Oral History Panel on the Development and Promotion of the National Semiconductor 32000 Microprocessor

Participants:
Donald Alpert
Subhash Bal
Robert Freund
Giora Yaron

Moderated by:
Richard Sanquini

Recorded: February 26, 2008
Mountain View, California

CHM Reference number: X4430.2008

Richard Sanquini: Good afternoon. I'm Dick Sanquini from National Semiconductor. And our objective today is to highlight the history and the contribution that National Semiconductor made to the microprocessor industry, with the Series 32000. Today's discussion will be a benefit to both technologists and also to business people. Because we're going to discuss both the good and bad things and the learning's from those years when we developed the Series 32000. The Series 32000 was the world's first commercially available 32-bit general purpose microprocessor family. And a lot of the pundits that were out there at the time claimed it was best in breed. And we had some really tough competition. The competition was Intel, Zilog, Motorola, and TI. If you look at the events that occurred, due to many factors, the Series 32000 really was not a major player in the market. And we're going to get into some of the reasons it wasn't. And by the way, the only serious player really was Intel. I think what we're going to try to do is bring this era alive. And what we're going to talk about today is why we used the 32000, who were the customers, what was the market environment, how did the family stack up against competition, what challenges did we face both technically and from a business perspective, and why did the 32000 fail, what were the reasons for that. What contributions were made to the industry? I think there may be some surprises here for some of you.

Now, with us today are some of the key players that represent the 200-plus engineers and professionals that developed and marketed the Series 32000 in those hectic eighties. On my left here is Dr. Giora Yaron. He managed engineering from his perch in Tel Aviv. When he left National a few years ago, he became the president of a company called Indigo. Took that company to an IPO and that company essentially generated a market cap of over a billion dollars. Dr Yaron is a co-founder of five international start up companies. He was also the chairman of Mercury Interactive, and that company was sold to HP a few years ago. In your spare time, Giora. In your spare time, Giora is on the board of governors of the Hebrew University, and he uses his many skills that you have to help that university.

On my right is Bob Freund. Bob Freund managed the systems and software group for the 32000 during this era. And Bob was a software guru, in a company with many software gurus. And actually, you did pretty good in other engineering disciplines. Since leaving National, Bob decided to learn Japanese. And he's worked with Hitachi for a few years in many senior executive positions with Hitachi. Today he is the CEO of a company called Quadrasis, which is a fully-owned subsidiary of Hitachi. You can also find Bob on many systems and software standard industry committees. He's chairman of many and is a general gadfly in that particular space.

On my far right is Subhash Bal. Subhash Bal handled marketing for us at National Semiconductor for the 32000 at that time. Now Subhash upon leaving National got the entrepreneurial bug and founded a company called SwitchOn Networks. And after that, he basically worked with three other start-up companies, and many other large firms. Today, he is the country manager for Synopsis in India.

And on my far left is Don Alpert. And Don Alpert was our architect for the Series 32000 for a period of time. When he left National he went to Intel, the dark side. At Intel was the lead architect on the Pentium. And also, further on, he was the engineering manager for the Itanium processor. Now Don is a consultant in microprocessor technology, and the president of Camelback Computer Architecture, LLC, in Arizona. Now what I'm going to do is to pause, and we're going to start the interview.

Sanquini: Okay, let's start the discussion with Bob. How did the 32000 get started at National, Bob?

Robert Freund: Well, I mean those days, the valley was just bubbling in ideas on how to employ acres of silicon into higher function devices. And at the same time, other business ideas were popping up all over the place. There was this kind of undercurrent going on about workstations. And workstations were what was whispered about in the bars and places around the valley.

Sanquini: This was about when? 1979?

Freund: Well, yeah, it was ’79, ’80, in that neighborhood. And our vice president of technology, Pierre Lamond, always had the idea that the thing to do to help National move forward was to move it in the area of developing workstations. So they built a workstation group. And they needed an engine for this workstation. At that time, the gold standard of architectures was the VAX. In fact, we had one we used for all kinds things. Running Unix at that time, you needed one to use it effectively in a group. And so the architecture of the 32000 looks a lot like a VAX, kind of resembles it in some ways, very VAX-like. But that and also the fact that all of the silicon guys in the valley are in a cloud. And if Gordon Bell’s doing it, you’ve got to do it. So all of the semiconductor manufacturers were trying to do this thing that they variously referred to as a computer on a chip. And various instantiations of a computer on a chip were things like the 32000. There was a project for a while of putting a 360 architecture on a chip. Other people were trying to put PDP-8s on a chip. There was lots of experimentation in this crucible at this time. And that experimentation finally yielded the designs that we see today in the marketplace. And the 32000 was one of the options that a future history could have rewarded with success, but it didn’t. But it really was, from a software guy’s perspective, the ideal chip. It was ideal because it didn’t cause you to have to compromise your implementation due to the quirkiness of the underlying hardware.

Sanquini: Now, that instruction set was a superset of the VAX, right?

Freund: Well, superset, subset. We got to the point where-- one of specialties of mine was compiler design and optimization. Chip architects were trying to stuff more features, better instructions into the chip, said, “Look, we can go do this on the chip.” And the compiler guys are saying, “What would we use it for?” Because we were looking for very simple atomic things, which is kind of how the RISC argument ultimately went. But a very small part of the instruction set was generated by compilers. Probably because we just didn’t understand how to. But it did have a superset in some ways. From a purist point of view though, the fundamentals of it was that it was a simple 32-bit architecture. Simple is good. Simple is not denigrating it in any way. And that simpleness gives it kind of a symmetry, which makes it very easy to do several things within code generation. Lack of symmetry, you start special casing things.

Sanquini: Subhash, you were around at that time. I mean, you two folks were the earliest at National. What are your thoughts about how the 32000 started?

Subhash Bal: So a bunch of things came together. National had done pretty well in the past in the microprocessor world. In the sense that it took fair share of the market, but the market itself wasn’t necessarily very large. Products such as COPS, NSC800, Bit Slice Micros, Pace, and these kinds of products. So it was clear that there was something to be had here if we carry with our technology and come out with better architectures. Because the next big thing was going to be microprocessor revolution. More and more it looked like the equipment would be built around programmable products, and so microprocessor was the way to go. And since we did have a start in microprocessors, it was quite natural

that we build a better family, something which was much more advanced than the other competitors were thinking of at the time. And then that, along with certain other factors that came together, such as the vision of our then vice president of technology, Pierre Lamond. He wanted National to be on the map with respect to the best architecture out there in the world. And then along with that came along a bunch of very, very sharp engineers, all trained in high technology and processors, and so on, in Israel. So it was easy to form that group and get going very quickly. The right talent, the right quality and quantity, the right vision, and enthusiasm of the executive staff of National. Because that then allowed us to sort of get out of what was considered to be "jelly bean" products at that time. So all these factors came together at the same time. That's how it got started.

Sanquini: So basically, what we had was sort of a history of programmable devices, if you will. Then the workstation effort that we had at National kind of put this in a systems perspective, and drove, the spec and the applications for this product. Does that make sense, Bob?

Freund: Oh yes, certainly. I think that the architecture of the product really was driven from at least what the vision of workstations were at the time. It's a good thing semiconductors didn't have to depend on workstations for its entire growth; since it's really such a small market using today's perspective today. Back then, it looked like a big plum to go after. But the other thing that was kind of a-- a good and bad thing, good architecture, best architecture, in that we really carved out a lot of territory. We did several chips simultaneously: memory management units, clock drivers, the CPU chip, and floating point units. A That parallel effort, was required to fulfill this whole grand vision. But in some ways, although it presented a very nice marketing image was in and of itself a little problematic, as we'll see as we go on.

Sanquini: So we had a history of making micros. We basically had an application. And we had some talent in Israel. And I'd like to kind of get to you, Giora. How was that talent really structured, and how did you communicate? Because we had an engineering group in Santa Clara, engineering group. We had a group in Israel. Could you give us a view on that?

Giora Yaron: Yeah. It was a little bit harder and more complicated than described by either Bob or Subhash here. Because the architecture was "borrowed," so to speak from the VAX world. The marketing guys thought that all Israelis are smart. And the reality was a little bit more complicated than that. We were really asked to put on a piece of silicon more transistors than as a company had ever been put together. We were asked to have a process that didn't exist within the company. We were asked to use design tools that were not suitable for the task ahead. And we were asked to validate the system so that when it gets to the customer first time, it's going to be able to run Unix. And when you think about how many engineers we had and how many chips we were trying to design in parallel, I think that there's no company in history that I know that was trying to do so many things with such a team without really having the experience behind them or under their belt to demonstrate they're capable of doing it. So really, the task was something like climbing a 90% wall, totally slippery, and no place to grab something to climb with. And the only reason we eventually managed to pull off everything together really goes back to the quality of the engineers that we were able to assemble. I think we had in the country the best team ever. I think that if you go and ask, even today, about the quality of the team that we had then, it was recognized as an A team, but that was an A+++ team. But even that wasn't enough to get basically the silicon to work on the first cut. And very frankly, in our industry, you call it steppings, we were going through step A mask set, stepping B mask set. And once we finished the American alphabet, we moved to the Israeli alphabet to be able to get everything right. And even that alphabet wasn't enough, so we

had stepping A1, 2, 3, 4. And I mean, it was an unbelievable task. I think that the only reason we eventually ended up being successful in fulfilling the mission was we were young, naïve, inexperienced, which is really what makes the Silicon Valley to date what it is. If all the engineers knew what slippery walls they were about to jump into and try and climb, they probably would never have started the task. And we had that quality. Very smart people, very energetic, very naïve about the task, and that's what eventually allowed us to fulfill the mission. That was compounded by another difficulty, which is-- only today people are starting to appreciate it-- which is, working across geographies. And I still remember the big 4:00 a.m., 3:00 a.m., Santa Clara time phone calls. And I'm kind of waiting until, morning hours are going to get there so you'll answer the phone. So every time I move it one hour earlier, and gee, you are there.

Sanquini: Your early morning calls were enjoyable???.

Yaron: Sure. So, it was endless. And very frankly, I'm not sure that it would have ever worked if it wasn't the two of us-- maybe not necessarily me, but definitely you. Or a good communication between us, because you didn't have all the tools that we have today to enhance communication. And I think a lot of it contributed to the success, the fact that we were able to communicate problems back and forth. In fact we ended up translating this geographic gap, which is a barrier, into something that would work for us. Because we would talk early in the morning my time, before you'd go to sleep. And when you woke up, which was my early afternoon, we would talk again. So we actually ended up getting this machine to fix bugs and work and develop and work through the problems very frankly, 24 hours around the clock. And we really needed more than 24 hours. If we had 48 hours in a day, I think we would have known to use that as well. So I will get back to how we ended up solving many of these problems. But if there's a message to those who are listening to this tape, it's good to be naïve, it's good to be energetic, but you need early on some gray head around you to assist you in sizing the mission and making sure that you are properly resourced to make it happen.

Sanquini: Also we took on quite a task.

Yaron: No kidding.

Sanquini: You might elaborate on how we differentiated ourselves from, let's say, what was going on in the industry at that time. I mean, we had Intel, who was working on a CPU. We had Motorola working on a CPU primarily, and the same with TI and the same with Zilog. And we decided to take on the worlds first--

Yaron: All of them combined.

Sanquini: Well that, plus the world's first demand paged virtual memory. The first floating point.

Yaron: Floating points.

Sanquini: And interrupt control unit.

Yaron: 32-bit.

Sanquini: So we had to make that whole chip set work at the proper frequencies.

Yaron: It's a system.

Sanquini: And I think, to me, there was quite a bit of learning's, because we were building computers and simulating those computers with less power. I mean, we had big mainframes with 6 MP capabilities simulating computers that were maybe 5 to 10 MPs at the time.

Yaron: And as well as you know, the methodology that we ended up developing to debug those systems have given birth to several companies, one of which is Verisity which ended up being acquired by Cadence. The roots of that technology came from National Semi. We obviously didn't want to move into the CAD tools area, so therefore, we didn't have a problem with individuals that have worked with us, specifically in this case a bright guy by the name of Yoav Hollander, to take this technology and eventually commercialize it for the benefit of the industry. With the tools that we have developed today, when you run into a young engineer, for him it's a given. Oh, of course we use Verisity tools. Meaning, they don't even think that it was ever possible to pull something like this together without those tools. And this is just one side effect, because we were not focused on CAD tools. We were focused on something totally different. So we just had to develop our own tools to make things happen.

Sanquini: We'll get back to that, because I think it's an important point to explore that a little bit more. But I'd like to hear from Don. What's your view of how the 32000 family stacked up against the competition? What was your view?

Donald Alpert: Well, I think the '80s, which is primarily the period that we're talking about, was just an incredible period of evolution and companies entering and exiting the marketplace. And I think in this regard, the 32000, as Bob was saying, was kind of like best of class of these so-called complex instruction set computers. We learned from the VAX and those that came before, and coming up with a very clean instruction set. Right about the time that the industry, or at least academia, and then the industry, was in transition towards this so-called RISC or reduced instruction set computing. And then later the superscalar, having multiple instructions, executing in parallel. [Referring here to the "Swordfish" project] And I think that actually, technically, and in terms of architecture, I think the completeness of the solution, including we talked about silicon, but also the system software, as well as following those trends in the industry to deliver the compatibility, together with the performance that the silicon could provide. I think in that regard, technically, really it was quite an extraordinary story.

Sanquini: Okay. I'd like to get back to Subhash and talk about the market environment. We talked about how and why we developed 32000. Talked about how we structured our engineering group and how it stacked up. Let's go back, and what was the market environment really like in the '80s?

Bal: Okay. It was clearly a time of extremely high level of competition. The products like 8086, Z8000, for example, represented a class of products which were good, which were accepted. But people were looking for more. Because the problems that they saw that were going to come in the future were of a kind that necessarily could not have been solved properly with the 8086 or Z8000 class of machines. So in that sense, people were looking for something that gave them freedom to grow. Because this industry, sort of called let's say microcomputer or microprocessor based computer industry was ready to explode in growth. So there were different types of solutions being talked about. On the one hand, there were multi-user systems, Unix based for example. Then there were people talking about workstations. Companies like Sun, for example, were being formed. Then there was early variety of personal computers. They were not called PC at that time. Some people called it home computer, or small computer of some kind. Osborne, for instance, was born. So the microcomputer industry was going in different directions. There were issues related to what types of busses to be used, whether it was VME or multi-bus, or a variant or future variant of multi-bus, what have you. On the other hand, there were really, really successful architectures such as VAX that people loved and liked. And they wanted to see something that could be translated into microform. And essentially, in some ways, 32000 got its inspiration, if you will, from VAX, which was proven to be a good architecture. And if you could translate that into something usable in the chip form, that was just great. So people were hungry for newer things. Yet it had to be cost effective. It had to be performance oriented. And needed different types of software, all the way from operating system to application software. So there was a lot of demand, from many, many different sources looking for solutions. And it was a fun environment, as a result of that. And competition, I remember was very, very heavy. I distinctly remember that at one time, once we had announced our product family, and we were splashed all over the place. We were winning awards of different kinds in the media and so on, and people were acknowledging that this was a big--

Sanquini: Power point awards?

Bal: Those included. So one award we got was for one of the ads that we were running at the time. So I mean, you know, it was a fun time. All kinds of things were going on. And I remember that just the noise level had become so heavy that actually, there were our competitors that were looking for me to come over. And as a matter of fact, our notable competitor, Intel, had offered me to come over and run the 386 program. And it was kind of funny, because here we were, not even shipping anything really real in quantity, yet these people were running scared that something big is about to happen, so let's steal this guy away. And to my embarrassment, I was asked by my managers, "So, are you about to resign or stay?" I was completely dumbfounded, because I had not accepted the offer. So they said, "Well, there's an article, and Rosen said …" -- Ben Rosen used to write [an industry newsletter] -- And it was mentioned that something good must be happening to National's 32000 family, because Subhash Bal, the marketing chief of the product family decided not to quit and join Intel. So it was fun time. It was highly competitive, lot of work on the engineering side. Obviously folks in Israel and of course, back home in Santa Clara, were working very, very hard, but so were we. We were running around all over the world trying to sell our product and technology and winning accolades, that was important. Yet, at the same time, just hoping that these things come out and we actually deliver.

Sanquini: Little thing of execution there.

Bal: Exactly. So it was fun, and at the same time, stress.

Sanquini: Bob, you want to answer that?

Freund: Well, one of the things that I recall, because when you're doing the software and systems side of this thing, you're never really in the spotlight in a semiconductor company. Because you're the guy that's in a way a supporting role. In a way, you also help push the broom behind the parade, which is one of the things that we used to do. But we always got to the point of having to deliver something in demonstration form, a demo board, or a little system. No matter what the steppings actually behaved like, we had to make it work, okay? We managed to make these steppings at least look like they worked.

Yaron: Well, they did work, with limitations.

Freund: To some specification. So the question was, from a software perspective, we looked at performance and complexity of instructions. I mean, today, we fake out a lot computer systems, because we have so many instructions per second behind us we can emulate anything we need. So if it isn't there today, we can make it. Back then, we didn't have that luxury to do that. Every instruction per second counted. And at the time, some of the early processors weren't meeting their spec. And I recall hearing time after time after time at sales meetings we would go to and rep meetings, they'd say "Where's the 10 MHz part? Where's the 10 MHz part?" Now I know this takes down into a really overly simplistic view of all the challenges and difficulties that Giora and his crew were facing. But it really is kind of the one thing that can get you, the one thing is that they say, "You advertised this, you specified it, when are you going to deliver it?" And that lack of delivery or the timeframe of delivery might have changed things. Maybe if we had delivered initially to full spec. But on the other hand, we had too much to deliver. We were carrying a pretty heavy load at that time.

Sanquini: Giora, did we have too much to deliver?

Yaron: No kidding. But maybe just to compliment what Subhash said earlier about the competitive environment, and talking about companies stealing talent one from another. We ended up having, in a period of less than four months, six engineers, some of the top talent that we had, resigning. It was hard to see initially where there's a problem there. But one guy comes in and he resigns. "Where are you going to?" "Can't tell you." Few days later, another guy from another discipline comes in, he's resigning. "Where are you going to?" "Can't tell you." Well, at the end of the game, it became very clear that it was a mini-raid that Intel did on our engineering talent. These gentlemen included, in the architecture side, but Sondan [ph?], who was from the architecture side, Gideon Yoval [ph?] from the software side, Lev Lavil [ph?] from the circuit side, Doe Avnon [ph?] from the floating point side. Six of them basically in four months resigned. One more, and we were basically going to start suing. But there's a saying, at least in my country, that crime does not pay. And there's an interesting story there. You're all familiar with the major write off that Intel had to do for $470-some million.

Sanquini: 475 million.

Yaron: What's the exact number?

Sanquini: 475 million, according to the research that we did.

Yaron: $475 million. Okay, so $475 million, and it's a bug that statistically has a probability of ten to the minus twenty-eight, nine?

Alpert: 24,000 years.

Yaron: Whatever, 24,000 years.

Alpert: Yeah.

Yaron: Interestingly enough, the same bug was discovered in our product called Swordfish. And interestingly enough, it's the same engineer that worked for us ended up working on the floating point for Intel. So one actually needs to ask himself, if you have an event with a probability of-- what did you say? 24,000 years?

Alpert: Yeah.

Yaron: Two remote locations around the globe, what's the probability for that occurring. But there's only one thing in common, and that's an engineer that worked for us ended up working for Intel. So you want to talk about crime doesn't pay, well, this is one example of that. But going back to the more serious side, it was endless competition talent, because everyone was competing and trying to climb these very slippery walls. And it was really a question of who was going to be able to get it right and deliver before the other. We know the competition had theirown problems. I know that Intel advertised some of their problems with 286 and getting it to work, etc., etc. When I analyze why we eventually failed, I think it's for two reasons. We had an option to fail, because National Semi had a strong business in analog and logic. Intel just could not fail. If Intel fails with a microprocessor, well, that would be the end of Intel. Having an option to fail basically creates a situation where management does not give you enough resources to get things done. Whereas at Intel, Andy Grove himself was part of a task force, eight o'clock in the morning, would sit there, take notes, what are the resources that are required, assign the resources, and that was the end of it. So that was part of it. The other part of it, I'm not sure the 32000 architecture or program was born in the right company, so to speak. The DNA of National Semi was a high volume production analog, digital and the like. The sales force didn't know how to sell our products. They didn't realize that they were selling computers, for God's sake. You're not coming with your bag, okay? "Do you want this analog? Do you want this logic? You don't like this one? I'll give you this one?" You're selling computers, for God's sake. You're selling software. We didn't have the right talent, we didn't have the right compensation scheme for these guys. I think that designing on a computer is different than any designing any jelly bean product. So it was really a combination of things. We obviously have contributed our share because we have not executed in a timely fashion as we originally thought we would be able to. But we, as a company, we had a safety net. It wasn't a must for us to succeed. I think that if it was a must for us to succeed as a company, we'd have been able to cross this Rubicon and be successful at the end of the day. But history will judge. Or has judged.

Sanquini: That certainly was a real problem. We probably have agreement on that, with this panel that that was a real problem. Although I'd have to say that Intel had that problem early on in that they were a memory company. And memory doesn't require software.

Yaron: That's true.

Sanquini: They got religion in the late '70s, which were a few years before maybe the rest of the diversified companies like National or Motorola or TI. Because if you look at all the companies that were not major players in general purpose microprocessors, it was those players. I mean, Intel finally got religion.

Yaron: They had no choice, just to remind you. They had no choice. The Japanese reamed them out from the memory business. They had a major rival, IBM interested in them. At the time that the competition was kind of heating up, Intel had no choice. The Japanese have taken them out, they invented SRAM, they invented DRAM, they invented the e square memory, they invented flash memory. Few days later, few years later, the Japanese have taken over those industries. So they finally checked out from all the memory space, and it was live or die by microprocessor. We had very successful businesses in other areas, which again, is a mixed blessing. You don't place your bet on one line of business. But that's a mixed message. On the other hand, you don't necessarily have to assign as many resources required to make it successful. So I don't know. Intel didn't have a choice.

Sanquini: I think that's a valid point. But we ought to go back, I think because it's important, our execution at National versus the execution at say, Intel or Motorola is still-- in the final analysis, it would not make us more successful or less successful given the scenario you just depicted. But what I thought we did during that period was we had accelerated our learning. 32032 was very buggy. We got a little bit better in methodology. Roy Yamanouchi developed the correct by design methodology. We had less problems on the 332. And then finally, with the 532, we were able to run UNIX on our first silicon.

Yaron: Ran Unix out of the box.

Sanquini: --that product out, running UNIX on first silicon. So it was quite an accelerated learning. And I think you ought to speak to that a little bit, about that learning. Because I think that had a lot to do with the tools that were available and that we were developing. Because we were concurrently developing design tools, CAD tools, while we were designing the chips. And we got better and better at this. Also, from Bob's perspective, we were designing software suites to be able to test the entire chipset. And that got better and better as we progressed. The 32032 we had very little. By the time we got to the 532, it was much, much better.

Yaron: It took us a long time to master the technologies, the CAD tools, the verification tools, and the likes. But I don't know that it was any different than it was with the competition, being Intel, Motorola, and the likes. But we did not have this buffer provided by a sales force and a marketing force that would buy us the time to get it done. So by the time we got it done, we basically were in a situation where Intel was designed into the PC and to a large extent it was game over.

Sanquini: Yeah.

Alpert: You know, I think I was going to say, not to take anything away from all the business complications, but I think if you look at what the 32000 tried in about 1980 to pull off, it's the same challenges that were really fraught with difficulty, I think, for the competition, through the '80s and beyond. And if you look at it, there was a question of putting all that functionality in place for a systems solution that works. And I think the semiconductor companies knew how to do one chip and make it work. But to make them all work together wasn't something they mastered. And if you look in the middle of the decade, even Intel and Motorola trying to decide, do they put memory management on, or do they leave that for the customers to roll their own? What about floating point? What about cache? It was very kind of difficult decisions about what problems to throw over to the customers. And I think just in terms of getting all these things to work and the quality, I know I joined Intel in '89, and the 486 was coming out. And that product was fraught with these kinds of quality issues, where Intel did not have the capability of fully debugging it without the customers' help. So all the companies went through that.

Sanquini: You know, with that said, I think we ought to get back to why would the customers even want to use the 32000? Okay?

Freund: Okay.

Sanquini: Subhash, why would any of our customers even want to use the Series 32000?

Bal: Wonder why. Actually, there were some really interesting things that were going on at the time. One is there was this whole question of building a high performance system but in a cost effective fashion. And the applications are not necessarily exactly the same. So, now you have a good architecture, but are you now bound to using the exact same chipset all the time? Because that would not be cost effective. So 32000 series, its architecture was actually designed in such a way that you could have an easy growth path. You can start from all the way down to 8-bit if you wanted to. 8-bit data goes to 16, to 32-bit. If you wanted a very large amount of memory to be managed and you needed a memory management unit for that purpose, sure enough, without changing software, you can add memory management and manage the memory. If your application demanded floating point, that was the only time you would use the floating point chip, otherwise you won't use it. If you just wanted a high performance machine, which had good instruction set, then the 32000 microprocessor by itself was good enough. So you can play with its power. By power I mean application oriented power, memory management, floating point, what have you. You can play with the bus size, because that determined the cost of the other chips that were around the processor. So from a hardware perspective, the customer had a lot of choice how to build their system, if it was done in such a way that was also upward compatible. So that was nice. The other thing was that at that time, software, of course, has always been an issue. And most of the programming at that time was being done in assembly language. And so people wanted to find a way of doing something more productive, which compilers could do, but we did not have an architecture that can support compilers all that well. So this 32000 had the promise of supporting compiler based software. Because the instruction set was so designed that you could have efficient code, which you can use compiler generated code, compiler for efficiency of software development. So all in all, it had a very good mix of hardware and software capability, if the software indeed was eventually going to be available. Clearly, this was going to be supported very nicely by the system. So it was a combination of these kinds of things.

Sanquini: Our initial customers had a European bent. What do you read into that, you know, when you map that back into the attributes of the Series 32000?

Bal: That's interesting too. One of the things that was going on at the time is that companies like European companies wanted to also win in their market space. They were not necessarily that well known for having leading this thing, positioned in the computer business. So there were companies, such as Siemens, for example. And there were entrepreneurs within Siemens that wanted to take the company forward, wanted to take some risk, if you will. And at that point in time they put together a system which was the state of the art. So they were a little bit more motivated. They did not have the baggage of the past as much as let's say some of the domestic players did. And if you have some good entrepreneurs within a company driving the business, then that was all a nice fit.

Sanquini: Bob, what were your insights there? Because you worked with the Siemens stuff.

Freund: You mentioned my good friend, Strock Zimmerman, Siemens. Strock Zimmerman was the guy who was running the little systems group within Siemens. And he was primarily a software guy doing a Unix-based product there. Now, Siemens, you have to understand, is really-- at the time was more into circuit breakers, power transmission, motors, that kind of traditional electronic business. And their offices near Munich, were very orderly indeed. In fact, they were so orderly that at six o'clock every night, everybody was expected to be out of the building and home, and they locked the doors. And they let dogs out between the building and the fences. But as you know, or maybe you don't, all good software is written only when the moon is out. Okay? So any bunch of good software folks worth their salt are working all hours of the day or night. So one evening, they had been working hard on a project and found that it was past six. And the doors were locked, and the dogs were loose. And it was close to midnight at this time. But that didn't stop them. They decided they had to get home, anyway, right? So they took a run for it across the compound, with the dogs right behind them, up the fence, crossed the barbed wire, and escaped. A week later, the entire software--

Sanquini: You did that Bob?

Freund: No, no, I didn't do that. Hans Strock Zimmerman did it and his software guys. So within a week, Siemens moved them out of that Siemenstadt near Munich into their own offices in Munich, which was in a much more relaxed environment. But that's the kind of cultural impedance mismatch that you see. I think there was some degree of that in National. You certainly saw that in Siemens. I mean, Siemens was a real hardware company. Trying to do software, so many things are different. At National, as well, we had our differences in culture, hardware and software. But I worked hard. Trying to explain Michelangelo to a horse is difficult, but we did mange to get a few points across.

Sanquini: Subhash, who were some of the other customers?

Bal: In Germany there was another very good customer of ours, Bosch. And they were representative of a nice control system that they built around our products. Then there were French telecom companies that were looking to architecture such as these. In the U.S.-- oh, and there was another one actually in the U.K., a company called Acorn, which was a-- home computer was their target. And they turned out to

be one of our customers. Eventually, as you now, Acorn has grown into or at least somehow become another company called ARM. So apparently the genesis of that company was the founders of Acorn. It's a very successful entrepreneurial company. Then in the U.S., there were multi-user companies, such as Sequent, for example, that was formed in Oregon. A bunch of Intel executives formed that company. Datapoint was another company that used our product. It turned out that a workstation business Tektronics was a very good example. Tektronics actually had our competitor's product and we designed them out, because they again, for the goodness of 32000, they liked it so much that they actually gave up their existing design, and they came with us. So there were companies in the microcomputer business, whether it was multi-user system or workstation or small home computer like systems, control systems, these were really our major customers. And at that time, the workstation industry, all the start ups, were in the workstation. So most of the workstation companies actually were going to the 32000 at the time. I left in 1984. And by then, we had about 250 OEMs that were actually working with the 32000 family. And many of them, indeed came out and were successful with their products. Particularly, I remember Siemens Unix-based system on a 32000 lasted for long time in the marketplace.

Sanquini: So we certainly had the customer base primed for the business. I'd like to get back to some of the challenges that we faced in the architecture and design and in software as we were going forward. Because they were enormous in bringing out the products. And I do think there were some learnings as we talked about previously. And maybe we ought to start with the architecture this time. What challenges did we have? Because when we finally came out with the 532, and I think you were the lead architect there, at least one of the lead architects there for the 532.

Yaron: He was the lead. Not one of, he was the lead.

Alpert: Yeah, yeah. I think by--

Sanquini: Uri Weisser was still part of the 532 program.

Alpert: For the architecture, yeah. Yeah, I was there for the architecture. I think at least as-- I joined in '85, and I guess there was already some kind of turmoil at that point about how committed some of these customers were, where to find new ones. And I think one of the biggest challenges that we had was such a diversity of customer applications. You know, we hinted at some very large scale multi-processing. There was a whole emerging market of what they called either mini supercomputers or super mini-computers.

Freund: The segment that never happened.

Alpert: For the 32000, yeah. Parallel processing with some vector extensions. And at the same time there would be single user workstations. And we wound up along the way getting a laser printer application--

Yaron: Imbedded application.

Alpert: --Embedded. And so we would be sitting there looking at vector processing algorithms and how to make the floating point run, and then we'd look at postscript and how to draw circles in these images. And it was a challenge to address that breadth of market. In fact, I remember actually one thing that we hadn't talked about, even when we had that session last night was there was like an add-on chip that a few people managed to work on that did in fact add some type of DSP vector processing capability to the 32000. I think it was supposed to be able to enable a combination printer/fax machine at the time. And it was supposed to be able to do the analog--

Yaron: 580.

Alpert: --the analog telecommunications over a modem. So it was-- you know, it was every day there was kind of a new opportunity or kind of a new need to analyze. I mean, it's a challenge to try to develop a technology that can address such a range of markets.

Sanquini: I think there was a tipping point though, where we started to look at those. I call those imbedded applications. The fax/printer applications. And I want to talk a little bit about this. We still were going at the general purpose microprocessor market up until the introduction of 386,. But prior to 386, we worked with Burroughs for a few years, and we were designing-- again Bob, you did a lot of work in that particular area with Burroughs. Maybe you could talk a little bit about that. Our success or failure with Burroughs kind of triggered all of these other applications, okay? Because we were working on general purpose, 532 was going to be the product. We at this time had learned a little bit about how to get a product out. And Bob, why don't you talk a little bit about Burroughs.

Freund: You know, Burroughs had at the time a broad collection of miscellaneous equipment, a lot aimed, I would say, at the financial industry, financial market products. Things such as check microreaders, sorters. They had ATM machines. They had all kinds of, even some programmable products. Some financial workstation products. And they were really interested in having a universal solution, kind of a core engine architecture that they would use for everything. The idea there was roughly this. It was that our tool investment and the training of our software engineers is the single most expensive component of developing a new product. And they would like to be able to preserve that investment product to product to product, as they move the engineering team to different things and new things. So that would all be enhanced if they had a wonderful chip that they could use in many of these different applications, whether it was visible as a workstation, or a programmable product, or a PC type thing, or if it was stuck in as a communications controller, it was all the same as far as the software guys were concerned, they could use the universal tool set. As you can see, and what's developed to a large degree in the marketplace today is that's been wanted by a lot of companies, and that's a pretty universally recognized need today. But that put a lot of stress on parts of the system that weren't silicon. I mean, one of the stresses it put on us was on compilers and tool sets. Because it's a different thing to generate a program, a load module, that is going to be run by an operating system that you expect to be there to provide certain underlying services. To be able to write a tool set that's going to generate some code that you may be burning to a ROM and run on bare hardware, okay? Now, you'd have to provide other utility, other functionality in a ROM-able way, typically. And not only that, but we had other architectures as well that were slight modifications. We had to develop a compiler technology around those kinds of needs where we had multiple front-end languages and multiple back-end processors and targets to develop. So that Burroughs situation really opened my eyes, I guess, to the importance of

preserving the engineering knowledge base in the customer's mind, his application, software knowledge base, and preserving that from product to product was important.

Sanquini: So that was a successful program, though.

Freund: I think it was.

Sanquini: It was a very successful program.

Freund: I mean, we developed compiler technology that ultimately was purchased by other companies, and is still, I mean today there are sons of that technology, including I would even go so far to say as Java is fundamentally in that same idea of common intermediate languages abstracted from targeted architectures where just in time compilation at the time of installation is done. That's the exact same thing that Java does today.

Sanquini: And Burroughs had made, from a business perspective, a commitment to completely move over from Intel to the Series 32000. And we had all of their engineering and manufacturing facilities pretty much converted I think. This was a two-year program, Bob?

Freund: Yeah, yeah. They were working hard and long. And their concerns at the time for Intel is they didn't think that Intel had enough reach. They didn't think it would go far enough up their pyramid of applications as they saw it. But ultimately, we saw a torpedo coming at us amidships, as we found out later when the Intel management team went to visit Burroughs.

Sanquini: Yeah. And I think this is also an indication of something Giora mentioned earlier. And that is that we're a company, and a diversified company, and the senior management, being our CEO and his staff, basically, really weren't cognizant of the importance of when you're selling a microprocessor, you're really selling the company. We were blindsided, because we had dealt with, Burroughs, Colorado Springs and Rancho Bernardo. We actually never called on Detroit, where the chairman and the CEO sat. So we never talked to the chairman and the CEO. And actually Grove and Moore made a visit with Jack Bush, who was here previously. He was the account manager for Intel. They convinced them that the 386 was a lot nearer than it really was. But they did that because they had a relationship and what I call a credibility technical backlog with them. So they said, "Don't make the change."

Yaron: We'll eventually get there.

Freund: There's a very important thing though that happened at Burroughs. When a company makes a commitment to an architecture to build all of its products around, they're basically betting the farm. They're betting their company on you. And they expect you to be able to bet on them too, okay? And that's one thing that we just didn't have the management culture to know how to do at the time.

Sanquini: No, we did not understand that at that time. That's something, I think it was a good lesson for all of us. And that was the tipping point, Don, that then forced us into looking at these embedded applications. And we said, "Okay, that was the first sign that the general purpose microprocessor, it really is going to be a difficult market." And then the coup de grace came when the 386 came out and Compaq basically ran with the 386 at that time. But let's get back to some of these challenges. What about the engineering design challenges, Giora?

Yaron: Well, maybe to go back to what Subhash had to say, which is working with European customers and having more European customers. In a sense, it was a mixed blessing, if you will. First of all, having customers is a positive. But having them European immediately dictates that the level of their demand is, I would say almost ten fold over other traditional, at the time at least, American industries. Take a company like Bosch. Bosch doesn't buy a product just because it's functional. The fact that it's functional for seven days, nothing cracks, that's fine. But they go and characterize the product-- I don't know if you remember, we had tests where you check the sensitivity tests on temperature and voltages plus, minus five percent, and ten percent. They would find in one of the voltages in one of the instructions there was a small hole where it didn't really show that it was completely functional there. No, no, no, they wanted to understand why it's not going to be functional. Now, were they doing it because they were bad people? No. They were great people. The best engineers you can find were in Bosch. Hardcore, traditional engineers that knew that the floating point is going into-- I don't remember how they priced the robot, but it was something in the million range. So we were really the heart of that specific piece of gear. And that reputation, if you buy Bosch, it never fails. I mean, never fails. It's raining, it's freezing, it's hot, it's whatever. It never fails. So they would characterize the products to an extent that-- I don't know that anyone in the industry used to characterize the products the way they did. So in a sense, we owed them a lot because they forced us to look at our methodology and basically clean everything, and I mean, everything. There wasn't a single stone uncovered that they left, both them and Siemens. And we thought we were doing good, because, you know, the thing is running Unix. Well, what do you want? You wanted something running Unix. It's running Unix. And it doesn't crash. What do you want? They put it into the oven, take it out of the oven. They would freeze it, but it always work. They would cook it, they would drop coffee in it, and they expected it to run. So if we eventually ended up delivering extremely high quality products, a lot of it had to do with Siemens and Bosch teaching us how to meet specs. Meeting spec is not cutting any corners anywhere. And every bug that you leave and you kind of circumvent and you thought that you are cheating the system. At the wrong time, that bug is going to show up. And we've had some of those at Siemens workstations, in the banking industry. So they really taught us what a demanding customer is all about. Which was a good preparation for the Japanese market when we went into the embedded market. But as you said, it was endless challenges, because none of the tools were there, and we had to develop them from scratch. So what's the moral of the story? Try and assess the size of the complexity before jumping into the pool before you discover you're jumping into the pool and it's not really full of water. And in some cases, we were jumping into an empty pool.

Sanquini: Well, in many respects though, progress in our industry has actually been generated in that manner. You're fulfilling a need. There's a need, and if you're going to fulfill that need, you have to learn how to develop that solution to fulfill that need, right? That's number one. And just about, I'm sure, every one of your companies that you're working with, they have a need right there that they're filling with a compelling solution. And fortunately or unfortunately, the other way it's not so good either, where you have the product and you have no customers and you try to figure out how to develop the customers. That's even worse.

Yaron: No question. We did the right thing. There was a customer, there was a demand. But the question was, “How to get from here to there?” And very frankly, again, it was a huge organization, but if you are to single out some individuals, with a guy name of Sidi Yomtov , I don't know if you remember the guy.

Sanquini: Very, very, well.

Yaron: If it wasn’t because of him, we probably would never be sitting here or never shipping product altogether. The guy was capable of taking an engineering team and working them 48 hours around the clock without even taking a nap in between. He would just go outside, bring the food, feed the guys. And he just sat all over them forcing them to execute.

Sanquini: I remember when he did the bug reduction of the 532. He was in charge of the simulation for the 532.

Yaron: Simulation and chip design. Simulation was only a leg of that.

Sanquini: And chip design. I’ll never forget him, because of his never ending passion and energy to meet our tough schedules.

Yaron: How can one forget him?

Sanquini: Because if you remember at that time, you were using the mainframes to do the simulation. And those old mainframes took 18 months to complete our simulation efforts.

Yaron: Yeah, yeah, yeah.

Sanquini: 18 months I think to simulate the 532. And he was the key driver for that. And the one story that I’ll never forget is not a story, it’s that basically you were using the mainframes 8 to 16 hours a day. when we went home in Santa Clara, you took over the mainframes.

Yaron: In theory, when you guys were asleep.

Sanquini: When we went home, right, to go to sleep. And when we’d get near the tape out, you basically took some liberties. And actually we made the tape out because you went into your 24 hour work mode.

Yaron: A guy has to do what he got to do.

Sanquini: I got a call from the finance group saying their computers are being held hostage by Tel Aviv, okay? And what is going on? And I said “Well, I’ll call Giora,” which I did. And unfortunately, Giora didn’t

answer his phone. He always answered, but he didn't answer his phone that morning. And I had to tell accounting that "I'll call him again." And finally he got the tapeout done before we finally communicated… But that was a good joke.

Yaron: You have to remember, we didn't have Andy Grove that would sit in the meetings every morning and say, "So how many MPS do you guys need? 20, okay, here's 20 MPs. Fine. Go and use it." So we really had to do some stuff that was very unconventional at the time, like taking over some of the computers. But another story along the same lines. We ended up using the CPUs that we have developed to build simulation boards where we'll run the simulations over our own CPUs that we have developed, basically to simulate the next generation stuff. And one of the problems we did have, interestingly enough, you know, we would develop the products in Israel, ship them overseas to get them manufactured. On the way back, the U.S. government didn't allow us to ship it back to Israel because it was over the MPs that were allowed to be shipped without a specific license from the U.S. government. So we are developing the damn thing, we are shipping it out just to be manufactured at the low cost location, and want to ship it back to test it, it's impossible to do it. So a lot of things were going on, some of which had to do obviously our own capabilities of dealing with the complexity of the technology. But even regulations on shipping products from one geography to another ran into all kinds of stumbling blocks having nothing to do with us. And going back to the simulation stuff, you are trying to build a-- what was the 532, in terms of MPs? I don't even remember. How many MPs? You were trying to do a--

Alpert: How about 5 to 10.

Yaron: 5 to 10 MPs machine with a 5 MPs machine to simulate, and then to simulate all the permutations that the system could run into. And then to simulate the full chipset. I mean, this was an unbelievable job, no question.

Sanquini: Yeah, it was well done by Sidi . Bob, what about the challenges in the software area that you faced?

Freund: Well, I said before one of the challenges that we had was the problem of developing a compiler set that targeted multiple architectures underneath that. Because National had more than just the 32000 line. We had some imbedded controllers, we had some NSC800 stuff, the old stuff. We had some of the COPS line was there too. We were supposed to generate code for all of these things. And customers weren't going to be happy with just one programming language either. I mean, originally, there was a lot of Pascal legacy to the 32000. The original architects came from a kind of Cal Tech Pascal nickel's worth-- I call them nickel's worth languages, where you can't even write a memory allocator in the thing. But that Pascal orientation permeated a lot of things. But then when the market turned out to be a C-based market, and then later on C++ market, we had to take a sharp turn. And not knowing what the next turn was going to be, since the whole compiler choice thing was very in flux at that time. Every month, there was a new language that was out there. We developed the technology to use a common intermediate language. Just if I can turn a little geeky on you just for a moment. One nice thing about a common intermediate language is that you can represent all programs in a tree form, a hierarchy form. And if that format is common between all your languages, you can make just one investment in global optimization and apply it to that tree. And that immediately gets you more product to deliver in terms of a software of view, saves you money. And then you take that optimized trade and then do code emission for the

specific targeted processor that you're working on. As we were going through this common code generation thing, we had kind of an internal conflict with some of the early guys, the Pascal oriented folks that had originally architected the thing. Because we found as we were doing the compiler work is that a lot of the fanciness of the machine's capability and the addressing modes and stuff like that, we just had no use for them. We just could not and did not generate code to use some of these features that the chip had, because we couldn't find a more efficient way just on using just some of the basic stuff in the chip. So that was a challenge. It also taught me that simplicity is a good thing.

Sanquini: Always.

Freund: Always, always a good thing.

Sanquini: Let's move on. Unless anyone have anything else to add in terms of the challenges?

Alpert: You know, there was just one thing I was thinking when we were talking about some of these different customers, and maybe particularly because I joined in the middle. I was just thinking, kind of the bigger picture of the computer industry at that time, and particularly, say mini-computer, was that you had some of the major vendors were starting to apply their own microprocessor technology to their product lines. We talked about Digital with the VAX, Data General had their own chips. Hewlett Packard was doing their own chips. And I think it left a-- I don't know the exact numbers, but certainly a large portion of them, a lot of players in the industry that didn't have the capability. That were in the mini-computer industry but didn't have that capability, weren't prepared to invest in it, as well as the start ups that were looking to do these multi-processor systems that needed to have some type of microprocessor solution. And I'm just thinking that was probably a lot of the success that National had was it was meeting that real need. But then of course, what happened was, even those mini-computer manufacturers, by the end of the decade or the early '90s, they might have developed nice microprocessor technology, but they could no longer afford the silicon investments to sustain that. And so it was a trend really, I think throughout that decade that the high volume 32-bit microprocessor manufacturers, like Intel and Motorola, were going to be picking up that business. And even Motorola slipped by the end of the decade, and a company like NCR, that had been doing their systems, that had adopted their architecture, went over and picked up the Intel architecture. So in a sense, I think this whole kind of, where did 32000 fit in and what were some of the problems, even once the execution was straightened out were really just kind of an indication of the whole industry trend.

Sanquini: Yeah.

Freund: The bones of many of those companies litter the Route 128 corridor in Boston, don't you think?

Yaron: Yeah.

Bal: So Dick, I'd like to add, just explain a little bit more on what Giora has been talking about. This kind of product line, actually bottom line was, which we had started to recognize, was that it needed really strategic selling. And strategic selling meant on the one hand you work with the CEOs and the presidents

of the companies, then all the way down to the architects of the target customer. So what that meant is, not only the sales force had to do its job with respect to relationship building and keep that going. But at the same time, our engineering sales force in the field, namely FAEs, needed to work with the real engineers. And so it was pretty tough at the time, because these people, for obvious reasons, were not trained or incentivized to actually sell these kinds of products. So one of the things that, when you recognize this, if you're able to achieve at least in the technical sense that the technology sells, the FAEs, were able to form a group of FAEs, which were more or less dedicated to selling microprocessor technology. And that helped quite a bit. But that came much later. And unfortunately, some of our competitors at that time, if they had problems, were not able to really take advantage of that. I distinctly remember, we had a big feature of the demand page virtual memory, and 68,000 people were not able to deliver the product. And we were the only ones who understood the problem they had and we didn't even have that problem. Now, that could have been a golden opportunity for us to actually displace most of the 68000 based designs at that time. But we just did not have the firepower of the FAEs to go out and make this happen. So at all points, it was not really just the salespeople, the people who backed them up, and eventually, our own execs, who obviously were busy selling other products in the rest of the company. It was all a sense of lack of strategic selling, which became obvious later on.

Sanquini: Well, you're leading us into the next topic, which is why we failed. And that certainly is one reason. I'd like to open it up to all of you. You want to add to that? Bob?

Freund: I think we got off on the wrong foot, actually. In some ways, we were aggressive, young, and I think Giora said naïve. I might say maybe we were stupid in some ways, maybe purposefully so. Because to see the truth then would have meant a disaster for us. Because we started off laying out this complex chipset. Multiple chips with colored pencils and Mylar, and then digitizing them, okay? And then trying to get all the Mylar sheets to line up in the right places.

Yaron: There wasn't enough Mylar in the universe, I think.

Freund: There wasn't enough Mylar in the universe.

Sanquini: By the time I got there, we moved to Calmas.

Yaron: We started with Mylar.

Freund: The first chips were done on Mylar.

Yaron: Totally quick.

Freund: And it was almost like it would have been better for them just to wait until they got the better tools than to start with that Mylar. Because I don't think it got them a timely market advantage, starting with Mylar.

Yaron: We had to convert eventually.

Freund: Yeah, we had to convert.

Sanquini: Actually, that was the state of the art of the industry though, at that time. And I think that was the reason it took so long for everyone to get a chip out.

Freund: But we were shooting for the moon, meaning we're using the tools.

Sanquini: But we took it an order of magnitude in the degree of difficulty.

Freund: We added dimension to it. I mean, other guys were busting their guts trying to do one chip at a time. You know, we were trying to do what? Five?

Bal: Five.

Freund: Five chips at a time, on Mylar.

Yaron: That eventually will run Unix.

Freund: That eventually will run Unix. So getting off on a bad foot was one of them. That, and betting on so many things that just had to be right. That we didn't have. We had to create. Giora mentioned process. There was CAD tools, there were test tools. There were compilers, compiler technology.

Yaron: Compiler technology, yes. We developed compiler technology for our groups.

Freund: We had to develop compiler technology even before some of the ground breaking work on optimization had been written, we were using it.

Yaron: We're software guys, in the software industry, and we were coming from a jelly bean industry.

Freund: So you have this impedance mismatch.

Yaron: Yeah. I think a lot of things we would have done differently if we would have known then what we know today, obviously, but that's the easy part. I think that the complexity was one notch above the rest of the industry, being Intel, Motorola, and the like. So we were trying to climb a steeper mountain. And we were coming from behind because we were a high volume manufacturing company with jelly bean and analog and that kind of stuff. So first of all, the gap that we had to close was probably bigger than the gap that a company like Intel had to close. But again, if I have to prioritize why we eventually lost, while the

execution was an element of that, I would still come and say it's the ability of a company like National to put the other elements that are required for success in their sales marketing management awareness. Because really, a commitment to an architecture is not a commitment for an analog device. Analog device, you design it in today, you design it out tomorrow. You can probably do it five times in a given year.

Sanquini: Okay, we'll get back to that in a moment, okay, Giora, we're to just hold it there.

<break in recording>

Sanquini: Why don't you just summarize your thoughts on why we failed, one more time?

Yaron: Okay, so first of all, I think we have set out to do a mission that was above and beyond what anyone in the industry was basically aiming for. So if Motorola and Intel were somewhere here, we basically took the architecture a notch up, which is something that Bob has described earlier. So not only that we were shooting higher, we were really coming from behind. Basically coming from the analog jelly bean kind of product lines. So we have a really steep mountain to climb. So that was one on the technology side. And when I'm talking about technology, it's process technology, design technology, verification, technology testing, technology-- you name it, every element that has the word "T" spelling technology, we really had to invent, because nothing was available. It wasn't Cadence that you could go and call them and buy the verification tools and etc., etc. So that was one element. But in my mind, this contributed but did not lead to the loss of the war.

The biggest problem was like I think Bob mentioned earlier, as a company, we were doing too many things. And we couldn't bring the focus to one single product line like the 32-bit microprocessor like Intel did. So beyond execution issues, if you can kind of put them aside for a minute. We didn't have a qualified sales force that knew how to sell it. We didn't have the technology people supporting the sales force to be able to do it. We didn't have incentive programs that would allow the sales force to do it. Meaning, you go out with a bag and you sell-- you start in the morning to sell jelly beans, and the evening, you bring the cash. In our case, you have to win the design, and that's several months. When it goes into production it takes probably another year or whatever. You need a totally different incentive program. Early on, we didn't have that either.

And last but not least, we failed to understand the strategic importance of such a design win in the eyes of the customer. Because, yeah, it takes you about two years from the time you get started until you see revenue. That's our view. But the customer view is that if he has made the wrong decision, he's now stuck with you because for the last two years, that's what he has been designing. So if he is going to throw you out, he's going to start another two years kind just like it. So really, the sales force starts at the lowest level with the engineers, goes to the managers, directors, VPs, but ends up at the CO level and the chairman of the company, especially given the fact that what Bob said, they don't want to train the software engineers to do a little bit of Motorola, a little bit of Intel, a little bit of National. They want to make sure that their engineers are interchangeable, so to speak, can move around between different programs. So it's really a proper decision. So we were selling still to the engineers, they loved us, it's the best architecture. We were selling to the engineering management, they loved us, still great. Even the purchasing managers loved us. But that didn't count a lot when Intel flew in the Gordon Moores of this

world and Andy Groves of this world that ushered them in to the corporate level, made sure that they understood that this is not a small program they're making a decision, but they're really betting the company on a given architecture. So I think it's the combination of all of those.

But again, it all started with us not really understanding what we were getting into. Being young and naïve, and I think with the gray hair that you see around here today, we probably would have made other mistakes. Not the same mistakes, and hopefully less. I don't know if you guys agree, but hopefully less.

Sanquini: Yeah, I agree. I think one of the lessons here was that as complex as the technology was, and as complex as our problem solving was from a technical perspective, basically, large business decisions were made on the basis of the company. And I can see Intel saying, "This is our business, we only do micros and some memories. But we only do micros, and are you going to sign up with National Semiconductor, who basically makes everything?" And I think it's the same for Motorola, and I think it's the same for TI. Broad, diversified companies. Very, very difficult to get their attention. Even though you have the technology. Because if you really look at it from an architectural perspective, every one of our competitors except Intel had a better architecture than Intel.

Yaron: Yep.

Sanquini: And we had a better architecture than our competitors and Intel. It didn't have to do with the architecture or the--

Yaron: But I'll tell you, there's one thing positive that came out of it. Several positive things came out of it. But one of which, it was so demanding, it was so difficult, it was so impossible, that it really melded the team to a point, 25 years later, the guys are still socializing, the wives are still socializing. Meaning, this is something you rarely see anywhere. And it all goes back to the fact that we were working around the clock trying to climb this steep mountain. Successful or not, it did have a positive thing in building a team and a team spirit that's rarely being found in the industry today.

Alpert: Yeah, actually, I'll just second what Giora was saying. I know it felt that way in my career. And some of the memories of the people and the experiences are bigger than life. But also then, just following on what some of the others on the panel here have said. I think that there's also-- we've kind of talked about, oh, there was execution. Then we got the execution right. There's some part about being a chip company, regardless, analog, digital, that you think, "I'm going to develop components. And if I get that component right, then the customers will design it in." And I think it really is different, this kind of architecture choice. It's the difference between the way chip companies think and system companies. And what I remember very distinctly was with that Swordfish technology where we had superscalar and the floating point and cache memory integrated, and managed to get really the compatibility because we had the software, system software as well. You guys had me flying all over Europe for about six months talking to customers. You know, to system manufacturers. And what I heard consistently was, "Boy this is a wonderful technology. Right now we're really confused about this RISC versus CISC transition. If either Intel or Motorola were offering it to us, we would use it." And the reason was not-- the silicon quality. What they said very clearly to me is that their number one criterion for selecting an architecture was the installed base of application software. And when that's what you need, you either get it kind of by luck, like Intel did, or maybe somebody like Apple can go out and make a substantial amount. But that was

really, by that point, what was the difference. And I did move over to Intel, and a lot of wonderful things about Intel. But Intel didn't understand that at the time either. They had an 860 program. And they were going out to the same customers and trying to sell both of them. And it was really Compaq that set Intel straight and said, "We care about the software. We want stuff that's going to run the X86." And if you don't make 100% commitment to make sure this is going to happen, they funded NexGen and some other startups. So it really is the software. And all of us that have spent our lives in the chip industry--

Yaron: Fail to see that in the--

Alpert: Yeah, you know.

Sanquini: I think you're absolutely right about the chip industry in the '80s.

Yaron: Today it's a done deal.

Sanquini: I think a lot has changed today, and I think we were at the forefront of it--

Alpert: I think so.

Sanquini: --because we were providing solutions with software back in the '80s. We learned how important that was. And today, you can take any chip company in any of the target markets that are growing today. Take a wireless company, take a chip company like Marvell or Atheros, or any of those companies, and it's the software, it's the drivers, it's what makes their solution work.

Yaron: Interestingly enough, even Intel failed to understand it. They had an 8 up, what is 432 processor. They had--

Alpert: Oh yeah. A 960.

Yaron: A 960.

Alpert: An 860.

Yaron: They had enough products of their own failing to understand that unless you really give several orders of magnitude, the software is the one that prevails and dictates the agenda.

Alpert: Even these hardware manufacturers we're talking about, they have more software engineers than hardware engineers now, because that's really what's-- it's not only necessary, but it's the value of making these solutions work, and sometimes differentiating them.

Sanquini: Yeah, I think it gets back to your point about it was really, really hard to do the things you were trying to do. And any other companies that you've worked with since then, it's really not been that hard.

Yaron: It's a piece of cake in comparison to what we were trying to do.

Freund: And part of the reason it was hard is that we didn't have a roadmap, okay? We started this program, we had to make our own roadmap. We had to figure out how many different colors you needed to color a map so nothing-- the answer's five. But we didn't have a road-- so we had to invent and discover, sometimes a little late, the things that had to be done. Whereas today, it's more cookie cutter, you just know.

Yaron: You are going into an uncharted territory and you keep on walking and writing the book and the next chapter as you walk. It's not like, "Ah, okay, the prior guy did it this way. Okay, we'll modify it a little bit."

Freund: Right.

Yaron: You are really walking into uncharted territory. And the next day is the next day, and you have to find the next day metal without really understanding what's behind that hill when you get to the top. It was difficult.

Sanquini: Given we weren't a major player, I think probably good to move then to the next topic. What contributions did National make to the industry at that time?

Yaron: I can start. In terms of a design methodology, I think we made several contributions in chip design methodology or silicon design methodology. I think we made significant contributions in verification methodology, which is now part of the industry. Tools that are provided by Cadence today which acquired Verisity. And I think we made some significant contributions in the reduction, if you will, but that's something that Bob and Don can probably speak to better than I can, in the ability to reduce VAX 780 into a small tiny chip. All of those were things that we kind of left behind for others to take to the next stage. And maybe you guys can talk about compiler technology and architecture technology when you reduce a VAX 780 into a small piece of silicon.

Freund: Well, just on the compiler side of it, is that I've already mentioned the common intermediate language, that's either interpretable or you can co-generate from gives you an execution environment before you have hardware. And then you can actually test it on your hardware when you do get your hardware. Also, the prior generation of micros, the hardware engineers had this tool they used called an ICE, and in-system or in-circuit emulator.

Yaron: Right.

Freund: Right? And that was a way where you would substitute an instrument, a plug, for the microprocessor that you're emulating, and you could do all kinds of things. We found out, you know, clearly, those things wouldn't work. We had to bring those tools inside the chip, okay, and to bring the kind of breakpoint capabilities that used to be in in-circuit emulators and put that in the silicon, simply because of clock speeds and propagation delays wouldn't allow us to do anything else. And the way that we kind of developed a way of doing international software development-- I was not involved in the hardware side. I'm sure there was a lot of stuff that you had to invent then too. But we didn't have all the nice collaboration tools. We didn't have ways of checking in software. We didn't have CVS really to check and maintain version control and so forth. And we didn't have compiler, we didn't have quality assurance mechanisms established for software at that time, at least not in the chip industry. Now one of the things we did do is we dragged a lot of particularly financial market test techniques into the software testing that we were doing to try to assure we had a higher quality product. But again, that's importing technology and applying it in ways that hadn't been done before. So I think that was a small bit of a contribution.

Bal: I think other than the technology at some soft level, we kind of set a target that 32-bit architectures are real, could be built, and there were reasons for other people to build them. Because the market was there, that was the next thing. So in that sense, we set like a high level road map in some ways you can say. But I think another very important thing, which is somewhat personal also, and I think it probably applies to many people, is we also learned what not to do. You know, things to do of course, and all the hardware skills and software skills you bring to the party that's at the technical level. But just in terms of business you go after and how you go after a business and things that you do and don't do. If there are adversities, you still are prepared to go against the adversities. Those are all kinds of, I think soft skills that are eventually very important in the marketplace. I think we learned a lot from the program.

Alpert: Yeah, I was going to say, I think in terms of the technology, and obviously with my architecture focus, I think that the fact that National was able to get through that decade maintaining the compatibility and still employing these advanced type performance design techniques of superscalar, pipelining, cache memory, and so on. A lot of other companies really struggled and didn't do well through that. You mentioned Intel, we talked about a number of alternative architectures that were proposed. Motorola stumbling on the 88K, Digital stumbling as well. That I think it was a proof that in a sense, the architecture doesn't matter that much. It's a matter of really the design techniques and the software that go around it that give you the performance, that give you the value and the compatibility. And besides that then, I'd say just in general, I think from architectures, a lot of people that learned an awful lot and went on to do other things. I think we've heard Les Kohn's name mentioned. Les, of course, at Intel did the 860 architecture. He also-- I don't know for sure where he is now, but he was Niagra with Sun's multi-threaded architecture. Uri Weisser went on to Intel. He was the head of the MMX architecture development.

Yaron: Intel fellow.

Alpert: Yes, and an Intel fellow. And I just know, even from the younger people that I worked with on the team, there were three of them that got their master's degrees based on work that we were doing in developing those products. So it really was cutting edge. It was both new--

Yaron: Bleeding edge.

Alpert: You know, it was creative and practical. So I think that everyone that passed through that brought a lot of that experience to other areas.

Sanquini: So really a lot of technical contributions that we brought to the party. And I think setting the bar, as you point out, Subhash, pretty high in terms of systems solutions with the chips as early as the 1980s. That, and just driving the software issues that-- the uniqueness of pulling in ICE. I mean, Intel promulgated the ICE concept, and we made it useful inside the chip basically. I think your point about the CEOs and founders of-- I mean, just here in this room, the five of us are CEOs or founders, okay, of companies.

Freund: Or several times. <laughs> It becomes a bad habit, Dick.

Sanquini: If I look at the contributions, there's an awful lot of CEOs in the industry. Just recently, Mosha took over Xilinx.

Yaron: Yeah.

Sanquini: Yeah. And you indicated Yoav Hollander, basically--

Yaron: His own company.

Sanquini: The Verisity founder, Yoav Hollander, invented a random number generator that we used for verification of chips and made an industry product out of that. So he created a company and that company was sold to Cadence and that tool is now propagating throughout the industry, that's just one--

Yaron: You can't imagine today doing high end designs without the methodology that we developed back then.

Sanquini: Good. Okay. If we look at the 32000 individually, what did you get out of working on the 32000 during those years, and what would you like to leave people with?

Alpert: Yeah, there's probably so many things. But you know, I think one at a kind of a personal level for me it was the first time as a manager instead of just a technical contributor. And that was quite eye opening. I wouldn't say I learned to be a good manager at the end, but at least I learned to start to think like a manager. I remember I was sitting there on a Saturday, which of course was a day off in Israel.

Yaron: A day off?

Alpert: Well, it was supposed to be, at least part of the day. And I was sitting there working on one of these specs and I figured, "Oh boy, I wish I didn't have to do this right now. Somebody else can do it."

And they'd given me a req [personnel requisition]. And I suddenly realized, "Whoa, why don't I spend my time trying to hire somebody to do this instead of doing it myself." So that was I think the number one kind of thing that opens up as a manager. And the other thing that I'd say I learned, but maybe not well enough because I repeated some of those mistakes again is, you know, we talked about all of the new things going on, the brand new instruction set. Multiple new chips and so on. I think that I learned for product development, as opposed to pure research, for product development, I think you can't try to do more than one, or at most two, big new things and hope to succeed within any type of schedule, budget, whatever other real world constraints there are to making a product successful.

Sanquini: Giora, how did the 32000 contribute to your career?

Yaron: Well, first of all, after National, I ended up taking a company called Indigo public. Which, it was one of the top ten flyers in Wall Street at the time that we took it public. Took it public as a $1 billion, it traded all the way up $3.3 billion. And since then, I'm starting my own companies, two of which have been sold to Cisco, one has been sold to Conexant. And the largest of them all, Mercury, was sold to HP for $4.9 billion. And really, I came to National after doing my post-doc at Hughes Aircraft. So I probably had several patents, several publications in peer reviewed journals. But something that has to do with taking a concept to a point where you ship a product to a customer and he is eventually willing to sign a check, all of that ended up coming from experience that was gained at National Semi. So a lot of the business experience, if not all of it, I really owe to National. You know, I used to be the young kid coming into a meeting. Now I am the old guy walking into the meeting. So people are kind of looking at me, "So what kind of word of wisdom does this old guy have to contribute here?" A lot of it are concepts that were acquired at National Semi. And it's the kind of stuff they don't really teach you in Harvard Business School, as they say. So it's endless.

Sanquini: Bob, how about you?

Freund: Well, I think one of the important things that I really learned was the importance of intellectual honesty and the importance of being able to say no. And the importance of being able to fight for your ideas, but pick your battles wisely. And I think that's really what stuck with me most of all, and done me very well. So there were incredible conflicts every day. You had all of the usual issues in terms of budget. You had all the issues in terms of geography. Cultural backgrounds, not only national cultures, country of origin cultures. But also company cultures, silicon culture versus software culture battles, sales versus engineering battles. And management versus everybody battles. Those things are really important. And how those battles played out really determined the course of action the course of history. And I think that I could have done more and better at the National days if I had battled a little differently, at least I'd like to think that's true. But I learned how important that is to do today. So I think that's important.

Bal: Okay, so I was an engineer at National before I got into the 32000 program, which is very important in my career, I think, because it allowed me to transition out of engineering into a minefield, actually. But that was the nature of the thing. So it taught a lot of business skills. Because I've never looked back, and the last 25 years have been related to business. Just like Giora, I have many, many start ups, take them out to-- and this effort actually turned out as I look back, was a start up within National. It was really funded by a company called National Semiconductor, but other than that, everything was new. Everything was new. I mean, different types of people. National culturally did not have these people, so many people

were hired from outside. So how a start up works is something that-- or things to do, not to do, all those kinds of things, I think I got out of this effort. And since then, I've been involved in four or five different start ups, an IPO included. Sometimes even in start ups, doing things which sort of go back and say "Hey, this is something that I should focus on." I distinctly remember in one situation we had so many different ideas, and they all made sense. But ultimately, from a marketing perspective, I put my foot down and said "No, we're going to do only one, and we're not going to do five." Because there were different ideas. Everybody said we could do it. I think that really went back to something as simple as that. That is, just focus on something that you really believe in. Of course, there are other skills involved too, but that was one thing.

I think I learned a lot about strategic selling. Everything that happened after that were all related to strategic selling. There was nothing that we won later on which was strictly on the basis of technology or technical goodness. It needed involvement of the whole company. And it formed a relationship with other companies, particularly start up going to sell into another big company that was extremely important. So to be able to show the commitment and have relationship from top to bottom, that type of strategic selling is something that we learned. And in addition to that, like Bob said, also gave opportunity to work around the globe actually. I was very lucky at that time in National, where I was one of the few people who could go around essentially anywhere. We went to all kinds of countries that could use these kinds of products. And that was fun to be global at that time. So that was very enjoyable. As a matter of fact, I remember at one time I was in Germany for pretty close to two months just to get my counterpart in Germany educated. And he was a computer scientist, so he knew it all before he joined the company. But he needed to be educated in what really matters. Because I had learned that out here for two years already into it. So just that type of interaction, working with different types of people. Engineers in one country, customers in another country. Europe in particular at that time was a very popular destination for us. And of course, all the issues needed to be dealt with in the United States. So world of experience.

Sanquini: Boy, I can empathize with all of you. I'd have to add that I really learned how to run a high tech business. I mean the--

Yaron: A global high tech business.

Sanquini: The 32000-- and previous to that, I had run analog businesses at RCA and had run the microprocessor, actually it was a microcontroller business, 1802, at RCA. That was nothing compared to the 32000. We stressed every vector of a business, whether it was design, CAD tools, we were at the bleeding edge. If it was manufacturing, we were at the tightest feature size. If it was software, we had to have bug free tools, and how to get near bug free tools. If it was architecture, you know, partitioning, in addition to super scaler, we also had to partition the solution. And in doing all of this, which are all difficult technical details, how to really manage cash.

Yaron: Minor issue.

Sanquini: And National gave us that opportunity. I mean, because we had in our group, if you remember, the 32000. And we had microcontrollers, and a thing called the UART, okay? And that thing called the UART and the microcontrollers funded. Because Charlie Spork ran the company profitably. He would say, "Dick, its your business, make a ten percent profit. Okay, and if you want to invest that in 32000, you

can invest it in 32000." At the end of the day, those three groups, UARTS, Microcontrollers, and the 32000 must to make a profit. And we did that. Believe it or not, even with all the heavy 32000 investment, it drove the corporation nuts because they always wanted to pull out the 32000 and leave it basking in its glorious red ink. But that experience really helped me in all the other businesses I've ever managed, including a number of startups. So that it really does implant a entrepreneurial DNA into you as you go forward. Because National and basically that 32000 program allowed us all to learn an awful lot.

Don, what would you like to leave an engineer with? What thoughts do you have for today's engineer that you'd like to leave with them?

Alpert: But I think what I usually do, and this certainly applies in this case. You know, it was just an odd, rare set of circumstances that got me over to National. I got in microprocessors because I couldn't get a job with mainframes in 1980 at Amdahl. I worked at Zilog and then I was moving over to Israel and just happened to hook up with National with Uri and with Giora. And it was just other things going on in my life that led to those opportunities. And just take advantage of whatever comes along and make the most of it. What you learn and the people that you meet. The rest of your career will take care of itself.

Sanquini: How about you, Giora?

Yaron: Well, I'll say two things. Rule number one is, before heading for a mission, try in a proper fashion size there are enough resources that you have to allocate to execute the mission. Rule number one. Rule number two, once you have embarked the mission, never take no for an answer. And the number of times that we got into a brick wall or what we thought at the time was a brick wall, endless number of times we got into that situation. We refused to take no for answer. And one way or another, we found a solution. As I mentioned before, it would have been in design methodology and compiler technology and verification technology, every aspect of the technology. We ran into a brick wall, we got no for an answer, we refused to get no for an answer, and eventually came up with a solution. So if there's a moral to the story for the younger generation, we're going to be fading out of history quickly, by the color of the hair at least, is that the name of the game is really to make the impossible into something that's possible. And I'd like to think that we might have maybe written a small chapter, if you will, in the history of the silicon valley, which is known for demand for innovation. I think we've written a chapter in how to run a global organization. You talk to people in Cisco today, everyone is talking about collaboration, meaning we collaborate without all those collaboration tools, just get him on the damn phone at four o'clock in the morning and starting to communicate. So all of those are really starting new chapters in new books. And I'd like to think that the younger generation will keep on writing similar chapters or even more advanced chapters, if you will.

Sanquini: Bob, what would you like to leave--

Freund: Well, I just want to point out that I got a lot done with that software group communicating with a TWX machine. But that was a teletype.

Sanquini: You haven't heard that word in a long times.

Freund: Yeah, a TWX machine. But no, I think that the- if you're young and you think you're smart, a good way of trying to figure out how smart you are is try to find a project which is an impossible project, and try to make it happen anyway. Independently, you know, it's probably not going to happen. Independent of the fact that there's just a lot to invent. Independent of the fact that maybe the environment isn't quite right or the tools aren't there, or it's certainly not going to be-- if there's any element of comfort in the job, you know it's the wrong job. It's the best job for a young engineer, is to do something impossible. And that's I think the thing that was true for me and for many of the other guys on my team.

Sanquini: Subhash?

Bal: So secret sauce, or a killer application, those things are of course very important, innovation included in that. But I've come to believe that it's very important to distinguish very clearly between ultimate vision and what's practical to do today. Nothing wrong, as long as the secret sauce is present right from the beginning, that's the core and around which you can build. A lot of products don't go for everything all at once. It typically-- I mean, if you win, that's great. But I don't think the world really moves that way. It's okay to do it in phases as long as you're ahead of the game. But do keep that secret sauce going. Also, I feel it's very important that whatever ideas you have in terms of implementation, what's going to be going out in the marketplace, you wait it out working with customers and have a team, a good team going with marketing guys, sales guys, whatever it takes for a particular situation. But translating your ideas into an actual thing should indeed be a function of working in the field. Wait it out. Otherwise, you may indeed end up doing something which is brilliant technically, but may not be useful.

Sanquini: I really have just one simple message basically from the 32000 program, and that is that the best technology alone doesn't necessarily win, and that the young engineer today should really focus on, even when he's in the best technology, collaboration and what I call technical credibility backlog building for himself as he networks around. So I think that's the way I'd like to leave it. Very good. I think you guys did a very good job, and I think we've communicated.

Yaron: Great.

Alpert: Thank you Dick, for organizing this.

Yaron: Thank you for the opportunity.

END OF INTERVIEW

