Gordon. My first encounter with Gordon is when he threw me out of his office." That was probably right. I didn't recall it, but anyway. At various times Allen and I got into arguments, and I said, "Allen, I hold the engineers responsible for the efficacy of the product. Not the marketing guy. No one's going to tell an engineer to make a product and the product fails and the engineer said, 'Well, look, somebody told me to make this thing.'"

That was the ethic; I managed to hold engineers responsible for the product's efficacy, and to stay with the product until it was done.

MAN: What happens when it was too hard?

GB: When it was too hard, management had a responsibility to continue support or the group had a responsibility to decide that it was too hard and that you should back off on it. Shut down or redirect, or to come forward as rapidly as possible with the knowledge that in fact the project should be changed.

The 11/60 is one that I tried to get killed. The guys were furious with me; they said, "Why are you trying to kill our project? You helped start it." I said, "Yeah, but you guys are two years late! We don't need it. It's just going to cost the company money to do it. It's not an interesting product. We shouldn't take it to the market." The product management were coupled strongly enough into the product lines that they said, "No, we're going to sell a shitload of them." They signed up. And at marketing committee I used to poll, "Are you guys really sure what you're signing up for?". "Oh yeah, my guys tell me we're going to sell those." I'd say, "I don't think you are." That's one that I was overridden on.

On the 11/74, which was a multiprocessor 11, a really nice product, we had 100 on the line. Every month I used to come to Marketing Committee and say to Julius Marcus, "Juli, are you going to sell those? It looks to me like VAX is taking off. I don't think we need to introduce the product even though I love it. Shouldn't we stop it?" One time he said, "Yeah, you're right..." Juli was always on the fence and one day he fell off! But basically I feel very strongly that you only introduce good products that will make money. In this case it was simply that VAX wiped it out.

MAN: What about the role of creativity, innovation? Once the strategic decisions were made how important was the detailed engineering communications? Did Digital do very well at that?

GB: The quality of the engineering in a specific group were totally self-calibrated. I understood exactly who could engineer products and who couldn't. Who would you trust to build something? As far as I was concerned it was all execution. Some people sort of vomited code and logic on a piece of paper and some people did it elegantly. I and the key engineers all knew who those people were. We could tell vomiters from artists. Sad to say, the poorer engineering managers couldn't.

MAN: What were some of the most elegant engineering...?

GB: DEC had a lot of guys that were really very good, very creative, wonderful engineers. It had more that its share. [Dave] Cutler, [Roger] Heinen, people who could really take a large problem, decompose it, and structure and build an absolutely elegant system on tragic losses. These guys knew every detail about a product and how to build it. These were the kind of engineers that I related to. It was just like the engineering I did, described in this notebook. I basically knew everything about the PDP-6, about architecture and implementation in both hardware and software. In the mid-60's it was easy to do that. It's harder now, but you deal with it in a different way. But Cutler was like that. [Mike] Riggle was like that. Riggle to me was probably one of the broadest engineers I know because his knowledge included the magnetics, and dynamics for disks plus error detection/correction.

[END SIDE 3 -- BEGIN SIDE 4]

GB: Fred Hertrich, who built key storage products, lives in Colorado. He was a wonderful, broad, German engineer. He had five guys working for him.

The guys who really knew engineering were people whose team never got more than 20 or 30 people, a small team, and were focused and produced the really profitable winners.

The problem with Marlboro is it got out of control, and never got the right culture after [Alan] Kotok left. It just kept getting worse and worse until it sort of burned up into the sun. The VAX 9000 was done there, and a financial disaster for DEC. It got enormously bureaucratic and out of control.

[You ask] who are my engineering heroes? For example, Bob Supnik and Dick Sites in the VLSI. Let me not go on record here because I don't want to slight the great engineers and engineering managers.

To be the penultimate, engineers did a piece of the work themselves, and managed the project. They always knew what was going on. Cutler did VMS that way, and then he did the PL1 compiler that was the back-end for many of the compilers. Cutler is probably the biggest loss to the corporation for three reasons: his work, the <Alpha forerunner> project that Ken killed and had to restart, and NT. He's just got NT running that will drive another nail in the VMS coffin. NT is for New Technology operating system, and he works for Microsoft. That will end up on dozens of standard platforms to compete with DEC hardware.

MAN: You manage and you do it, too.

GB: That's exactly my model. We just talked to a guy for heading a research organization. He said, "How am I going to do my research and manage this thing." We said, "Don't take management that seriously. You'll be a shitty manager. No one will respect you unless you have a piece of your own. Think of how you're going to run it in halftime. You should have your own project. Do that and get help around you to do any of the stuff that you need to do." To be a successful research manager, you must create yourself. That's also my model of engineering management. That holds right up at the top.

MAN: Projects getting more complex, you said there were more team members needed. But was that true... that was also the birth of CAD systems, and...

GB: Sure, CAD was coming in and helping a lot of that.

MAN: What was the role of that? Did that allow you to work smarter?

GB: No, it allowed you to build more complex things more reliably.

MAN: But it didn't necessarily take more people?

GB: No, in fact, with CAD you were able to contain the time and the people and deal with the increased complexity. I'm not sure anyone in upper engineering management ever figured that out, outside of the guys in the VLSI. I don't know what their project sizes look like now. This really came home to me after I left DEC. The phenomenon is well documented in my new book and in the February 1989 IEEE Spectrum article by me. When we started Encore, a few people came out of DEC to build their product, Multimax. Charle Rupp came out and built a great large-screen terminal. They did the projects with very, very tiny groups. The Encore group, with hardware and software got to be 30 or 40 people, and that machine is still better than anything DEC has as a multiprocessor. The 6000 has six processors. Multimax with up to 20 processors is still a great machine and much better than the 6000. Better in terms of bandwidth and throughput. It was done with 40 people. At Stardent when we shipped Titan, there were 45 people. We did a compiler that was better than anything DEC has. We invented a whole language for dealing with graphics which a number of companies have licensed. We built a vector MP and the architecture for that and all the software. A very complicated machine. 45 people. The secret I've learned, at least after DEC, has been that rarely should you ever have more than 45 people to build a machine. KSR's engineering staff is about that size...and they've produced a revolution.

MAN: What's the role of the individual then and what is the role of the team? Is the role of the team to promote individual inspirations?

GB: The team is a collection of individuals. It's the role of "management" to see that the right resources are there and that people are cooperating and that the team members are productive and creative. They must have the right goals and plan. That's how I see it. In a hard project, what you want is excellent manager, leader, builder, and three or four people working with them as engineers to be part of that. That's the team, and it really functions. That's the model that Cutler used on things like the compiler. The start compiler is a great compiler and it was four guys. The CPU ended up with five or so. After a certain point you may need many compiler people doing different things, but...

JP: If we go to the good side of the VAX project for Digital, I think it was that there was a clarity of vision and there were a bunch of people who operated towards that same goal.

GB: Oh, yeah. Have you looked at the whole VAX strategy documents? Do you have the original VAX strategy paper? There's two things about VAX. One is the VAX architecture itself, and VAX and the VMS, the project -- the 780 -- that launched VAX/VMS. But that's just another computer. The main thing, what made the VAX work was the VAX strategy. It was approved by the board in 12/78. That was the vision of the single architecture. That came out of the trip to Tahiti. This is described in Rifkin's book. I have had enough "ah-ha's", invented enough things. The method in the High Tech Ventures book is like that; the ability to characterize a company in many dimensions, plotted this way and growing accordingly, is an invention. Unfortunately I can't tell you what day I did it. I can show you it in my notebook. It was on a trip going to the west coast. I was at NSF and I was coming back. It all kind of evolved about a DEC memo that I had written years ago called, "Heuristics for Making Great Products." Do you have that memo? Russ Doane, another guy to quote, had commented on that. Of all the things, [to look at], that's one. It's a "Folks, here's the secret. Here's how you do it. It's only 35 rules." My other inventions all had a similar pattern - generalization. I count my inventions as: a generalized flip-flop, the Unibus, general register a la PDP-11, and the VAX strategy/hierarchy. A few years ago somebody sent me a memo I'd written, called "NOD." No Output Division. I had just attended some meeting where everybody was sitting around in a review meeting, and this poor team was trying to get a product done. Everybody's commenting, "Well, I don't think you can manufacture it." "Well, I don't think you can sell it." "Well, I don't know whether I can serve it." And so I said, "Why are we building this?" It turned out it was an important product, but all these dumb reviewers were doing was covering their asses.

But back to invention. There was an incident in Tahiti where it was sort of invention; this is the computer hierarchy. Now that had been building in my mind four or five years. I had given several papers on distributive processing. I'll show you this one. It's an innocuous paper, dated in '75 and never published any place important. Basically the idea of a three level hierarchy was in that, that is you organize machines this way, and then the need to tie those together with one structure and then be able to do all the things that VAX was going to do was there. This was a business plan, essentially, for the company. Here's the VAX strategy. That way you can have 100,000 people working on something if you can state it simply; this stuff is basically called leadership. People can follow if you've got some basic plan...I mean shit, I can't go and tell anybody how to do it, because most people haven't enough depth or breadth or creativity to understand it.

JP: It's something that everybody can get their arms around at whatever level they are in the company.

GB: Yes, but VAX ultimately was a disaster because people gave up thought! The trick was to see what happened to the computer industry. DEC really didn't respond to any changes outside. It was really a tragedy. One saw this army of changes -- PC's, UNIX -- marching down. Everything was changing about them, and they didn't see it at all. That's the craziness. It's tragic. I tried to warn them. Certainly if they had done any of the things that were in my Computerworld article, then they could have gone marching down the line. It would have worked just fine. They wouldn't be in a whole bunch of markets, they wouldn't have that mess there, but they'd be making a lot more money now.

MAN: See, it's 5:00. Right now, Patrick and I have just been laid off from Digital after like eight years ago I began.

GB: You are?

MAN: The week you left was the week I began formally. Isn't that great? So we appreciate what you're saying. We can relate to it.

GB: It's a kind of stupidity. Hiring and firing by spreadsheet. The spreadsheet mentality used to drive me absolutely crazy. You just take the numbers and run them out, and allocate percents, independent of the group. The fact that Jack Smith has any power is absolutely criminal, the ultimate spreadsheet mentality. This reflects the fatal flaw in Ken's personality. How did the company got out of control? The controls were absolutely clear. Even if they had no plan, the numbers said they were out of control. What was the controller doing? The fact that these guys are paid **anything** is just a crime! They should be paying the company, because they wouldn't have anything else to do! They are just so incompetent that it blows my mind. The myth that a Ford-trained controller is any good is crazy.

MAN: I love the theme that you've constantly had in going from closeness to customer into real requirements versus isolation of that. The interesting reasons for both.

GB: DEC's problem in the future is how do you get the engineer closer to the customer or the guys building the products, or even the marketing people who can help sort that out?

MAN: It sounds like a lot of what's called systems integration nowadays was to a large extent doing custom CSS, software services in the '70s?

GB: Yeah. It's the economics of who supplies the software. Where did the software come from? Now the thing is that virtually all application software comes from deep understanding by small start-up groups. You can't do that in a large company. We used to talk about application software, and rarely can you do that in a large company. It means that you have to have chemists that are going to do it, and you can't get that mix in a large company. You can't manage it. There's just too much cultural diversity. Also, it's almost impossible to mix hardware and software sales. Sun's done the right thing with Sunsoft.

JP: What in your mind were the periods of engineering greatness at Digital? Which can you identify?

GB: I think the VAX strategy... Last night at the [Computer] Museum, I met a guy doing planning in Boca just before the PC. He said, "God, when you guys came out with VAX with one architecture, we knew we were dead.

We didn't know what to do." Then when I was at NSF, I met an IBM guy who came and said "What should we do against that?" And I said, "I've thought a lot about that. I wrote all the things I would do if I were IBM, and that's in the VAX strategy document. Too bad you don't have that! Of course, it was designed to drive you guys bullshit. I'm glad it did!!" I got a kick out of that.

The other periods were: '60-'66, formulation of computers; and the product lines during '66-'70.

JP: What happened with the PC's?

GB: The PC's were an unmitigated disaster.

JP: Was that engineering entrepreneurialism at work?

GB: There was a lot of entrepreneurialism, or perhaps it might better be called court politics. Everyone focused on trying to please Ken. What do I think about that? Ken was the sole manager, or king, of the PC's. I want to lay the PC disaster totally at his feet. He really drove the PC strategy, and the development of it. This included getting Stan to open the stores, telling Andy how to run PC's, and ultimately forcing Andy out when the poor products didn't sell.

MAN: What was the role of central engineering then? Didn't you say anything?

GB: I tried. On the other hand, I was working on VAX and the strategy including VAX clusters. Every six months he came to me and he said, "I want you to manage all of it. Everyone should report to you." Already I had about 8 direct reports. I would get about five more. Something like that. I remember writing a memo and saying, "These guys ain't going to work for me." I think at the time it was Clayton who running it; I said that Dick was the best guy to run it, and we should help him run it. He said, "No, I want Avram reporting to you. John Clarke, the terminals person, all these guys reporting to you. If you run it you'll make it work." I said I couldn't, I was just totally overwhelmed with the difficulty of satisfying all the constraints -- especially the organizational ones and Ken's brasses such as pushing the PDP-8 as a PC.

MAN: But there was some strategic link missing. When Avram Miller came to you and said, "We're going to sell 3000 of these the first year." When he got all of the good PDP-11 engineers under him, when they decided to re-engineer everything again, what was going on?

GB: It turns out that he had a very famous box designer, Ken, right by his side to design boxes for him. Avram put a whole new bus structure in the PRO. The software was new and it was just a disaster. I had meeting after meeting on the software to try to get that into some shape. The bus was a proprietary bus. By the way, at the same time, I was trying to get the PDP-11 licensed as a chip. I said, PDP-11 ain't going to go nowhere. That whole thing was clear. I was voted down everytime on the chip license. The PDP-11 ain't worth anything. It isn't worth anything as proprietary architecture. Get it out there. If we do it right we might be able to stop INTEL and Motorola. But we had no one over on the marketing side. Ken was busily shooting marketing people, opening stores. We spent all of our time arguing about the PRO versus the Rainbow versus DECMate. And Ken wanted them all to win, saying DECMate is the right way. We're going to put application software on it. And I said, "Ken, the machine is braindead. It's run out of its feeble old memory. You can't write programs for it." In a sense all of them had that problem, although the -11 was one we could work with.

There was the original billion dollar mistake, which was making all three PC's and Avram and his projections and all that crap. That was in conjunction with the marketing guys. Meanwhile Ken is flogging the marketing guys to sell what Avram wanted made. Couldn't make any decisions about how this thing is going to be distributed. So if there were minutes of the Operations Committee, [they] would reveal that in fact all the time was spent working on that. We had the marketing committee. The Operations Committee was spent arguing about how important these PC's were going to be, and all the product lines wanting a piece of the action, not wanting to give it up. Then to say, Andy you take it and figure out how to make it work. It needed to be divisionalized. The company was arguing about it. Then they'd use the machines to pit each other against... And Ken wanted the PDP-8 to win, and we had this marginal word processor. I said "Ken, that's a terminal. Think of it that way. Get the cost down. Don't try to make it do Dibol, and do all this other crap. It ain't a computer for business." Ken's lack of understanding of software and where it was and could come from is probably the single most important cause of the failure in that thing.

MAN: But Barry James Folsom's vision was kind of close.

GB: Barry's was pretty much right on. After I left, [Ed] Fredkin went out to DEC and I think I even went out to DEC, had breakfast with the Operations Committee and said, "The PC wars are over. It's clear what you do.

Absolutely clear as hell." The billion dollar mistake was one thing. That was an organization and product mistake, when the whole market was trying to decide. We lost. In retrospect, we may not have had a chance there, given what IBM did. So you could say look, no matter what, we would have lost that particular battle. But once the PC had formed it was absolutely clear in '83, '84, exactly. History was written. It was so clear. The tragedy is that DEC didn't see it at all. Even HP saw it and prospered by it!

MAN: And ironically the Rainbow was a much better engineered product than the IBM PC.

GB: No. Not really. You can't argue that one way or the other. The important thing was that it was incompatible. The bus standards had been set, the BIOS had been set, and that's it, you just go do it. Don't think. Just execute. It was a good strategy, but it was an ASCII terminal. A PC at that point was a commodity. It was something you sell by commodities. DEC had really good low-end engineering in Clark's group in Taiwan and in terminals, and the ability to produce things low-cost in all kinds of places. All DEC had to do was to do that. Talk about what happened at DEC, why it went braindead during the VAX era, the PC... forget the VAX... I'll share in the billion dollar mistake. I'll take $300 million of that error because I was working on the VAX. Ken gets $300 and the marketing guys get the other $300 million...

[END OF SIDE 4 -- BEGIN SIDE 5]

GB: It was being built as a chip at that point. This was before the MicroVAX. You'll have to look at the date. But the absolute faux pas was not adopting the PC. There is no excuse for that. Zero excuse for that. We had mismanagement everywhere in the organization. I just can't fathom what went through anyone's collective minds. I know Fredkin went out there and begged DEC to do it. Because he saw that that was so critical. I saw it was absolutely critical. It was an absolutely no-brainer. I would say that was the single most important error that DEC made in what it did. The 9000 was probably the second most important failure. Just wrong technology.

MAN: Was the Trilogy technology that poor?

GB: They used some technology from Trilogy, but they did a lot themselves. It was too little, too late, and it was the wrong technology. At that point it should have been CMOS, or it might have been saved if it had been executed on time. I don't know. That was a major faux pas.

If I put down what was really at the root of DEC's demise, it was the destruction of the product lines. Do you want this in here, or not? It's what's happened to DEC from an outside perspective.

JP: One of the interesting things that I'd like to get which I don't think we've really covered is the engineering environment in a general sense at Digital. You've had some opportunity to compare it to other environments. In the early days there was a lot of creativity. There was a lot of entrepreneurial spirit. To what extent did Ken contribute to that? Did he come in real close to engineering at times and then step back? Was he always on you guys?

GB: He was. I'll say that three-quarters of the engineers that Ken got intimately involved with were turned off by him. He was not a good guy to engineer with. He would come in, and overrule and play 'I'm the boss and we're going to do it my way and you guys are stupid.' There were a few people that he got along with, and that could work with him. There were people that absolutely detested him and thought he was a lousy engineer. You ought to ask people, and also ask poeple how they felt about me in terms of the engineering environment I provided, the kinds of things that I did.

I too liked to get involved in every project. I liked to understand, review and help the areas. Ask Riggle, Strecker, Stewart, Lary, Grant [Saviers], Cutler -- guys I worked with on projects.

MAN: How would you describe the engineering environment that you think that you set up?

GB: OK. Basically the environment that I tried to create while I was there may have come from, I won't say a university environment -- because university environments very often can be very, very uptight and closed environments -- but the goal was an absolutely open, free exchange of information [place] no hiding, anybody can look at what anybody else is doing. I used to say that the place leaked information like a sieve. We didn't try to control information, especially from group to group. Everyone was free to criticize/review.

JP: Even with customers. DECUS was a perfect example, an opportunity to get software written and utilities written for computers.

GB: Our goal was very clear. Why did we form DECUS? We've got to get a bunch of software for the PDP-1 and we've got to share or we're not going to get it! In fact, it was built exactly on the IBM Share model. Why are we here? We're here to share software.

JP: Was DECUS seen as a valuable --

GB: Absolutely, we started it that way. The other thing about DEC engineering was that it was created as an open non-competitive [environment], lots of information, free information exchange. That's what the engineering committee was for. How does that compare with other organizations? In general, I think they vary. But I was informal. I tended to not be concerned with dress, or wear ties, or be stuffy. My uniform was "turtlenecks in winter, t-shirts in summer, suits if I had to talk to stuffy customers!" I always throw away the vest when I buy (at Filene's Basement) a 3-piece suit. I liked the environment we had. I hope others did, too. I know that people who didn't like it, told me -- and we tried to change it. The companies I've been involved with have tended to be that way. Very casual about when you come in, no time clocks, none of this stuff. But there are engineering departments that are not like that. They have hierarchies, call people Mr./Mrs., wear suits and leave at 5:00.

JP: IBM is certainly different.

GB: I think it's relatively more structured. At Ardent, we were trying to sell some stuff to IBM, and I was really very unimpressed with the PC engineering, though IBM does have some great groups.

JP: When you first joined Digital, what attracted you?

GB: Size and responsibility. I could go and design a computer _myself_ and write software _myself_ and get it built.

JP: You were 25, 26 years old too.

GB: Exactly. It was great. I could go and design a computer. I had worked as a co-op student at GE. In fact I had decided kind of not to be an engineer after working as a co-op student at GE. I was a Fulbright scholar, came back, started down the Ph.D. route at MIT, and I thought all engineering was like GE. I was building a tape unit for TX-0 and I [met] Ben Gurley. Ben Gurley was an absolutely wonderful engineer and wonderful man. I really liked the people: Ken, Stan, Harlan Anderson, Dick Best. The freedom and the range of all this work to do. In a sense, I didn't really want a Ph.D. I wanted to build things, and so, faced with all that and doing research, I thought, well, research is okay, it's better than working as a drone in an overstaffed engineering organization that supports itself with government contracts.

MAN: I'm wondering how computer architects are made if not born.

GB: I don't know. I haven't really spent a lot of time thinking about how that. A certain set of skills is critical. The undefinable characteristic; taste is critical. Ideas about simplicity and elegance.

All the patents or things I've been involved with turned out to have been taking an idea and generalizing it to the extreme. I have a patent on a thing called a multistable state device, which takes a flip-flop and makes it an n-state device. We actually used it on PDP-4 and -5. Knowing when you do something, what its function is, and how to make it more functioned, and the idea of elegance, making a part do more than one function -- definition courtesy of Russ Doane. The multi-stable state device, Unibus, general registers, Ethernet. Ethernet wasn't mine per se, but I wrote the paper on it when we introduced it. I said Ethernet is the Unibus of the '80s. Now Ethernet is the UArt of the '90s.

I also invented the UArt for the ITT system in 1962. I don't know how to train architects... I know some people have absolutely no affinity for it, or understanding of it or anything like that, or need for it. "We'll just go and build a bunch of stuff." Fred Brooks and I share the same views about this, I believe. We were on a technology advisory board for a little company, and the first thing we asked was who was responsibility for this product? Who's responsible for the vision of this product? We couldn't find anyone. We said kiss it goodbye. It'll have no integrity. You've got to have somebody who says I'm going to be responsible for seeing that thing through. By the way, my feelings about architecture, that's why you want to read the book on High Tech Ventures. I went wild when I wrote the pages on architecture. It gives my feelings about then. By the way, being able to implement is almost essential for an architect.

MAN: You've hit on most of the themes that we've encountered by the other groups, your peers and ex-peers. I love the continuity of themes and the strange mix you personally have between discipline and intuition. Between being a technical guru with a business perspective. Between being an architect and yet a builder, a manager and a doer.

GB: I hope my prejudices are in the book, in terms of everything I know. Particularly in start-ups, when you're hiring people, I put it down in rules. I put it down in laws. The head of technology should do this, the engineer should do this. Here's the things that I think you have to be good at. You must pass these tests. And I feel very strongly about the head of engineering being able to do things. You have to be able to go in and play one position. Write a piece of code, write a spec, whatever it is. That was in the start-ups.

[The following section added after the interview]

INT: Any more to say?

GB: Yes, I can't let this interview stop without condemning DEC's oerpaid, incompetent top-level managers who have screwed up the products and decimated the company. They've caused tens of thousands of people to be hired and fired because they were not looking at fundamentals of productivity. This is [illegible.]

The basic screw-ups were:

1. Worst overall control in the industry.
2. Not doing the right thing in PC's. This was clear within a few years -- certainly by 1984 -- that the standard was established.
3. Not exploiting the VLSI capability by using it to build multiprocessors. This would have saved both the

high-end (for TP) and workstations. This is described in my article in Computerworld that I hoped DEC would read.

4. Not finishing Cutler's machine and then having to restart it as Alpha. Then, establishing the architecture as an industry standard.

5. Going with MIPS, a thin company with a dead-end architecture.

6. The 9000 was stupid. Wrong technology and then poorly executed.

7. Destroying the product line structure and the ability to acquire and sell market-focused software.

8. Not getting much dominance in the commercial space, and allowing IBM to propagate the AS400 on the world.

9. Inability to take important, standardized technology such as DECnet and keep it proprietary so that DEC is forced to implement systems both proprietary and for standards.

10. A corporate guideline, called the first rule, that used to exist -- "Do what is right in every situation, employee, vendor, and customer..." -- just isn't followed. When a small company comes to me with the question "How do I make a deal with DEC?" I say, "Don't. They'll take forever to decide and then end up screwing you." The company simply has no sense of right or wrong at the working level.

[END OF TAPE]

DEC -- Gordon Bell, 6/28/91
Tape 1, Page 1

GB: Probably you want [to get on tape] Larry Portner, Bob Puffer. Andy Knowles was important, particularly in the way we worked, there was enormous stress ~~in~~ between marketing and engineering. The more you can try to center it into the different activities, the better off it is. Because it's a complex story, and you want to try to keep it that way and then talk about the interactions between the various forces.

There is a way of doing it a little bit chronologically, too. ~~I'm going to give you is -- I think there's dates on here.~~ -- when Glenn Rifkin approached me about his book he had an outline. He wanted to come interview. I said, "No, don't do that. I'd like to think a lot more about this. Why don't you give me every question you want and I will answer the questions and think about them in that context." I did that. He had an outline and I gave him comments on it, how I would organize it. There's a lot of stuff in here, essentially a lot of stories. As we go through the questions I'll try to remember it. He came back after that several times. (Bell Hands >20 pages of notes that he wrote for Rifkin's book)

I talked to Henry Burckhardt the other day, and I was out talking to Dave Cutler. ~~Cutler's still very angry at DEC.~~ Henry said, "Well, it turns out it takes eight years to get it out of your system You're just able to be calm about it now." So it's been roughly ~~actually I guess~~ eight years, and eight days now since I left. ~~I found out~~ The other thing is the less I become a DEC stockholder the better I feel. If I had purged myself of DEC stock a long time ago I would have been much richer, but secondly, I might have just totally ignored the company. Maybe when I finally don't own any DEC stock I may be able to feel good about the company.

INT: I think maybe one different perspective from the Rifkin book is instead of a political history we're trying to recreate a little bit of climate.

GB: I think that's right, absolutely. Everybody I think that you would ask, "Would they do it again?" Everybody would say absolutely. No question about it. I was involved in another company Ardent, which became Stardent. Crazy company, but the greatest group of people I'd ever worked with, outside of the VAX team. But we accomplished an enormous amount and we feel very proud of it. Some idiosyncra~~cies~~tic, crazy people running the goddamn thing, that you would rather not have to be there. That's the same way I feel about.. DEC ~~no~~ sometimes virtually everybody feels enormously proud of what they accomplished in that environment. With that as an overview, plus the ~~overview~~ understanding that Ken really was able to manage that environment for a certain time -- he was really a great industrialist. You'll see that in this [gesturing to paper given to interviewers] an enormous amount of conflict we had about working. Maybe one needs that stress and conflict. You probably do need ~~that kind of~~ stress and conflict, ~~but~~ and that was one of the things that determined the environment. Lots of it was unhealthy!

[Still going through papers to give to interviewers] There's this article in Computerworld. I wrote while at NSF in 1987? This is my current vita, which is how I regard all the things I've done, so it's long, which is everything including books and articles.

INT: We want to focus on engineering.

GB: Great.

INT: How would you describe DEC's historical approach to engineering?

GB: I hope you'll search the archives, first. For example, we ~~had~~ wrote many handbooks on doing engineering. Let me urge you to use them before you go to the oral history.

~~GB:~~ I think ~~it~~ the engineering approach varied over a long period of time. I kind of view the periods as almost associated with these major organizational changes. There was the modules, getting the company started, which I really don't know much about. There was the period of when Ben [Gurley] came, built a prototype PDP-1. I came a few months after that, and started working on the -1, and worked that whole structure, and the -4, -5 and -6. That was until '66. I left then, but the main change was the organizational change which was critical.

The products and the organizational structure are highly correlated. That's why I think for your own sanity that you may want to break it into these periods. Because where there was great organizational turmoil, there was a change in the way things were done. I would break it, Pre-computers, Computers from '66 when there was an organizational change into a product line structure until '74, and then the product line structure was getting in the way. Then we went back to kind of a functional organization, but with the market product lines. That in a sense got obsoleted by VAX in the early '80s, when we had all these people painting VAXes different colors, and pricing them differently and they weren't focusing on the market. At that period the marketing groups got destroyed, and not replaced by the right form of marketing group. I view the period from '82 or so, called the VAX strategy, which was all momentum. Denny the Dunce could have run the company. It didn't matter. It was standard, turn-the-crank, evolutionary engineering, nothing very creative, just slug it out. Then the PC's coming in, not being able to be integrated ~~very well~~ at all. Then this period of chaos that the company's been in for the last, two or three years; recognizing there are these things called standards, and UNIX and open systems and oh my God, what are we going to do with our life? We need a new strategy. There was a period of searching going on. There's an interesting paper on the growth of companies, and it talks about periods of growth, periods of unsettled change and floundering around, and then another period of growth -- you can go up or down during these periods ~~things~~, of course. It's companies trying to find out how to deal with this new environment. Given all of that, that's how I divide the world. Clearly the future is in question.

Let's go back to the question. The computers during the early '60s when we created the -1, -4, -5 and -6, architectures. that was a model of: "We'll get market by creating all of these products." It was quite an entrepreneurial environment. We had "Computer Special Systems developing products like Merry Tutors ~~working in there.~~ I remember doing a one-page pricing sheet once for new projects, ~~a one-page pricing~~ where you'd say how much is it going to cost, and then you'd kind of multiply that by 3.3 and you'd look at how many you thought you were going to sell, whether it was one or not, and put the engineering costs and price the option. That way of coupling engineers to market and products was great. ~~-- in the very first days,~~ It was coupling projects to a real customer demand. Like the -4 was built to sell to Foxboro and Corning and a bunch of other people in process control. The -5 came out of a special controller for Chalk River. The -6 was really our first attempt to build big computers. None of us knew a damn thing about software. We could write

This is described in the book, Computer Engineering programs, we could write compilers, etc. ~~we could do all this stuff~~. None of us had the foggiest idea of the issue of software, the cost of software, and the idea that there was a ~~balance sheet item that you really wanted to accumulate, and that a computer had a~~ kind of a balance sheet associated with it, that this code was worth something, you want to accumulate it. There's a lot in this new book I've ~~written about~~ DEC and the various ~~engineering~~ environments you can extract from, ~~particularly that one of~~ and McNamara wrote. We discuss my confession — 'Oh my God, why did I do a PDP-4 rather than just making some changes to the PDP-1, reimplementing it, because it was kind of an engineering thing.' I come down so hard on engineers from time to time about incremental improvements; the only reason I do it is because I know precisely what goes through their head. I've been there! They say, "Oh yeah, boy, if we do it this way we'll save six diodes, three flip-flops, and oh, we'll save $1,000 on every one we build. And oh, we'll get to rewrite the software, and it will only cost $3 million dollars!" "How many are you going to make?" "Well, ten. or a hundred"

Engineering was initially very entrepreneurial, then got into the product line structure, in 1966. which was again quite entrepreneurial. This is described in my letters to Rif Kir

JP: The PDP-6 was a real departure from the kinds of things that we were doing. Would you consider that the first real engineering risk the company took? Because that was a big investment for the company's size in relative terms.

The 4 was crazy. The 5 was a real contribution — it was the first mini to be used as a component.

GB: Absolutely. That was a big goddamn risk, in the sense that the PDP-1 was a copy of the TX-0, taking the modules and making that work. ~~The 4 and 5's were~~ [With] the PDP-6 we said, "We'~~r~~e been doing these little timesharing systems (the PDP-1 specials) and let's make a real computer now." I don't remember ~~how the~~ hell we ~~got started on it. I don't remember~~ why and how the PDP-6 got started exactly.

JP: But somehow the powers that be let it happen, right?

GB: Yeah, I know I proposed it, and set off and started thinking about it. Then got Allan Kotok got involved and then it started going. I think it was at around the time that the MIT CTSS was coming in, and we said, "Let's make a time-sharing computer," because we had made one around the PDP-1 that was too small in that it just wasn't big enough to do the kinds of things that we wanted to do. You needed a real calculator. The PDP-1 didn't have floating point. If you're going to share a system it needed to be able to execute Fortran at a reasonable rate. So the smaller linked machines just couldn't run fast enough. So we proposed it and got to do it. ~~So there~~ was that.

Prior to '66, extremely entrepreneurial. We had a good special systems group doing a lot of stuff. Then this idea of using a computer to do other things became clear ~~at that point.~~ I remember the PDP-5 came out of Ed DeCastro, its project engineer was an applications engineer assigned to build the Charles River Special front end. that. Using that to build special logic you'd use the computer and program it. That came out of a lot of our backgrounds, it was something that I learned at MIT when I was doing speech research. Special purpose systems aren't worth building, you just ~~use~~ program a computer ~~and do it that way.~~ ~~progr~~

INT: If the -6 was a change was it driven by any customer need or just engineering?

GB: As I recall, we didn't have any customer that wanted it.

INT: Australia comes to mind.

No they were the first buyer!

GB: That's how bad it was! We couldn't find anybody who would buy it! Initially in the business plan there was an 18-bit computer and a 36-bit computer. In '62-'63 there was a time when we had proposed this PDP-3 to Cambridge Research, and lo and behold they ordered the We can't ~~built~~ build it! goddamn thing. We said, "Holy shit." Harlan Anderson and I went over to them, and we were driving down Route 2, AFCRL, and I said, "Andy, what we're going to do is we're going to sell them a 36-bit machine. It's called two PDP-1's. 18 plus 18, it's 36. No problem, we're going to ~~dump them together~~ change the contract and we're going to solve it." which ultimately resulted in 20 PDP-1's. We had gotten the ITT order. I was the project engineer of that, and we were just sort of squirming to get that out. Here we get this whopping big order for a machine delivery in nine months. Of course the PDP-3 was just a souped up PDP-1. I didn't think it was particularly good. So we sat down with the guy with our new purchase orders and said, "Sorry, we made a slight change. We know we proposed this a couple of years ago when you set about getting the contract, and ordered it, but we're going to make a slight substitution." [LAUGHTER]

MAN: What was your role in going into the PDP-6?

GB: It was my project. I was the architect, and chief implementer and project engineer. My notebooks are around somewhere. I don't know if I know exactly where they are. "Procedures for engineers using a notebook." This is 1963. Here, 6-16-65, spreadsheet in terms of time, original specs, time, present time... The documentation for the birth of PDP-6. It looks like that's what it is.

We always tried to make DEC engineering highly entrepreneurial. There are a couple of slogans that I'm very proud of, one is "He who plans, does." This was coined by me in 1972 when I came back from CMU. Then I had a one-page memo on the make-buy, what you should be making versus buying.. a policy. Trying to make engineers be responsible for what they did. Totally responsible. Engineering in my view always had a lot more responsibility than it had control or ability to execute, so you had to do this otherwise you'd have all this tremendous fingerpointing.

But during that time there was a strong product-centered kind of responsibility, hierarchical thing, somebody being responsible for the new set of modules. Then when we did the -6, there were the circuits guys. Dick [Best] managed the circuits guys. I oversaw all the computers I directly managed the PDP-6 ~~computers~~ project, did the architecture, the logic design of the processor, everything except the floating point. Alan [Kotok] did the floating point. Dave Brown did the memory controller. I'd say "Dave, how's it coming?" "It's not done yet." So I ended up doing the memory controller, and then there was somebody else doing the tape controller. I'd say, "How's it coming?" "Not very well." I ended up doing the tape controller! I designed virtually all the logic of the machine except the floating point unit.

Then we started putting it into production. I think Alan was responsible. also, Bob Savell, I believe. Then I went down and started working on software. That's when Dit Morse, was trying to run the operating system. The notion of a operation system, per se, and then separate compilers and utilities had a structure and interface to it. We were trying to get this operating system up, and every~~day~~ nite Dit would come up and change the calls. The next day I'd ~~have~~ hear: "My compiler, I can't use my compiler, what the hell is this?" This happened over a period of time. Dit wasn't able to to manage his time. I worked on the data structures and lay out and what the calls were going to be and all that, and Dit was implementing it. I finally said, "Dit, I'm going to run this project, get out." He's the first, and only guy I ever fired. The irony was that Dit was a really smart guy. I've spoken to him since then. Then when I came back in 1972 and headed engineering, Larry Portner, who was running part of the software on the -6, came to me and said, "I'm going to hire Dit to do a file system." I said, "Hey, that's great. He's certainly a bright guy and he clearly can do that." Six months later, Larry comes in and says, "I fired him." And I said, "Well, maybe you learned something, that was yours, it isn't mine." But Dit had done a reasonable job. He had built the architecture for the file system, the PDP-11 file system, which is probably the base file system that we still have for everything. But it was funny.

MAN: So you had trial by fire in teaching yourself software?

GB: I had written a compiler when I was a Fulbright scholar in Australia. I'm not a software guy, but I occasionally program.

[END OF SIDE A -- BEGIN SIDE B]

GB: I'm not a classical engineering manager. I don't know whether I would work for me or not.

JP: That's an interesting point. Besides brains and ability were there other characteristics that you looked for when you were building your team?

GB: I guess in different times, while you're doing a project team you're weighing project kinds of people and you look for somebody that's ~~(~~good at that~~)~~. Every month or so I'm involved in starting a project up somewhere, basically it's the same old stuff. Do these people really have a very strong technical base? Do they have a process? My view of how you engineer is pretty much embedded in ~~this~~ the new book. The thing I think I have is the ability to take a lot of complexity and to structure it into smaller problems; at least that's how I see myself. Things have to fit together. I ~~fundamentally~~ think I'm an architect. That's an intuitive thing. There are times I'm an engineer, if you have to look at waveforms, or go down deep and do something, I can do ~~that stuff~~ it. There are pieces I don't feel comfortable about doing in software now. But I mostly understand things from the electron level to the integrated spreadsheet stuff. So I'm interested in all computing, the chain and how you build it. Structuring that complexity. All the books I've written have had that flavor.

MAN: You evolved that role because you stepped in and you just did it.

GB: Right. When I went to Carnegie, there were a bunch of people in marketing leaving because it was an organizational change to the product line in 1966. That wasn't the issue, [though], nor was it the issue when I left in '83. I thought the company was in great shape [then]. [I left because] I had fulfilled the contracts that I had made with myself. In '83 it was clear, ~~that was the~~ after my heart attack ~~thing, and~~ that I wasn't able to deal with the stress. I think I just saw it wasn't working with Ken and Jack. I wanted total absolute control over engineering: that was it. There wasn't any negotiation. ~~there,~~ There wasn't any room to move. It was very simple. I pretty much knew what was going to happen when I left. The company would run fine for awhile and then Ken would screw it up – perhaps beyond repair!

¶ As an engineer there's only one way you can know something, you have to construct an experiment to prove it. Leaving, as far as I'm concerned, was the beginning of an experiment. It took a little while to execute. I wanted to be wrong. It was an experiment I wanted to fail, but in my gut I knew what was going to happen. Intuitively I knew the characters involved.

MAN: As the product lines developed, how much did you work with the cross-functional groups?

GB: ~~We're~~ DEC was an engineering environment, strongly entrepreneurial in the 1960s. People driving to build projects. Memory testers, special systems. Ed DeCastro was in the special systems group initially. Somebody would say, "I want a controller to do that," and they'd go off and build these various projects. We were all together up in Ottawa for the [PDP_-5] controller. On our way back we said, "That's going to be a computer. You go off and build that." That's when I went off and did the -6. Kotok and I did an architecture for PDP-5, and said, "~~Here~~ Ed, go build it like that. Here's what we've learned about the PDP-4 that goes into the -5. The historical approach was that.

The engineering committee, d~~uring a lot of this~~ over a long period of time ~~I think,~~ played an important role, as a bunch of different things. Initially Ken communicated with engineers, ~~about things~~, we all communicated about project status. It was project status that was dealt with [by the engineering committee.] It was a place for a consensus about what something was going to be, or ~~what was~~ how to solve a large problem. It be interesting to get the engineering committee's minutes.

I think if

you look; Ed DeCastro I would guess wrote one of the first~~, as I recall,~~ engineering standards. We started embedding engineering standards. The combined knowledge of how you do engineering, (the process) and what the product constraints are. Half the companies don't do that. It's really important to have those down in one place where you can look at it and somebody can say, here are the standards that we've agreed to; E.g Environmental standards. Things that could work with other things, because as you built bigger systems you ha~~d to ha~~ve everything running in the same environment. Th Stet Things like that were very important ~~about~~. The engineering committee was the place where engineers from every group ~~[meeting~~ met together]. [You should ask] Allen Kent what his historical approach to engineering is; Allen's a key guy. Tom Hastings is too. Kent maintained a lot of that. Kent and Hastings were both very good scribes, and ~~are~~ keeping order in all of this. In my view, a lot about me versus a lot of this is knowing precisely what everybody can do, and how they operate and what their ~~infinite~~ good or ~~infinite~~ bad portrays. ~~in everybod~~y. I characterize people in many dimensions. I look at them under a microscope. For example when we did VAX, I hired Hasting to do programming; he maintained the VAX architectural strategy. Strecker is very good at that, very clean, very precise writing. Kent was like that in the physical area. I look for people like that at various times. You need ~~that~~ them on every project, whether it's an editor or what. Know what you've got. I'd say it is an intuitive feeling of knowing what you need to have to make a project succeed. My book shows what I mean.

[GOING THROUGH BOOK] This is something I call a technology balance sheet, and this is what I look for in a company. When I go into an engineering organization I look at 12 dimensions. This is a refined view over time. This is a part of a whole theory in the book because I look at a company and I say that in a start-up there are 12 dimensions, and I look in engineering and there are 12 dimensions ~~there~~ that are important. You've got a bunch of people here, the people dimension. I ask who's running the project, how do they work as a team and all of that. Then, whose vision is the product? Does it reside in one head, or can you show me the structure for the integrity of this project? Then I look for other things like, here's your process. What standards are you adhering to? What are your internal standards? What is your technology -- what set of skills do you have that you uniquely can execute that nobody would allow you to do things? So this is being very explicit about what I feel, how I think I've operated intuitively. I have this intuitively, and I just forced myself to put it down [on paper]. This book, in a sense, is a highly structured [view] of how you start a company. Here are 12 dimensions, this is how these dimensions should grow over time. Here are the states that the company should go through, very machinistic. I believe you can start a company just like you can write a new piece of software, with the same reliability. You do all the things by the numbers. That's absolutely totally non-intuitive. I spend my life being either one way, or being in both of those camps.

MAN: Were these criteria applied to the 11 and the VAX projects?....

GB: I think a lot of them were there. I think all of these had the character that everytime you engineer something, you're always stretching to do something that's better than the ~~next~~ previous one. So in a sense the project is always going to be in some kind of trouble because you're always stretching. When I consult I'm very careful not to interact at a certain point, because I know I can make the project undoable. [LAUGHTER] There's some stuff going on at Intel now. I said, "OK, go run the test chips. I will not talk to you about the project until later...Let's see what the tests show." I risk not getting the tests. We called this "ntl"-ing a project.

There were strong product lines, and there was a lot of pressure on product lines. The -8 product line was there. The 18-bit product line and the 36-bit product line. Then there were these splinter application product lines, the Edu product line and others. You had two fundamentally cross-purposes product lines. You had the product product lines, and you had the application product lines. By the way, the '66 to '72 era was a hard era to get another machine formed. You had all the resources all tied up. Who's going to do something else? It's the classic problem of "Shit, how are you going to do something ~~else~~ new? The 8-product line is going to make the next 8."

¶ Out of that came the proposal for the 16-bit machine. That was the famous X. The whole X story would be an interesting story, ~~in there~~. I've probably got some notes somewhere when I talked with Henry Burckhart a few years back about the X. I'm not sure Henry told me everything. Henry implied that if ~~we~~ DEC had made the X they would have probably stayed. Ken ~~almost~~ basically forced the formation of DG. Couldn't have done it better. Here it is, [READING A MEMO ALOUD] "Re the DG Formation: As I said before, it would be unclear why the folks left to form DG. At Carnegie, Ed DeCastro and Henry visited me to discuss the new X. I blessed the X and wanted to see it built but the group did a number of things to get it rejected, i.e., telling everyone that it was a more difficult project than the PDP-6, which is the last thing that everybody wanted to hear. The X group did experiments with large boards which Ken was also against, having just switched to even smaller boards for a wire wrap machine. This further alienated Ken et al. I don't know what the group was doing at the same time as the X vis a vis raising money or thinking about DG. I'm sure they weren't designing the machine, as the Nova looked nothing like the X. Ken and Ed DeCastro aren't great communicators, and Ed was moved organizationally into an untenable spot. Having been responsible for making the most money for the company in the 8 line, the X group was paid the ultimate insult. Ed was put to work under John Jones who worked for Stan, both of whom he had no respect for. John was a bright MBA student of General Doriot with a physics BS, who made a market selling PDP-4, 7, 9, etc. to physics for pulse height analyzers..." That's kind of how I think it happened. Whether the X would or wouldn't have happened. But once it did, once the DG was out there, then DCM had to form to compete with it. That was an aborted horrible effort under a guy by the name of John Cohen who didn't know anything about computers. He had a little team doing it. ~~that.~~ Finally it became such a disaster, that Roger [Cady] who was handling the PDP-8 at that point, came over to handle the PDP-11 project.

Then the PDP-11 formed out of that, just in time to start to stop DG. The -11 still wasn't quite good enough to stop DG. Things like the LSI-11 were critical to doing it, and then the 11/45 was critical to doing it, and then when the VAX hit, ~~well,~~ it just ~~totally~~ really stopped them dead.

MAN: Did the -11's emerge to then consolidate and pull together the scatteredness?

GB: Yes, there is a memo that I have somewhere. The only regret I have by the way is not keeping every memo that I had ever received from Ken. You get a feeling of his many faces. I do have a file with about that many of them. Have you read a lot of them?

INT: We have seen some of his correspondence from the early days up through I would say, late '60s, early '70s.

GB: OK. I had proposed the formation of central engineering, ~~and I think it was~~ in Feb. '74, that's when it was all pulled together. At a Woods Meeting in Bermuda of the Operations Committee. It rained there and it was cheap + easy to get to

JP: And that was to consolidate resources?

GB: Because I came ~~in~~ back from Carnegie in June 1972 as a staff guy. I had memory engineering and power supplies, two of my favorite things. I was essentially VP of engineering but yet I had no resources. So I played staff guy pulling together, looking at things in various ways, proposing different things. Then in February of '74 I basically proposed that everybody report to me. In that sense, I

This fake entrepreneurism produced almost 10, incompatible Operating Systems.

think, some of the entrepreneurism got lost, but it was fake entrepreneurism. Building another machine, that's not very entrepreneurial, or doing another version of the next release of Fortran. I look at follow-on the next one of the series given the technology shift, that's not particularly entrepreneurial.

Doing really different things is... I mean deciding that you need a terminal, or printer for example. Phil was creator ~~Lout~~ Laut. [sp?] and I proposed that we get into the terminals and graphics business in the early '70s. Phil wandering through numbers. He~~'d have~~ used. archaeological filing, piles of files. He was doing our numbers, he was my staff guy. The line guys were just rip-shit at him all the time, because he didn't care that much about the goddam numbers, about how they were running. Basically as a numbers guy he was very good. He got into rebirthing after he left DEC, has written five or six books. He and I got along very well, because we were both intuitive. "We think there's a market for this based on some of the numbers." "Let's get in on the terminal business." ~~He did a lot of things that purely were from looking at stuff.~~ He had a good sense of working from all kinds of things. When Central Engineering got going then you needed enormous methods. ~~Shit,~~ We measured every goddamn thing. You could say I'm very intuitive on one side, but now, go look at the engineering plans. How much stuff do you see today from DEC of this kind of stuff? I ran the whole damn place on these curves — or rather the managers ran them by the curves!

I ran engineering relentlessly. ~~This wa~~ For example s why Ken and I had ~~brutal~~ strong arguments about the issue of whether or not people were allowed to use semi-log graphs. And I ran the whole engineering department on semi-log stuff. What's the state of the art? Give me that point! I don't think any of it runs now that way, or has run that way for a long time. Certainly as measured by the output of the products, there's not much attention to ~~that stuff.~~ goodness

MAN: How who protected were engineers from the influence of marketing and manufacturing?

GB: When the organization went from I'll say more functional, when central engineering started, then the trick was to make a good coupling between engineering and marketing. We had all kinds of mechanisms for doing that. The marketing committee ~~is~~ was a place where the projects got proposed, to a series of what we call ed pots "POTS" -- different places where strategy was determined for various price bands, market uses, and things like that. ~~The interest~~ in product lines, essentially, this was the board of directors for ~~the~~ an engineering group. So every engineering group had a board of directors; ~~in fact~~ of all the marketing groups that were going to market ~~those~~ the product~~s~~. That was the basic mechanism, or one of the mechanisms, that was used for determining strategy. ¶ We earliest days, a lot of engineers interfaced with the customers. Gradually that got ~~sort o~~f buffered away. Then you clearly had to have better metrics ~~these numbe~~rs to measure where they were going.

~~Then~~ The product line groups had engineering. When I was there during the '72 to '83 period, the product lines did special hardware, special software. Lots of conflict there, because Ken very often wanted me to be responsible for all of these things. I had had enough of experience trying to do it, that I knew that I couldn't make it work. "Gordon, are you doing all these dumb terminals?" There would be a list of 17 different terminals. I looked at them and I characterized them, and every product group had its own terminal. Three or four guys, a little team off doing this shit. Some of them worked and some of them didn't work. I'd go on and try to influence them, and they'd say, "Oh no, you don't understand the market. It's got to cost this much and it's got to be like that. So bug off." So after a while he said, "I'm not going to spend any time on that shit, because there's just no reward for it." There was no way to deal with it. We just didn't have a good way of working it. A lot of the product line engineering stuff was very ill-conceived; some of it was an attempt to get crazy margins and do product differentiation when there was no differentiation needed. What they should have been doing all of the time -- it's clear in retrospect -- was the software, to go into the various markets to do unique things. That was the kind of character of engineering. Some products were important. The MINC was an important thing to get out, then the industrial PDP-11s were important. Masscomp formed ~~during that whole process~~ There were, a bunch of good guys from DEC Lab group. They self-destructed but the product was very good. They should have been ~~all right.~~ great.

Under the product line managers ~~they~~ we had ~~this~~ a dichotomy. I can look now at how bad VAX was for the company; it took a lot of entrepreneurial ~~stuff~~ energy and made everybody sell VAXes. It streamlined the ~~damn thing~~ organization, but that gave them an enormous number of resources, and they just blew it!. They didn't make investments. I can make a very strong case that VAX was very bad for the company, the winner / to run the company! and the loser. It took no brains at all. I look at the era of VAXes when the organization atrophied, went to sleep, and just got enormously bureaucratic and nothing came out. The product lines should have been out there groveling for software, ~~getting things, making~~ building deep applications, making deals, and burying itself deeply into customer relationships. A year ago, believe it or not, I tried to help sell a VAX to Cirrus Logic, which I'm on the board of. I said, "~~Jesus,~~ Cirrus, for God's sakes don't buy an IBM AS 400. That's the last refuge of the idiots. That's the worst possible machine that you could think of. Here we are selling PC components ~~stuff~~. ~~What~~ I would ~~do is I would~~ bite the bullet and ~~I would~~ make my whole ~~goddamn~~ company run on PC's. It takes ~~a~~ lot of guts, but it can be done." PC servers, etc ~~stuff~~. Believe me, all that stuff ~~shit~~ is available. You can get it. It takes ~~a lot~~ of balls to do that, but I personally would ~~do it~~. Then they started dealing with IBM, and IBM scared the shit out of them. So they went with IBM. ~~DEC made a good proposal.~~ DEC made an OK ~~good~~ proposal, of a VAX VMS running ~~doing~~ ASKs ~~stuff~~, and a UNIX ~~then another~~ machine that was going to do ORACLE, ~~stuff.~~ So they had three packages running on two computers and a network. And I said, "That's OK. They've got a lot of computational power there. You want to go distributed anyway. This forces you there." Actually the killer in that sense, was a thing that I would have had ~~a little bit of~~ trouble with too, which is you're running PC's, you're running UNIX on SUNs, you're running UNIX on DEC, and you're running VMS. That would have brought in ~~two~~ or four different systems. IBM says, "Oh just run this little rinky-dink AS 400, and ~~here's~~ the three applications ~~that are going to run there~~ into your PC network." That's what they did. Now the terrible negative thing that that taught me is that DEC had a lot of power in the various semiconductor companies around Silicon Valley. But what it made clear to me was that what DEC should have been doing was putting in a very strong field marketing organization that knew semiconductor companies, dealing with the accounts, the floor, the line, the inventory. They should have buried themselves into a bunch of markets or not have been in there at all. They didn't. They had kind of a weak sales organization and the salesman was out there scraping up software off the walls, ~~and~~ bathroom floors and every~~thing~~ where else and trying to sell it. And IBM just beat them on that. So DEC didn't ~~go~~ get those applications from third parties, make the deals vertically into these organizations and knock off. ¶ My model ~~of the world~~ is around SIC ~~sick~~ codes (?) and applications. There's a very strong model of how you do market segmentation in ~~here. [POINTING TO THE BOOK]~~ the new book. They just didn't do their home work. ~~I would have done all that~~.

¶ If I hadn't have left, I don't think DEC ~~it~~ would have atrophied. I can look at all the things that that screwed up and say these wouldn't have happened. I can guarantee you DEC wouldn't be firing people. ~~If I were alive. I mean~~ It would have probably killed me to stay though. The idiocy of DEC today is so stupendous that I couldn't have let it ~~do that.~~ exist.

MAN: Open systems: What did it used to mean, and to what extent were the seeds of open systems ideas sown back in the '60s and '70s style at Digital?

GB: ~~The issue~~ of Open systems~~, I think they~~ came out of UNIX. Then the fact that UNIX was then out there, and then people started porting ~~that~~ it to different platforms and t~~hen~~ people said, "~~Oh shit,~~ I don't have to write my own operating system, I have kind of a standard, ~~here~~." When we formed Encore, it was way ahead of its time, but it conformed because of UNIX, we can go out ~~there~~ and that's what the world wants. They want the standards. They want to be able to port software from place to place. I'm not sure ~~that~~ it was Open-standards-equals-UNIX at that point, ~~but it's hard to... how~~ I define open systems ~~is~~ at a given level: ~~can~~ you can buy ~~a particular~~ system or component from more than one source. At the PC level, ~~yes,~~ I can buy it from multiple sources. I can buy a platform and hardware. In fact, I divide the world up into this [DRAWS], and then I can show ~~you~~ the dilemma of DEC of the 1990s. You've got platforms ~~here~~, environments ~~her~~e, and applications ~~here~~. So you have to have ~~all~~ three ~~of those~~ levels. ~~Now~~ Here's the DEC dilemma today. ~~You've got environments here, you've got the VMS environment. You've got the VAX here, going to, from what I read in Fortune, is a thing called Alpha. You've got ULTRIX that sits here as a platform. MS DOS, you've got Windows, and then you've got NT as an environment there. Then you've got SCO UNIX, then some form of OSF, I don't know what the hell it is. Whether this and this are related, I can't tell you. I really don't spend much time looking at that. Then you've got MPS 32. That's going to transition to a 64-bit architecture.~~ You've got the

Fundamentally, there are a total of 6 - 7 differt applications platforms, all of which will change from 32 to 64 bits. No way, can the sales dept assimilate these or customers understand them.

~~INTEL architecture. You've got X86 transitioning to a 64 bit architecture that will be a different platform. So you've got 6 potentially different platforms. Now let's talk about applications. The main line is you've got a set of applications that run on VAX and VMS. You've got a set of applications that run on VAX and ULTRIX. You've got a set of applications that run on MPS 32 and ULTRIX. These are different applications in a binary sense. You've got the X 86 running on these two platforms and I think these are both available from DEC. You want to run UNIX or DOS on a PC, you can buy both of those from DEC. Now I've got MPS 32. They're going to do something and they're going to run on that also, and then they're going to run over here on this platform. So you've got A4, A5, A6, A7, sets of application platforms. So I say who could possibly sell anything out of this grab bag?? No one can understand this stuff! Then they sell Apple over here too, I think. But this is the dilemma. These are all different, unique. Here you've got this problem.~~

MAN: Well, this is a hell of a strategy, Gordon, but wasn't that the same case in PDP-11 land?

GB: It was ~~That's~~ nearly ~~exactly~~ the same thing, that's why I said I am going to run engineering!! ~~Precisely, you're right!~~ DOS, ~~you had~~ RT-11, ~~you had~~ RSTS, ~~you had~~ RSX-11A, M and ~~then you had~~ D. we ~~You~~ had a lot of environments ~~here~~, and they were ~~absolutely~~ all different.

MAN: Not to mention the 10/20's.

GB: ~~The thing there was~~ At least ~~you~~ they were separated in pricing. The 10 and 20 were over 500K and the~~se~~ 11s were 50K to 300K, ~~or something like that~~. Some of these were actually price segments. ~~In~~ RT-11~~, that~~ was a single-user system. But look at this, there's a little segmentation here. This is bigger systems, this is desk-top stuff, and you could say these are PC's, workstations, mainframes down to workstations and stuff like that. ~~So you've got something there~~. But that was at a time when you had a strong product line structure where these guys had figured out how to use them. They were even using the operating systems to segment their markets. So you had people who knew what was going on there. Who at DEC knows anything about any of those environments other than the VMS environment?

MAN: I remember a quote I heard you say that one of the dumbest things that Digital ever did was not to separate the VAX from VMS.

GB: Because we would have been able to make the whole thing portable, and then ease this transition. You could in principle then write the portable software. The fact is these are really different environments. That's the problem. These are different machines. Just as these were similar machines, but finally we had emulators under M th~~at you could~~ to run RT-11 programs ~~under there~~. But ~~these are~~ absolutely all differe~~nt environmen~~ts. The more they evolve ~~the~~ more different they ~~become~~.

MAN: In PDP-11 land you just said let's go with 11M plus for compatability. Toss the others.

GB: Yeah. I'm probably the greatest computer genocide guy going. There's a kind of paradox. It's one of the most brutal things that you can do to an engineer. – kill their product But the worst thing th~~at you can do to~~ for an engineer ~~in a funny way~~ is to ~~have them~~ face the market and have ~~it~~ a product fail in the marketplace. The guy would say, "Oh shit, we didn't get to take it to the market, what a dumb organization." But once it goes into the market and no one buys it, ~~oh God! So~~ it's very hard. So what's the kindest thing to do, and what's the most profitable? Take your losses early. I remember two of those. One was... the 11/60 and the 11/74

JP: Bob Everett and Jay Forrester said that that was a really critical thing to recognize and admit when you've got a dog or when something's not working. Just stop.

GB: Get out. Exactly. Roseanne [Giordano] and I were talking last night and I said, "Roseanne, do you finally understand about the PDP-10, that I didn't kill it?" And she says, "Yeah, I think I understand that." It's well-known that I'm the guy who did the PDP-10 in. And she said, "No, you really didn't. It was the group, they did themselves in." You could say, well, it was bad management. You didn't select the right people. That's not the way DEC engineering worked. Goddamn it, I have never "put" anybody on any project. Engineers selected what projects they ~~go~~ worked on. You can use coercion, you can do everything, break their arm and knees, but you can't force them to work on something they don't want to. That's my philosophy, and that's the way DEC engineering always ran. I don't believe in interchangeable engineers. Engineers are solely motivated by their drive or willingness to want to work and accomplish something on a project. If you fall out of that for any reason, namely somebody's made to work on something, or assigned, ~~I think those things~~ all get in the way of good engineering. When PDP-10 got cancelled, it was an absolute vote of no confidence, ~~from~~ on my part, I was confident that those guys wouldn't be able to produce a computer. They had tried for a number of years. They had no management that was skilled in ~~their ability to~~ design ~~things~~, and they had no designer or architect. On my balance sheet, they came up zero. They had all the DEC processes and stuff like that, but they had no skills to speak of. They had a good circuit guy. They had no logic. They had no architect and they had no management. So what do you do? ~~Boom,~~ You shoot them. Get rid of them.

JP: That begs this other question which is one of yours. Is there an engineering ethic?

GB: Yes — it's commitment to the project and team! Also, it's the responsibility for the product's efficacy ~~I'm sure the engineering ethic changed... at one level which was~~ I think ~~the strongest~~ this ethic ~~which I was happy to see~~ has transcended DEC. It's on the west coast to a lesser degree. ~~There is a massive engineering~~ this ethic ~~overall that~~ I always thought mostly existed only within Digital. ~~And that is~~ It's you make a contract when you start a project and that contract is that you're going to finish that project and make that product work. If you sign onto a project, you finish the ~~damn thing~~ product and make it work. That's basically the agreement. I didn't let people transfer out of projects. So to me that's an ethic. There's an ethic of what's important. You make good products. You don't leave the project until it's done, or until it's stopped for some reason. There was an ethic that said they had a responsibility as to the efficacy of the product.

When I went out to Stardent, Allen [Michels] introduced me to somebody and said, "Here's Gordon. My first encounter with Gordon is when he threw me out of his office." That was probably right. I didn't recall it, but anyway. At various times Allen and I got into arguments, and I said, "Allen, I hold the engineers responsible for the efficacy of the product. Not the marketing guy. No one's going to tell an engineer to make a product and the product fails and the engineer said, 'Well, look, somebody told me to make this thing.'" That was the ~~There's an~~ ethic ~~there~~ ~~at one time~~. I hold ~~them~~ engineers responsible for the product's efficacy, and to stay with the product until it was done. I & ~~[illegible]~~ managed to

MAN: What happens when it was too hard?

GB: When it was too hard, management had a responsibility to continue support or the group had a responsibility to decide that it was too hard and that you should back off on it. Shut down or redirect, or to come forward as rapidly as possible with the knowledge that in fact the project should be changed.

The 11/60 is one that I tried to get killed. The guys were furious with me; they said, "Why are you trying to kill our project? You helped start it." I said, "Yeah, but you guys are two years late! We don't need ~~the thing~~ it. It's just going to cost the company money to do it. It's not an interesting product. We shouldn't take it to the market." The product management were coupled strongly enough into the product lines that they said, "No, we're going to sell a shitload of them." They signed up. And at marketing committee I used to poll, "Are you guys really sure what you're signing up for?". "Oh yeah, my guys tell me we're going to sell those." I'd say, "I don't think you are." That's one that I was overridden on.

On the 11/74, which was a multiprocessor 11, a really nice product, we had 100 on the line. Every month I used to come to Marketing Committee ~~in~~ and say to Julius Marcus, "Juli, are you going to sell those? It looks to me like VAX is taking off. I don't think we need to introduce the ~~that~~ product even though I love it. Shouldn't we stop it?" One time he said, "Yeah, you're right..." Juli was always on the fence and one day he fell off! But basically I feel very strongly that you only introduce good products that will make money. In this case it was simply that VAX wiped it out.

MAN: What about the role of creativity, innovation? Once the strategic decisions were made how important was the detailed engineering communications? Did Digital do very well at that?

GB: The quality of the engineering in a specific group ~~really ...All the groups themselves~~ were totally self-calibrated. I understood exactly who could engineer products and who couldn't. Who would you trust to build something? As far as I was concerned it was all execution. Some people sort of vomited code and logic on a piece of paper and some people did it elegantly. ~~In a sense~~ I and the key engineers all knew who those people were. ~~I~~ We could tell

~~This~~

vomiters from artists. Sad to say, the poorer engineering managers couldn't.

MAN: What were some of the most elegant engineering...?

GB: ~~We~~ DEC had a lot of guys that were really very good, very creative, wonderful engineers. It had more than its share. ~~I'll tell you a tragic story about who isn't engineers and what just happened.~~ [Dave] Cutler, [Roger] Heinen, people who could really take a large problem, decompose it, and structure ~~it~~ and build an absolutely elegant ~~thing~~ system are tragic losses. ~~Who~~ These guys knew every detail about a product and how to build it. These were the kind of engineers that I related to. It was just like ~~this crap~~ the ~~PDP~~ engineering I did, described in this notebook. I basically knew everything about ~~that machine~~ the PDP-6, about ~~those machines and the~~ the architecture and implementation in both hardware + software. ~~At a time~~ In the mid 60s it was e~~asier~~ easy to do that. It's harder now, but you deal with it in a different way. But Cutler was like that. [Mike] Riggle was like that. Riggle to me was probably one of the broadest engineers I know because his knowledge included the magnetics, and dynamics for disks plus error ~~detection~~ detection/correction ~~we had. Very good~~.

[END SIDE 3 -- BEGIN SIDE 4]

GB: Fred Hertrich, ~~[disk engineer]. He~~ who built key & storage products who lives in Colorado, was a ~~W~~wonderful, broad German engineer. He ~~Had~~ had five guys working for him.

The guys who really knew engineering were people whose team never got more than 20 or 30 people, a small team, ~~and then~~ were ~~just~~ focused and r~~olled~~

IN

and ~~got the thing out~~ produced the really profitable winners. ~~Everybody in there was just really good~~. ¶ The problem with Marlboro is it got out of control, and never got the right culture after [Alan] Kotok left. It just kept getting worse and worse until it sort of burned up into the sun. ~~I suspect~~. ~~I mean~~ The 9000 was done there and a financial disaster for DEC. It got enormously bureaucratic and out of control. ¶ [You ask] who are my engineering heroes?

VM

Foresight, Bob Supnik, ~~in the VLSI area.~~ and Dick Sites in the VLSI. However, let me not go on the record here because I don't want to slight the great engineers and engineering managers.

¶ To be the ~~paragon of engineering was~~ penultimate, were engineers who did a piece of the work themselves, and managed the ~~engineering~~ project. ~~and did engineering, too.~~ They always knew what was going on. Cutler did VMS that way, and then he did the PL1 compiler that was then the back-end for ~~I think now probably all~~ many of the compilers. Cutler ~~probably more than anyone~~ is probably the biggest loss to the corporation for ~~two~~ three reasons: his work, the projects (alpha forerunner) ~~ts~~ that Ken killed and had to restart, and He's just got ~~the system~~ NT running that will drive another nail in the ~~DEC~~ VMS, coffin; ~~he's built a thing called~~ NT, is for New Technology operating system, and he works for Microsoft. That will end up on dozens of standard platforms to compete with DEC hardware

MAN: You manage and you do it, too.

GB: That's exactly my model. ~~of how you work with this.~~ We just talked to a guy ~~who was trying to hire a guy~~ for heading a ~~the~~ research organization. He said, "How am I going to do my research and manage this thing." We said, "Don't take management that seriously. You'll be a shitty manager. No one will respect you unless you have a piece of your own. Think of how you're going to run it in halftime. You should have your own project. Do that and get help around you to do any of the stuff that you need to do." To be a successful research manager, you must create yourself. That's also my model of engineering management. This holds right up to the top.

MAN: Projects getting more complex, you said there were more team members needed. But was that true... that was also the birth of CAD systems, and...

GB: Sure, CAD was coming in and helping a lot of that.

MAN: What was the role of that? Did that allow you to work smarter?

GB: No, it allowed you to build more complex things ~~and build them~~ more reliably.

MAN: But it didn't necessarily take more people?

GB: No, in fact, with CAD you were able to contain the time and the people and deal with the increased complexity. I'm not sure anyone in ~~the~~ upper engineering management ever figured that out, outside of ~~the guys in the~~ VLSI. I don't know what their project sizes look

Stet

like now. This really came home to me after I left DEC. The phenomenon is well ~~known~~ documented in [illegible] new book. also in the Feb'y 1989 Spectrum article by me. When we started Encore, ~~we had~~ a few people ~~that~~ came out of DEC ~~that built multi-MACs~~ to build their product, Multimax., ~~and then~~ Charle Rupp came out and built ~~his~~ a great, large screen terminal. ~~And we~~ They did ~~it~~ the projects with very, very tiny groups. The Encore group, with hardware and software got up to be, ~~I don't know,~~ 30 or 40 people and that machine is still better than anything DEC has as a multiprocessor. The 6000, ~~they had~~ has six processors. ~~Bullshit.~~ ~~I mean they had 20 and 85.~~ ~~The machine~~ Multimax with up to 20 processors is still a ~~good~~ great machine~~, a~~ and much better machine than the 6000. Better in terms of bandwidth and throughput. It was done with 40 people. At Stardent when we shipped Titan, then ~~there~~ were 45 people. We did a compiler that was better than anything DEC has, ~~got.~~ We invented a whole language for dealing with graphics which a number of companies have licensed. We built a vector MP and the architecture for that and all the software. A very complicated machine. 45 people. The secret I've learned, at least after DEC, has been that rarely should you ever have more than 45 people to ~~design~~ build a machine. KSR's engineering staff is about that size ... and they've produced a revolution.

MAN: What's the role of the individual then and what is the role of the team? Is the role of the team to promote individual inspirations?

GB: The team is a collection of individuals. It's the role of "management" ~~of the thing~~ to see that the right

Line 46

resources are there and that people are cooperating and that the ~~corporate~~ team members are productive and creative. ~~That~~ They must have the right goals and plan. That's how I see it. In a hard project, what you want is ~~you want~~ excellent manager, leader, builder, and three or four people working with them as engineers to be part of that. That's the team, and ~~that~~ it really functions. That's the model that Cutler used on things like the compiler. The start compiler is a great compiler and it was four guys. The CPU ended up with five or so. After a certain point you may need a ~~pile of~~ many compiler people doing different things, but...

JP: If we go to the good side of the VAX project for Digital, I think it was that there was a clarity of vision and there were a bunch of people who operated towards that same goal.

GB: Oh, yeah. ~~You've~~ Have you looked at the whole VAX strategy ~~stuff~~ documents? Do you have the original ~~paper on that~~ VAX strategy paper? There's two things about VAX. One is the VAX architecture itself, ~~and VAX~~ and ~~the~~ Stet VMS, the project -- the 780 -- that launched ~~the whole thing~~ VAX/VMS. But that's just another computer. The main thing, what made the VAX work was the VAX strategy. It was approved by the board in 12/78. That was the vision of the single architecture. That came out of the trip to Tahiti. This is described in Rifkin's book. I have had enough "ah-ha's", invented enough things. The method in ~~this~~ the High Tech Ventures ~~[illegible]~~ book ~~[pointing]~~ is like that; the ability to characterize a company in ~~these~~ many dimensions, plotted this way and growing accordingly, is an invention. Unfortunately I can't tell you what day I did it. I can show you it in my notebook. It was on a trip going to the west coast. I was at NSF and I was coming back. It all kind of evolved about a DEC memo that I had written years ago called, "Heuristics for Making Great Products." Do you have that memo? Russ Doane, another guy to quote, had commented on that. Of all the things, [to look at], that one. It's a "Folks, here's the secret. Here's how you do it. It's only 35 rules." my other inventions ~~fo~~ all had a similar pattern — generalization. ~~The~~ I count ~~[illegible]~~ my inventions as: a generalized flip flop, the unibus, general registers ala PDP11, and the VAX Strategy/Hierarchy. ~~The other memo~~ Somebody sent me a ~~few years ago was a memo~~ memo I had written, called "NOD." No Output Division. I had just attended some meeting where everybody was sitting around in a ~~room,~~ review meeting and this poor ~~old~~ team was trying to get ~~something done~~ a product done. Everybody's commenting, "Well, I don't think you can manufacture it." "Well, I don't think you can sell it." "Well, I don't know whether I can serve it." And so I said, "~~Shit, you know,~~ Why are we building this?" It turned out it was an important product, but all these dumb reviewers were doing was covering their ass.

But back to ~~the~~ invention ~~thing~~. There was an incident in Tahiti where it was sort of invention; this is ~~a~~ the Computer hierarchy. Now that had been building in my mind four or five years. I had given several papers on distributive processing. I'll show you this one. It's an innocuous paper, ~~this was~~ in '75 and it was never published any place important. Basically the idea of a three level hierarchy was in that, that is you organize machines this way, and then the need to tie those together with one structure and then be able to do all the things that VAX was going to do was there. ~~Here, in~~ This ~~there~~ was a business plan, essentially for the company. Here's the VAX strategy. That way you can have 100,000 people working on something if you can state it ~~like this~~ simply; ~~that~~ this stuff is basically called leadership. People can follow if you've got some basic plan...I mean shit, I can't go and tell anybody how to do it, ~~But you can just write it down.~~ because most people haven't enough depth or breadth ~~to~~ or creativity to understand it.

JP: It's something that everybody can get their arms around at whatever level they are in the company.

Yes, but VAX was ultimately a disaster because people gave up thought!

GB: The trick was to see what was ~~a what~~ happened to the computer industry ~~DEC is they didn't see anything happening~~. ~~They~~ DEC really didn't respond to any changes outside. It was really a tragedy. ~~Here you~~ One saw this army of changes (eg. PCs, UNIX) marching down. ~~and~~ Everything was changing about them, and they didn't see it at all. That's the craziness. It's tragic. I tried to warn ~~a little bit~~ them. Certainly if they had done any of the things that were in ~~this~~ my Computerworld article, then they could have gone marching down the line. It would have worked just fine. They wouldn't be in a whole bunch of markets, they wouldn't have that mess there, but they'd be making a lot more money now.

MAN: See, it's 5:00. Right now, Patrick and I have just been laid off from Digital after like eight years ago I began.

GB: You are?

MAN: The week you left was the week I began formally. Isn't that great? So we appreciate what you're saying. We can relate to it.

GB: It's ~~this~~ a kind of stupidity. ~~thing of their knowledge is all embedded in them.~~ Hiring and firing by spreadsheet. ~~God, it~~ used to drive me absolutely crazy. The spreadsheet mentality~~;~~ You just take the numbers and ~~you~~ run them out, and allocate percents, independent of the groups. ~~OK, use so and so or buy such and such.~~ The fact that Jack Smith has any power ~~is there~~ is absolutely ~~a crime~~ criminal ~~to humanit~~y, the ultimate ~~in~~ spreadsheet mentality. How did the company got out of control? The controls were absolutely clear. Even if they had no plan the numbers ~~were all there that~~ said they were out of control. What was the controller doing? The fact that these guys are paid anything is just a crime! They should be paying the company, because they wouldn't have anything else to do!

This reflects the fatal flaw in Ken's personality.

They are just so imcompetent that it blow~~s~~ing my mind. The myth that a Ford-trained controller is good is crazy.

MAN: I love the theme that you've constantly had in going from closeness to customer into real requirements versus isolation of that. The interesting reasons for both.

GB: DEC's problem in the future is how do you get the engineer closer to the customer or the guys building the products, or even the marketing people who can help sort that out?

MAN: It sounds like a lot of what's called systems integration nowadays was to a large extent doing custom CSS, software services in the '70s?

GB: Yeah. It's the economics of who supplies the software. Where did the software come from? Now the thing is that virtually all application software comes from deep understanding by small start-up groups. You can't do that in a large company. We used to talk about application software, and rarely can you do that in a large company. It means that you have to have chemists that are going to do it, and you can't get that mix in a large company. You can't manage it. There's just too much cultural diversity. Also its almost impossible to mix hardware and software sales. Sun's done the right thing with SunSoft

JP: What in your mind were the periods of engineering greatness at Digital? Which can you identify?

GB: I think the VAX strategy... Last night at the [Computer] Museum, ~~there was~~ I met a guy ~~who was~~ doing planning in Boca [illegible] just before the PC. He said, "God, when you guys came out with VAX ~~and that~~ with one architecture, we knew we were dead. We didn't know what to do." Then when I was at NSF, I met an IBM guy who came and said "What should we do against that?" And I said, "I've thought a lot about that. I wrote all the things I would do if I were IBM, and that's in the VAX strategy document. Too bad you don't have that! Of course, it was designed to drive you guys bullshit. I'm glad it did!!" I got a kick out of that.

¶ The other periods were: 60-66; 66-70's —forming computers the Product Lines,

JP: What happened with the PC's?

GB: The PC's were an unmitigated disaster.

JP: Was that engineering entrepreneurialism at work?

GB: There was a lot ~~of that, absolute~~ entrepreneurialism or perhaps it might better be called court politics. Everyone focused on trying to please Ken. ~~at work there. Absolutely, entrepreneur. And it was everything supporting it. The ultimate.~~ What do I think about that. ~~It was one of those that~~ Ken was the sole manager or King of the PCs. I want to lay the PC ~~strategy~~ disaster totally at his feet. He really drove the PC ~~strategy~~ strat, and the development of it. this included getting Stan to open the stores, telling Andy how to run PCs. and ultimately forcing Andy out when the poor products didn't sell.

MAN: What was the role of central engineering then? Didn't you say anything?

On the other hand, I was working on VAX and ~~that~~ the strategy including VAX clusters

GB: I tried. Every six months he came to me and he said, "I want you to manage all of it." ~~[TAPE CUTS OUT]~~ Everyone should be report to me. Already I had about 8 direct reports. I would get about five more. Something like that. I remember writing a memo and saying, "These guys ain't going to work for me." I think at the time it was Clayton who running it; I said that Dick was the best guy to run it, and we should help him run it. He said, "No, the terminals person, I want Avram reporting to you. John Clarke, all these guys reporting to you. If you run it you'll make it work." I said I couldn't, I was just totally overwhelmed with ~~it~~. the difficulty of satisfying all the constraints — especially the organizational ones and Ken's biases such as pushing the PDP-8 as a PC.

MAN: But there was some strategic link missing. When Avram Miller came to you and said, "We're going to sell 3000 of these the first year." When he got all of the good PDP-11 engineers under him, when they decided to re-engineer everything again, what was going on?

GB: It turns out that he had a very famous box designer, Ken right by his side that would design boxes for him. ~~He~~ Avram put a whole new bus structure ~~on that thing~~ in the PRO. The software, was new and it was just a disaster. I had meeting after meeting on the software to try to get that into some shape. The bus was a proprietary bus. By the way, at the same time, I was trying to get the PDP-11 licensed as a chip. I said, PDP-11 ~~this thing~~ ain't going to go nowhere. That whole thing was clear. I was voted down everytime on the chip license. The PDP-11 ain't worth anything. It isn't worth anything as proprietary architecture. Get it out there. If we do it right we might be able to stop INTEL and Motorola. But we had no one over on the marketing side. Ken was busily shooting marketing people, opening stores. We spent all of our time arguing about the PRO versus the Rainbow versus DECMate. And Ken wanted them all to win, saying DECMate is the right way. We're going to put application software on it. And I said, "Ken, the machine is braindead. It's run out of its feeble old memory. You can't write programs for it." In a sense all of them had that problem, although the -11 was me we could work with.

There was the original billion dollar mistake, which was making all three PC's and Avram and his projections and all that crap. That was in conjunction with the marketing guys. Meanwhile Ken is flogging the ~~shit out of~~ the marketing guys to sell what Avram wanted made ~~do stuff~~. Couldn't make any decisions about how this thing is going to be distributed. So if there were minutes of the Operations Committee, [they] would reveal that in fact all the time was spent working on that. We had the marketing committee. The Operations Committee was spent arguing about how important these PC's were going to be, and all the product lines wanting a piece of the action, not wanting to give it up. Then to say, Andy you take it and figure out how to make it work. It needed to be divisionalized. The company was arguing about it. Then they'd use the machines to pit each other against... And Ken wanted the PDP-8 to win, and we had this marginal word processor. I said "Ken, that's a terminal. Think of it that way. Get the cost down. Don't try to make it do Dibol, and do all this other crap. It ain't a computer for business." Ken's lack of understanding of software and where it was and would come from is probably the single most important cause of the failure in that thing.

MAN: But Barry James Folsom's vision was kind of close.

GB: Barry's was pretty much right on. After I left [Ed] Fredkin went out to DEC and I think in fact, I even went out to DEC, had breakfast with the Operations Committee and said, "The PC wars are over. It's clear what you do. Absolutely clear as hell." ~~All I can explain was that~~ The billion dollar mistake was one thing. That was an organizational and product mistake, ~~and~~ when the whole market was trying to decide. We lost. In retrospect we may not have had a chance there, given what IBM did. So you could say look, no matter what, we would have lost that particular battle. But once the PC had formed it was absolutely clear in '83, '84, exactly. History was written. It was so clear. The tragedy is that DEC didn't see it at all. Even HP saw it and prospered by it!

MAN: And ironically the Rainbow was a much better engineered product than the IBM PC.

GB: No. Not really. You can't argue that one way or the other. The important thing was that it was incompatible. The bus standards had been set, the BIOS had been set, and that's it, you just go do it. Don't think. Just execute. It was a good strategy, but it was an ASCII terminal. A PC at that point was a commodity t~~hat kind of thing~~. It was something you sell by commodities. DEC had really good low end engineering, in Clark's group at Taiwan, and in terminals and the ability to produce things low-cost in all kinds of places. All DEC had to do was to do that. Talk about what happened at DEC, why it went braindead during the VAX era, the PC... forget the VAX... I'll share in the billion dollar mistake. I'll take $300 million of that error because I was working on on the VAX. Ken gets $300 and the marketing guys get the other $300 million...

~~[END OF SIDE 4 -- BEGIN SIDE 5]~~

GB: It was being built as a chip at that point. This was before the MicroVAX. You'll have to look at the date. But the absolute faux pas was not adopting the PC. There is no excuse for that. Zero excuse for that. We had mismanagement everywhere in the organization. I just can't fathom what went through anyone's collective minds. I know Fredkin went out there and begged DEC to do it. Because he saw that that was so critical. I saw it was absolutely critical. It was an absolutely no-brainer. I would say that was the single most important error that DEC made in what it did. The 9000 was probably the second most important failure. Just wrong technology.

MAN: Was the Trilogy technology that poor?

GB: They used some technology from Trilogy, but they did a lot themselves. It was too little, too late, and it was the wrong technology. At that point it should have been CMOS, or it might have been saved if it had been executed on time. I don't know. That was a major faux pas.¶ If I put down what was really at the root of DEC's demise, it was the destruction of the product lines. Do you want this in here, or not? It's what's happened to DEC from an outside perspective.

JP: One of the interesting things that I'd like to get which I don't think we've really covered is the engineering environment in a general sense at Digital.

You've had some opportunity to compare it to other environments. In the early days there was a lot of creativity. There was a lot of entrepreneurial spirit. To what extent did Ken contribute to that? Did he come in real close to engineering at times and then step back? Was he always on you guys?

GB: He was. I'll say that three-quarters of the engineers that Ken got intimately involved with were ~~massively~~ turned off by him. He was not a good guy to engineer with. He would come ~~on~~ in, and ~~he would~~ overrule and play 'I'm the boss and we're going to do it my way and you guys are stupid.' There were a few people that he got along with, and that could work with him. There were people that absolutely detested him and thought he was a lousy engineer. You ought to ask people, and also ask poeple how they felt about me in terms of the engineering environment I provided, the kinds of things that I did. I too liked to get involved in ~~engineering~~ every projects. I liked to ~~get involved in the engineering~~ understand, review and help all the of areas. ask Riggle, Strecker, Stewart, Lary, Grant, Cutler — guys I worked with on projects.

MAN: How would you describe the engineering environment that you think that you set up?

GB: OK. Basically the environment that I tried to create while I was there may have come from, I won't say a university environment -- because university environments very often can be very, very uptight and closed environments -- but the goal was ~~an~~ absolutely open, free exchange of information [place] no hiding, anybody can look at what anybody else is doing. I used to say that the place leaked information like a sieve. We didn't try to control information.. especially from group to group. Everyone was free to criticize/review.

JP: Even with customers. DECUS was a perfect example, an opportunity to get software written and utilities written for computers.

GB: Our goal was very clear with DEC. Why did we form DECUS? We've got to get a bunch of software for the PDP-1 and we've got to share or we're not going to get it! In fact, it was built exactly on ~~the~~ IBM Share model. Why are we here? We're here to share software.

JP: Was DECUS seen as a valuable --

GB: Absolutely, we started it that way. The other thing about DEC engineering was that it was created as an open non-competitive [environment], lots of information, free information exchange. That's what the engineering committee was for. How does that compare with other organizations? In general, I think they vary. But I was informal. I tended ~~never~~ to not be concerned with dress, ~~up and~~ or wear ties, ~~and all that~~ shit. I ~~actually~~ liked the environment we had. I hope others did too.

or be stuffy. ~~The~~ My uniform was "turtlenecks in winter, T-shirts in summer, wear suits if I had to talk to ~~a~~ stuffy customers."
I always throw away the vest when I buy (at Filene's Basement) a 3-piece suit.

I know that people who didn't like it told me so — and we tried to change it.

~~All~~ They companies I've been ~~in~~ involved with have ~~been exactly~~ tended to be that way. Have been very casual about when you come in, no time clocks, none of this stuff. But there are engineering departments that are not like that. They have hierarchies, call people Mr/Mrs., wear suits, and leave at 5:00.

JP: IBM is certainly different.

GB: I think it's relatively more structured. At Ardent, we were trying to sell some stuff to IBM, and I was really very unimpressed with the PC engineering ~~there.~~. IBM does have some great groups though

JP: When you first joined Digital, what attracted you?

GB: ~~The~~ Small size. Size and responsibility. I could go and design a computer myself and write software myself and get t~~hem to build it.~~ it built.

JP: You were 25, 26 years old too.

GB: Exactly. ~~Shit, thi~~ It s is great. I can go and design a computer. I had worked as a co-op student at GE. In fact I had decided kind of not to be an engineer after working as a co-op student at GE. I was a Fulbright scholar, came back, started down the Ph.D. route at MIT, and I thought all engineering was like GE. I was building a tape unit for TX-0 and I [met] Ben Gurley. Ben Gurley was an absolutely wonderful engineer, and wonderful man~~, wonderful~~

engineer. I really liked the people: Ken, Stan, Harlan Anderson, Dick Best ~~I absolutely liked the people~~. The freedom and the range of all this work to do. In a sense, I didn't really want a Ph.D. I wanted to build things, and so, faced with all that and doing research, I thought, well, research is okay, it's better than working as a drone in an ~~over~~ overstaffed ~~agency~~ organization that supports itself with ~~gov~~ ~~govnt~~ government contracts.

MAN: I'm wondering how computer architects are made if not born.

GB: I don't know. I haven't really spent a lot of time thinking about how that. ~~There's a~~ A certain set of skills is critical. ~~and feelings about that~~. The undefinable characteristic, ~~I'd sort of like to say~~ taste ~~in terms of how people should do things, or what kinds of environments you want~~ is critical. Ideas about simplicity and elegance.

¶ All the patents or things I've been involved with turned out to have been taking an idea and generalizing it to the extreme. I have a patent on a thing called a multistable state device, which takes a flip-flop and makes it an n-state device. We actually used it on PDP-4 and PDP-5. Knowing when you do something, what its function is, and how to make it more functioned, and the idea of elegance, making a part do more than one function. (definition courtesy of Russ Doane) The multi-stable state device, Unibus, general registers, Ethernet. Ethernet wasn't mine per se, but I wrote the paper on it when we introduced it. I said Ethernet is the Unibus of the '80s. ~~No~~ Now Ethernet is the

UArt of the '90s. ¶ I also invented the ~~UArt~~ UART (for the ITT system in 1962). I don't know ~~how~~ how to train architects... I know some people have absolutely no affinity for it, or understanding of it or anything like that, or need for it. "We'll just go and build a bunch of stuff." Fred Brooks and I ~~totall~~y share the same views about ~~thing~~this / business s. We were on a technology advisory board for a little company, and the first thing we asked was who was responsibility for this product? Who's responsible for the vision of this product? We couldn't find anyone. We said kiss it goodbye. It'll have no integrity. You've got to have somebody who says I'm going to be responsible for seeing that thing through. By the way, my feelings about architecture, that's why you want to read ~~this~~ the b~~ook, actually~~ on High Tech Ventures. I went wild ~~here and I~~ when wrote probably the pages on architecture. It gives my feelings about them.

By the way, ~~a the key~~ being able to implement is almost essential for an architect.

MAN: You've hit on most of the themes that we've encountered by the other groups, your peers and ex-peers. I love the continuity of themes and the strange mix you personally have between discipline and intuition. Between being a technical guru with a business perspective. Between being an architect and yet a builder, a manager and a doer.

GB: I know ~~A~~ll my prejudices~~, everything here,~~ is in ~~that~~ the book, in terms of ~~this is~~ everything I know. Particularly in start-ups, when you're hiring people, I put it down in plotted enormous numbers of different things, like the revenue that a group was bringing in on one axis, and the amount that they were spending on the other, and then shifted that in time to see where we were spending money. Virtually everything translated into numbers. ~~My bias was...~~ I used to say, I think numbers people are different than I am, because they get something out of numbers, but then I concluded that they don't. They don't get anything out of the numbers. They only get obvious, dumb things. "This number is bigger than that number." Lots of insight there! You know, what's the price advantage of one versus the other, and that stuff doesn't come out of a spreadsheet of numbers. I forced a tremendous number of metrics. Stan [Pearson] drove that planning process. It just forced an enormous number of different numbers and ways of looking at product strategy and product direction. All of that was aimed at the allocation process, and convincing ourselves that the group knew what they were doing. It was self-management. Do you know what you're doing? You force poeple to analyze something from ~~a bunch of~~ all angles.

MAN: That was a good way of developing product strategy.

GB: The trick was always to get new things in. As we got larger, the~~ir~~ research group was important, and then the formation of advanced development within a pro group. Right now I'm helping

Microsoft form a research organization. So I look at what they're doing, and I say "S~~hit,~~ you guys~~, you~~ don't need a research organization, you need an advanced development function first because if you ~~put the~~ do research ~~in in this thing~~ it's going to come up with a bunch of ideas and there's no way to get the idea into the product ~~-~~ groups.

[END OF TAPE 1]

MAN: Once product strategies were developed, product managers took them over?

GB: Product managers came sometime in the '70s. Engineering was getting so complex in terms of the groups that it had to interface with -- marketing, manufacturing support, other product groups. ~~They were looking outside, they were looking in~~side. They were selling, and so the product management function took over that whole collective set of activities, dealing with product logistics. Those were very hard jobs. Then the engineers were then free to go work on the product per se. In the old days the engineers did a lot of that stuff, "Hey, I'm proposing this, here's where it sits, manufacturing is going to build it for this much..." But as those functional groups grew, the damn things got out of control in terms of communication.

JP: Besides consolidating resources, was it your goal to provide a buffer to the engineers so they can engineer?

GB: Sure. That was the first thing; in fact there was no time for engineers to ~~go~~ do these other intergroup communication things. So you had a product management function within VAX that ~~really dealt with the~~ was the buffer, so the guys doing the engineering didn't do anything but build it. Over time, I suspect that got too large so ~~then in fact the~~ engineers never saw a customer. In the allowing IBM to propagate the AS400 on the world.

8. Inability to take important, standardizing technology such as DECnet and keeping it proprietary so that DEC is forced to implement systems both proprietarily and for standards.

9. ~~Being a really poor partner to every co company it ever partners with~~. Basically ~~the~~ a corporate guiding, called the first rule, that used to exist ~~in~~: "Do what is right in every situation: employee, vendor, and customer;" ~~is g~~ just ~~plain crap~~! isn't followed. ~~Invariably when~~

When a small company comes to me with the question: how do I make a deal with DEC, I say don't — they'll ~~&~~ take forever to decide and then end up ~~s~~ screwing you. The company simply has no sense of right or wrong at the working level!

rules. I put it down in laws. The head of technology should do this, the engineer should do this. Here's the things that I think you have to be good at. You must pass these tests. And I feel very strongly about the head of engineering being able to do things. You have to be able to go in and play one position. Write a piece of code, write a spec, whatever it is. That was in the start-ups.

[END OF TAPE]

~~I~~ca
GB: Anymore to say?
GB: Yes, I can't let this decimated the interview stop without
condemning DEC's overpaid, incompetent top-levels managers who
have screwed up the products and company. They've caused 10's
of thousands of people to be hired and fired ~~with~~ because they were
not looking at fundamentals of productivity. This is trivial.
The basic screw-ups were: won't ~~a~~ overall control in the industry.
0. 1. Not doing the right thing in PCs. This was clear ~~in~~ within a
few years (certainly by 1984) that the standard was established.
2. Not exploiting the VLSI capability by using it to build multiprocessors.
This would have saved both the high end (for tp) and workstations.
This is described in my article in ComputerWorld that I hoped DEC would
read.
3. Not finishing Cutler's machine and then having to restart it as
Alpha. Then, establishing the architecture as an industry standard.
4. Going with MIPS, a thin company with a dead end architecture
5. The 9000 was stupid. Wrong technology... and then poorly
executed.
6. Destroying the Product Line Structure and the ability to acquire
and sell market focused software.
7. Not getting more dominance in the commercial space, and one