10/00

Carver Mead Interview Part IV
(With Dick Lyon, Schlumberger AI researcher)
By: Gene Youngblood
(Document 4 of 4 of Mead Interview)

DICK: You asked about what it's like to design algorithms when it's not just Pascal programs or something. It's a different game, but people who design algorithms always do it with some target computing technology in mind, whether it's a general purpose computer or full custom silicon or maybe you're designing the interconnect pattern for Carver's chip to make a new instrument model. Whatever your target technology is, that defines the game you're playing when you design algorithms. The neat thing about custom silicon is that you've got a lot of new kinds of games other than just trying new programs. So designing musical instruments is a new kind of game. You come up with algorithms that you could have written in a general purpose programming language but you probably wouldn't have, because if you had a general purpose computer to play with, you would have done something different -- more complicated, probably; more efficient maybe. If you're designing from bare silicon and you have to make a new algorithm, the game is wide open. All you know is you're making it out of submicroscopic transistors. If you're designing within the context of somebody's silicon compiler, you're a little more restricted than that -- you know you can make it out of predefined module types and register types and bus structures and so on. so that defines the rules of the game. A programming language puts a lot of preconceptions in your head about what you should do with a computer.

CARVER: In fact you do things very differently in different programming languages. Not because you couldn't do the same thing in two different languages but because the way you think in one language leads to a way of doing it which is natural in that language. Even though you could have thought in another way it's not as natural. So the language of silicon has certain things that are extremely natural to it. The idea of regularity matches very well the idea that communication is very expensive. If you think of things locally and have them communicate not too globally with too many other things, a): it helps you to think about it and make regular structures, and b): it also makes things that are very high performance. That was the part that the old silicon hackers missed. They didn't understand that a regular design can produce better performance than some irregular way. Because you force your thought processes to encompass the overall way that the thing works.

DICK: You still get questions from people who don't know enough about it who are steeped in the old lore but don't understand very much. They ask things like “If you design a chip in the Mead-Conway structured style, how much performance do you have to give up: 20 percent, 30 percent?” I say that's a silly question. The question is how much performance can you gain. They think of the structure as being an artificial limitation, which it is, but it allows you to do things you couldn't or wouldn't do otherwise. And you get to invent the structure, which means you make your problem easier not only at the bottom level of hacking transistors but at the next level up -- you can do creative design and algorithm planning at a higher level and make it easy to build the whole thing and make it efficient. Instead of doing it the old way where you draw many pages of schematics and then have somebody try to force it into the chip.

GENE: Why can't we get both VLSI and high speed at the same time?

CARVER: In 1956 when I was a student it was thought that if you make a bigger transistor it always got slower. That was the common experience of the day. So this one company said if I make a tiny transistor I can make it faster, and if I make a bunch of these little transistors and put them all in parallel, why isn't that faster? Well it is faster. Because all the parasitics that loaded down the big transistors aren't a property of the individual transistor; the real problem was that people were scaling things wrong to make the big transistors. They were thinking about it wrong. And these guys had figured it out. They asked me to prove scientifically that you don't have to get something that goes slower and takes more power if you make it bigger. So I analyzed what was actually going on there and that's actually what got me started thinking on this whole train of scale -- the very first consulting job I ever had. It turns out that it's the same kind of misconception when people assume something is constant that isn't when you scale.

GENE: If that's the case, then, can we put a number on the speed limit of the Von Neumann computer?

CARVER: We can do that very well. Let's do a one-chip machine. Things are scaling in such a way that the power/delay product is getting better by something between the square and the cube of the scaling factor. As you get close to the limits you don't get the whole cube law any more. So if you take the minimum-size transistors -- quarter-micron -- you have gate delay times in the 10 picosecond range. People have already made circuits with 30 picoseconds and the latest I heard was 18 picoseconds in submicron CMOS. And it's going to get better. So the speed limit of one isolated gate not driving anything will be in the 10 picosecond range; see, the problem with the Von Neumann architecture is they have these damn long wires you have to drive, which is inherent in the large RAM configuration -- so you need either long wires or many stages of delays. So you can say, well, what do I have to do to drive that thing? Well, the wires in terms of the technology underneath, are getting longer because you still want the wire to go clear across the chip because you're building a bigger machine; on the other hand the transistor itself is getting better able to drive, because the channel lengths are shorter and the submicron transistors are stronger for a given length. So when you add it all up, the actual speed gets better about linearly with the scaling. So from where we are now in device size -- say 1.25 micron -- that's a factor of five from the .25-micron technology, and it runs at about 20 MHz. So you can look forward -- if you're going to build a microcomputer that fills a chip at a quarter-micron and you have good processing and you've done things sensibly, you can look forward to in excess of 100 MHz clock rates for those things. That's already respectable by the standards of people like Cray. And you can look forward to the same kind of complexities you have on those kinds of machines. Right now you have complexities like you have on VAXes on one chip, there's no reason you can't have a complexity on the order of a Cray on one chip by that time. And the power per unit area would be about the same as it is today.

GENE: So the ultimate speed of the Von Neumann machine is about ten times what it is today, that is, ten times a Cray?

CARVER: No. You'd be just about getting in the range on one chip where Cray is today by the time you do all that. The speed limit on a single chip is about a Cray. That's a good number to keep in your head: you'll get about one Cray on a chip, including the clock rate and everything, by the time you push ordinary MOS devices as far as they're going to go. That is, if you choose to make a Cray. Which is an elaborated or extended version of the Von Neumann architecture with vector processing. So we're talking between 50 and 100 megaflops depending on how well its done.

GENE: And how much speedup will we get from new materials like gallium arsenide? A factor of ten more?

CARVER: I doubt it. First, fundamentally you can't make it as small as silicon. The reason gallium arsenide is fast is because the electron mobility is high. The reason the mobility is high is because the effective mass is low. This means the tunneling distances are longer. There's a direct relationship between effective mass and how far you can tunnel. Since tunneling is one of the things that restricts how small you can make devices, to the extent that that's the limiting factor -- and it does enter in submicron devices -- you'll bump against it sooner with gallium arsenide than you will with silicon. Exactly how all that comes out is not known right now. Nobody has yet done it. Meanwhile people are working on some very fancy things called superlattice structures which give extremely high mobilities. But exactly how the device structures are going to scale is something we need to know. These are compound layers whose lattices match and you can make the bandgap go up and down by a sort of stairstep thing. That makes a quantum effect on the electrons. It basically shapes the bandgap in a nice way and already McGill has worked out a theory for that and the measured shift in the photo-emission has turned out to agree exactly with his theory. That would be the HEMT device. That may get you the factor of ten or even more in terms of raw speed. But you can't take the numbers for individual devices and apply them to large-scale circuits because. For instance what's wrong with gallium arsenide technology right now is not the individual devices. They work better than you would have thought. They're better than my first predictions, because there's a fluke, there's a violation of Murphy's Law that happens. When an electron goes in under the gate you'd think it would come up to saturated velocity -- which is only a little better in gallium arsenide than in silicon -- but it turns out they do a ballistic overshoot, so they actually, for short channelings, go faster than the saturated velocity because they haven't started to have enough collisions yet. That was a violation of Murphy's Law of the first order. It was wonderful. The real problem with that kind of technology is that there's only one kind of device -- a depletion mode device. So you have to have the gate voltage of opposite polarity from the drain voltage, which means you have to level shift between every stage. It's horrible, just like the old vacuum tube things. And it turns out that ends up taking more space than the circuit by a lot. So it

isn't at all clear that the densities will ever be competitive with what you can do in silicon.

DICK: But for that same reason it would probably make possible very high density, very fast memories. And you might still be able to get a Cray-size Von Neumann machine on a chip because the memory is where most of it goes. In fact as you everything up and the amount of global wiring gets more and more dominant, the amount of memory gets bigger and bigger, the power of the arithmetic processor goes up a little bit and that's it. So the biggest thing goes into more wire, the next biggest thing goes into more memory, and the least gain is in the processor. This is what people have been doing. When you grow a Von Neumann machine in any technology that's what happens whether it's on a chip or not. Cray is a departure from that because they figured out how to put a lot more in the processing; but it's less Von Neumann-like than other machines.

CARVER: Cray is one of the people who from day one realized that, well, he said two things. He said computer design has two problems. One is how you get the heat out. The second is how you keep the thickness of the wiring mat within bounds. He thought the key to his design was getting the heat out. Well we can take care of making more complex things with less heat by just scaling down the silicon. So that part isn't the key; but the other part is, the wiring management. And he was the only major computer designer that I know of who specifically twenty years ago identified the thickness of the wiring mat as the dominant thing about computer design. Everybody else was saying well you have to have the x-y-z instruction and there's this and that, and he was saying no, it's the thickness of that wiring mat.

GENE: Will there ever be a Von Neumann machine with a one nanosecond clock.

CARVER: There might be. We already got you to under ten nano-seconds with our little chip at a quarter-micron. You could do it today with a small enough Von Neumann machine. A real simple machine. You'd come close to that today using submicron CMOS. They've already demonstrated .8-micron CMOS with 35-picosecond gates. You could no doubt make a credible one-nanosecond small Von Neumann machine. If it was only eight bits wide and had only three registers. Or if you built it like Cray did and pipeline the hell out of it. And we could certainly make these bit-serial things to run that fast. That's our music chips and Dick's speech chip, where you do one bit at a time. Because the signals only have to go to the next stage instead of going clear across the chip.

GENE: Is RISC architecture relevant only to Von Neumann architecture?

CARVER: Yes, and what it really means is something they did early on and forgot about later. People used to build all machines as reduced instruction sets because they couldn't afford anything else. The best example of that was the old Data General Nova, which had a real simple instruction set. Ever since then they built in ever more elaborate instruction sets, believing that it was going to make their computers run faster. And it turns out that what limits your computer is the load instruction and the store instruction and the index calculations. If you do those fast it doesn't much matter what else you do. And it doesn't at all matter how fast you do arithmetic. So people finally went back and rediscovered the fact that had been known for a long time by good computer designers: that it doesn't really much matter what you do with instructions if you get the basic ones real fast. So they said why put in all the others, why not just make them up out of these primitives? You have to have a compiler that's smart enough to do a decent job of that.

DICK: The other argument is that the machines with complicated instruction sets don't really get any advantage from them unless you have a compiler that's smart enough to figure out when to use complicated instructions. With the RISC you may get better performance even with a very simple compiler.

CARVER: But in fact nobody has yet done the real experiment of actually making a complete system with a complete operating system and everything on one of these RISC things and actually honest-to-god putting it side by side with some other computer and given it the smoke test. That'll no doubt happen but it's incredibly late in coming, because the figures people quote are concocted examples rather than real honest-to-god things happening.

GENE: Surely there will always be a supercomputer, because there'll always be the ten-million-dollar machine.

CARVER: Oh but you know what it's going to be? It's going to be one of these chips that has the Cray on it and a

bunch of memory and discs and all that, which will eat up a whole lot of money, and then it's going to be a box that has one kind of special purpose chip on it which does the signal processing things, and another card that has another kind of chip which does the geometry calculations, and another card with chips that do whatever else. When you talk about extending the instruction set...it's like right now when you buy a machine you get an arithmetic accelerator and you plug it in and it runs ten times as fast doing floating-point. So you'll have something that runs ten-thousand times as fast doing signal processing. And something that runs a million times as fast doing graphics rendering. And something that runs a hundred-thousand times as fast doing whatever else you're doing. There'll be these cards which are these accelerators for special classes of problems. And they'll plug into this thing. There will always be card cages and power supplies and fans (or cryogenic cooling systems) but the thing you plug in there is going to have a god-awful amount of capability compared with the little poor wizened Cray that's sitting on that one chip.

DICK: The interesting problem is to figure out how there will be ten-million-dollar machines. I mean, what part of that are going to spend your money on. I believe there will still be such machines but they'll probably be dominated by head-per-track discs or something like that. You'll spend nine out of the ten million dollars getting enough secondary storage and i/o bandwidth that's fast enough to keep up with the thing. Any one of our little chips can generate data a god-awful amount faster than any disc can soak it up. We solve the problem by just putting it out on a loudspeaker. We never store samples at all, ever; and it may be that for some applications you never would need to store all that data.

DICK: In the speech analysis problem, you come in with some sampled speech, which may be at a reasonably low rate like you get off a telephone system, maybe better quality. In the intermediate calculations you end up with places in your block diagram between the first block and the second block where the data rate may be 100 times higher than what you started out with. There's no way you're going to feed that back into your general purpose computer or put it on disc or anything else. But if you put specialized stuff out there to keep dealing with it until you've gotten close to an answer, until you've narrowed it down to what word was said, then you can throw away all that intermediate data. it's a tremendous amount of data but you don't need to keep it. You just look at it to extract the next stuff from it and you throw it away. Put it in the bit-bucket.

GENE: What you guys are doing is amazing.

DICK: It's happening so fast that even new engineering graduates aren't aware of the half of it.

GENE: Any given application always has a limiting case. In imaging, you don't need to compute more than what your display device can show.

CARVER: There's no point in making a picture faster than real time, any more than there is in me making samples faster than real time.

GENE: Alvy Ray Smith thinks you need a complexity of 80 million polygons per frame for photoreal natural phenomena.

CARVER: That's a Jim Kajiya measure: a complex scene has more than one polygon per pixel. Actually it turns out there are some very interesting problems associated with scenes that are much more complex than one polygon per pixel. Which is the kind our guys deal with.

GENE: So let's say we reach some arbitrary point beyond which no more computation power is needed, then you're saying that it's probably going to be possible to put that kind of power on a board and that's that.

CARVER: You just use it until you get a better idea of how to do it, better algorithms. The real interesting problem now is the origin of life problem, right? It's the thing I talked about with the "Newmath" company. They're working the origin of life problem. They want to design silicon algorithms so they're building things that'll help them do that -- and by the way they can sell that too, because other people will want to make silicon algorithms too -- people who want to make better graphics algorithms, who's going to help them? Because you've got to figure out if your ideas are right before you go and make the better graphics algorithm. So there's always going to be room for a thing that's more general than the specific thing because somebody has to invent the next one. So you'll need a development system that's an engineering work- station for people doing computer graphics, or speech or music -- which is not

really a general purpose computer but it more general than the final silicon algorithms it will produce.

DICK: This is exactly the idea that motivated my architecture: make a machine much more general than what I wanted to do. So that I could experiment with algorithms on that machine. So that the next time around I'll know what I want to do and I can put it in silicon.

GENE: So say we've got the limits of what we need to do on one card of specialized chips, and I've got my general purpose computer controlling it. So you're saying that at that point a supercomputer would be a system like that but one which would do all these things.

CARVER: And a lot of them. So that your instructions, instead of being a load instruction or a store instruction -- you still have those -- but what you really do is you say go Fourrier-transform this, go render this display list on the screen with the ray-tracing board, go analyze this speech with this other board.

GENE: But in fact a network in which those functions are distributed, each node with a particular specialty, would be equivalent to the supercomputer. You could synthesize a supercomputer by having small processors on a network.

CARVER: I think that thing you just said is going to replace the supercomputer as a concept: you have servers on a network. You have a server that does the Fourier analysis and a server that does the graphics rendering and so on, instead of thinking of it as being in one box. Because then it's just a much more convenient setup. So in fact I don't see any great big blue boxes with giant power supplies within a few years. I just don't see them at all. Because you can buy something that's got a thousand times the capability in one of these little boxes you hook on a network. That makes so much more sense.

GENE: So bandwidth is emerging as the major bottleneck.

CARVER: It always has been. It's a major opportunity that bigger bandwidths are available now. Fibers not only carry more, they go longer distances. The networks you're likely to find in a medium-sized company or research lab is the network that runs down the hall to the room where you've got all your specialized machines. So you can use this powerful machine from your own office. That kind of net- work is providing a lot of power to the people and you don't have to go out over the phone company to do any of that. That's a decision that can be made by a small company, not a big company. So what we're finding is that all of the definitions of these things are being done by the people who are building the stuff. And the government agencies and AT&T come along later and say, "Oh, gee! We didn't say it was OK." Well you didn't have to say it was OK, thank god; somebody got a chance to do it anyway. If we were waiting around for a network standard to emerge nobody would have any networks right now. In fact we've had networks since the mid-70s because people just went off and did them. When you leave the building, that's when the politics start. It's horrible.

GENE: Where are we now in speech recognition in relation to something that could be put into most user interfaces? Is this technology we've just been talking about leading to a solution of the user-independent recognition device within ten years.

DICK: Certainly within ten years there's going to be a lot happening. It's a dangerous field to make predictions about. If you look ten or twenty years ago people were making very optimistic predictions and not doing the right things. What's happened during that time is that, in many cases people have pretended that the problem is technology limited. So every time there's an advance in IC technology they make the next better thing, but it's still no good.

CARVER: It's just like microcomputers. The idea stream has not gotten better as fast as the technology has. So they make a much faster thing that's still lousy because it doesn't have any new ideas in it. That has happened a lot. Just like the microprocessor is an image of the ugly old computers of yesteryear.

DICK: When you record with just one microphone you lose a lot of information. So while your two ears don't have any trouble hearing me, this one microphone may have more difficulty. This is one area where there's a clear direction that's needed in the speech recognition business -- things that listen to people talking need to have two

microphones at least. They don't. None of them do. Because people haven't figured out how to take advantage of that. That's one of the things I'm working on.

CARVER: That's the stupidest most zero-order thing you can imagine, but it still hasn't been done to this day.

DICK: And one excuse they make is that it would cost twice as much silicon to analyze signals from two microphones. That's a totally lame excuse, because if you can make the problem work by throwing silicon at it you ought to. Because nothing else is working. But it's not that simple. You've got to have the ideas of what to do with the silicon other than put more memory in it and recognize more words poorly -- which is what people are doing.

The key thing that's motivating my own research is to figure out how people do it and model the human hearing system. That approach is not far enough along yet to show demonstrated good results, but it is far enough along to be extremely optimistic about it.

GENE: Not far enough on the biology side?

DICK: No, in my actual implementation of ideas. I can't yet show you a system that achieves spectacular performance. But I can show preliminary results that look encouraging.

GENE: Do we know how people do it?

DICK: We know a lot about how people do it, but there are still some open questions. There's an area that's not receiving enough research attention. There's a lot of research in the physiology of the hearing system and psychophysics, that kind of thing, but not enough on the computational approach -- trying to figure out what are the algorithms that the human is doing.

CARVER: In fact, work like the people here are doing is very much in the very near future, it will stimulate a lot of biology work because it will raise questions that have not been addressed by the biology community. Because they're not trying to build a system and unless you try to build it you simply don't get faced with those questions. So I think there's going to be a synergy between people like Dick and the biologists and the psycho-acoustic and psycho-vision people that is going to really...in fact both of us are spending a lot of our time doing that now because it's just a very gorgeous area that's almost untapped.

DICK: The biologists are starting to understand how important it is to look at the computational aspects, so we've formed some beginnings of some important collaborations with a couple of biologists.

CARVER: After all, there's this working computer that does this wonderful thing, that people haven't analyzed as a computer.

DICK: How do you compare how a microprocessor is put together to the way Carver's chip is put together. The best thing that works really well is a chip photo. Just show a picture of the chips. The difference between a regular design and a non-regular design doesn't need to be explained. It's just there, and strikingly obvious. Carver's chip is simple enough, that in addition to that you could actually write down the entirety of information that it contains. You could write the CIF-code (Caltech Intermediate Form) for the layout in five pages. It's a representation of geometry. You could have basically a printout of the computer text file from which the chip is created. it would be interesting to show the chip, the code that created the chip, and contrast that with the quantities of information that the chip produces. The leverage is the fact that you put a little bit of information into it to make the chip, and the chip is able to produce back huge quantities of information every second, which sound like music. It's an information amplifier. So the pictures and the listing, even though it's entirely unintelligible, would be interesting to see that there aren't too many bits there.

End of Carver 4