The Linley Group

MICROPROCESSOR report

Insightful Analysis of Processor Technology

SUTARDJA SEES SIMPLER SOCS

Marvell CEO Promotes MoChi, FLC to Reduce Design Complexity

By Linley Gwennap (June 1, 2015)

....

Continuing our series of interviews on the future of microprocessor design, we checked in with Dr. Sehat Sutardja, CEO of Marvell. He brings a global perspective to the company, having been raised in Indonesia by Chinese parents. After receiving a PhD in electrical engineering from UC Berkeley, he worked as an analog designer before founding Marvell in 1995, along with his brother Dr. Pantas Sutardja and his wife, Weili Dai.

Despite his business responsibilities, Sehat continues to be active in the company's engineering development and is a named inventor on more than 360 patents. A Fellow of the IEEE, he has received recognition as Inventor of the Year and Entrepreneur of the Year.

Complexity Threatens Moore's Law

The first thing I wanted to talk about is Moore's Law. I hear a lot of different things about whether it is stopping or not stopping (see MPR 4/20/15, "Moore's Law Turn's 50"). What is your view on how it is changing and how it will evolve in the future?

Sehat: I've been in this business since I was 12 years old, and this is no different from what I have seen the last 20 years. Every five years or so, somebody would say, "Moore's Law is dead." Yet the cost per transistor in smaller geometries continues to get cheaper.

But there is a different problem. After decades of increasing at 100% per generation, the cost of a mask set will reach the $10 million mark around 2018. This steep rise in mask costs is taking its toll on the bottom line, including at Marvell. For those of us that can still afford such an astronomically high cost of entry, we face more challenges. For one thing, modern process nodes tend to be very complicated. This causes the cost of developing next-generation products in new process nodes to be prohibitively expensive. Even at the 28nm node, we find that the return on investment has become very small or, in some cases, negative. The reason is that often we cannot sell enough chips of a given design [see Figure 1].

Not moving with Moore's Law is not a good marketing option. So the question you have to ask is, why are you building chips in a smaller geometry? It's because you want to put more things on the chip, not because you want a 0.1mm^2 die. But we are going in the wrong direction. We try to build more-complex SoCs because we can. But as we try to put more and more on the chip, the R&D effort increases rapidly.

So you're saying it's not the transistor cost that's the problem, it's the design cost.

Sehat: Yes. The design cost and the time it takes to finish the job when you have to build all the analog circuits, all the interfaces. By the time you integrate all these functions, you don't have time to do everything else,

Marvell cofounder, chairman, and CEO Dr. Sehat Sutardja. (Source: Marvell)

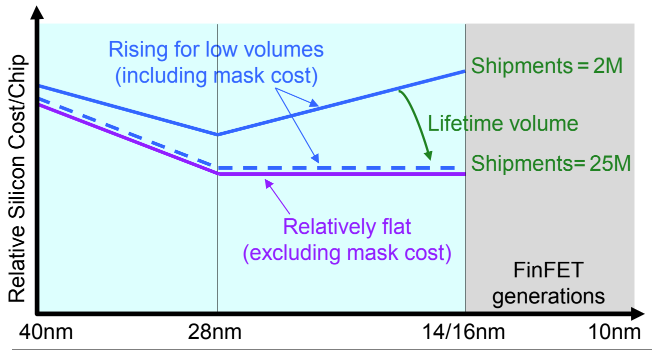

Figure 1. Advanced process nodes reduce profits. As R&D and mask costs continue to rise, building an SoC in FinFET nodes makes sense only if total lifetime volume is large. Cost assumes 80mm^2 die size in 28nm and two full mask sets per project. (Source: Marvell)

unless you have multiple design teams running in parallel. And if you integrate 100 different functions but one fails, you have to respin the silicon. This process is very expensive and delays time to market.

As the chips become more complicated, just pulling together all the IP, verifying the design, and getting it ready for production becomes longer and more expensive.

Sehat: And more importantly, as the task becomes more complex, the engineering teams are afraid to innovate because they don't have time. They just want to integrate. This is bad for business, because we need to innovate. We need to build new things, new technologies, but we don't have time.

MoChi Simplifies SoC Design

You think this drive to integrate, to go to the next node, is not doing us any good?

Sehat: It's allowed us to be lazy, and it's not good for the industry. The solution is not to abandon Moore's Law. The solution is to address this complexity issue so that Moore's Law can continue to flourish. By building chips in modular form, using a technology we call MoChi, we can focus on building the important stuff—things like processors and graphics that need the most advanced process nodes to move ahead.

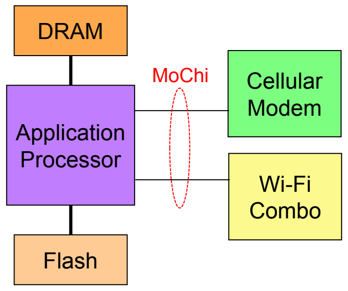

Figure 2. Sample smartphone design using MoChi. The application processor, cellular modem, and Wi-Fi are on separate chips but share access to memory and other resources through the low-latency MoChi interface.

Can you give me an example of how MoChi would work, say, in a smartphone?

Sehat: The vast majority of phones today use a single chip, because it will give you the lowest overall system cost. Why is that? Because when you integrate all the functions in one chip, they can all share resources. All the memory and peripheral functions are shared. If you split the chips—let's say processor and modem into two different chips—you have to duplicate the DRAM or I/O functions. That increases system cost.

My proposal is to build separate chips, but we want all the functions to behave as if they are in one chip [see Figure 2]. They have access to all the resources at once. Any resource that one chip has will be available for other chips. The trick is to extend the internal structure of the SoC through a chip-to-chip interface. From the programmer's point of view, it looks like one chip.

You need a fairly high-bandwidth low-latency interconnect.

Sehat: Right. The access time to get those resources has to be very, very quick. And also, the protocol must be efficient. We cannot use PCI Express, for example. That would severely degrade the performance.

You would still need all of these different components to be designed together by the same company so that they still interact the way they would if they were all on one chip.

Sehat: You bring up a very good point. We need to have standards; no such thing exists today. For the last three years, we've been working internally at Marvell to build that standard so that every chip group will follow one standard. So that every chip they build can talk to any other chip. So there's no conflict of address space, boot sequence, or interrupts. Everybody will follow the same protocol.

When you say a standard, you mean a standard within Marvell.

Sehat: Yes.

But then, do you see MoChi as a way to work with other companies' parts, or is it just a way for Marvell to deliver more value?

Sehat: The first benefits will come Marvell's way. It allows our teams to focus on innovation. They don't have to worry that "I need to have a Gigabit Ethernet PHY in my SATA box." There's a MoChi Ethernet PHY that's already proven; it's in production. Don't touch it, it works. If you want better ones, there's a group that's already working on a better one. No other groups have to suddenly become experts in Wi-Fi or Ethernet or SATA.

But you asked whether MoChi will be available [outside of Marvell]. We're actually making it available to our customers that build their own SoCs [ASICs]. They also find that they don't have all the IP they need, and they don't have time to deal with licensing all the IP. They just want to add value to their system. So now they can build their own MoChi chips that will talk to the other MoChi chips we already have. We're also making it available to partners, like FPGA companies. They could build MoChi-interfaced FPGAs to talk to our silicon.

Dis-Integration Can Reduce Cost

I can see how that works well if your customer wants to build some kind of accelerator. But you were talking earlier about smartphones, particularly the mainstream ones. Isn't this approach going to add cost?

Sehat: To the contrary, it will actually reduce overall cost. It will add a little bit of die area for the interfaces. Everything else stays the same. You could argue, "That will increase the cost, because there's a 1% increase in die size." I say, "Yes, but we won't need the efforts of hundreds of engineers. That's worth a lot more than 1% percent of the silicon." That 1% is in the noise.

What about packaging? If I only have one chip, I've eliminated the cost of the second package.

Sehat: Today's SoCs are very expensive because they have all kind of interfaces. A lot of cell-phone SoCs, high-end ones especially, have 500, 600, or 700 pins. These packages are extremely expensive. If you split it into two chips, each could have a smaller package with fewer pins, and the combined cost is lower.

How much bandwidth can you get through a single MoChi lane?

Sehat: With today's technology, 8Gbps in each direction is trivial. For 99% of the applications, this bandwidth is way more than enough. If someone needs more, we can give them multiple lanes.

Now you've got all these MoChi interfaces driving signals back and forth that would normally be on chip. Is it going to add significantly to the power of the system?

Sehat: The question is valid. The partitioning of the functions must be correct. The CPU, GPU, and other things that require a lot of bandwidth have to be on the same die. Then other functions can be put out on a separate die, like the [cellular] modem or Gigabit Ethernet.

Anyway, most of the time, it is sleeping. When the system's sleeping, this interface automatically goes to sleep, so it uses zero power. Today, most SoC designers don't even power down their internal buses, because it's too hard to do. When they place a node, they don't know what other functions might need that bus. So they just turn on everything. People cannot handle complexity. That wastes a lot of power in today's chips.

Compared to that wasted power, the power used by the MoChi interface is not that much.

Sehat: Yes. We can even throttle it down, but most of the time we don't, because of the automatic power down.

Replacing DRAM With Flash

You have also talked about a new memory architecture called FLC, or final-level cache. What problem are you trying to solve with that technology?

Sehat: Most computers nowadays are used by consumers. How many simultaneous applications can you run at any given time? Just a few. Yet the amount of DRAM that we put in our computers can hold 50 or 100 different applications. The reason is that you and I, the consumers, are accustomed to instant switching time from one application to another.

Right. When I switch to a new application, I want it to start now.

Sehat: Exactly. You don't want to wait two seconds. That is unacceptable to most people. Most users don't care how many mips or flops they have. They just want their system to react quickly. So all these main memories waste so much money and power. Most DRAMs are idle most of the time. Maybe 99.99% of time, 95% of the DRAMs are useless.

They're just holding data, waiting for you to do something with it.

Sehat: Yes. Knowing this is the problem, I came up with this technology called FLC. The idea is to move the main memory from DRAM into flash memory.

This data's dormant, right? It's just sitting there. You don't want to put it all the way back to the hard drive, because it'll take too long to fetch it. So you created this new area where you can put the dormant data and you can quickly retrieve it when needed.

Sehat: Right. But flash is still a thousand times slower than DRAM. So I need some DRAM, but this DRAM has to be very smart. The FLC must figure out which segments of the code or data space need to be in the cache and which ones can stay in the flash. That requires me to change the caching algorithm. Traditional cache algorithms are great for the CPU, but not for main memory. If 90% of time you have a hit, you still have to go to the flash memory 10% of the time. That means 10% of the time, it's a thousand times slower. That's horrible.

Our goal is a 100% hit rate for the application you are running. When you switch, it's no big deal if you miss once in a while, because flash memory can respond in 10 microseconds. No human being in the world will notice that something was delayed 10 microseconds. When people try it, we ask them which one is better. Nobody can tell the difference at all. And that's with only 512MB of DRAM versus 4GB.

How do you implement the algorithm that controls what to put into this cache?

Sehat: It's designed to handle big data sets. It monitors tens of thousands of these large data sets, keeps track of which ones are more recently used, and ranks them. The one you used most recently will rank on top. The one that you least recently used, maybe a browser window that you forgot to close last week, could be in bucket number 10,000.

The stuff in bucket number 10,000 can be purged out to flash if we need the space. What's the chance that something that ranks number 10,000 is going to be needed? Even if we purge it by mistake, so what? You haven't opened this website for three months, and when you do open it, you won't even notice it takes a millisecond to come up.

How much can this approach reduce the amount of DRAM you need?

Sehat: Drastically. I used to say 90%, but that scared the hell out of the DRAM companies, so now I just say 50%. Half of the DRAM in the world will disappear. We're talking $23 billion in savings to the OEMs [global DRAM revenue in 2014 was $46 billion], meaning $40 billion or $60 billion to you and I, because we never pay the OEM price.

But the incremental cost is going to be extra flash memory. How much flash do you need?

Sehat: As much as you want. If you want a system to have 16GB of main memory, you use 16GB of flash.

You're replacing DRAM with flash, but flash is cheaper.

Sehat: Less than one-tenth of the price.

FLC Implementation Issues

Where would the FLC cache controller be? Would it be part of the processor or on a separate chip?

Sehat: This cache controller sits in between the processor subsystem and the memory controller, inside the main SoC [see Figure 3]. Every transaction from inside the chip has to go to this cache, because it is the final-level cache.

So it has to be right there with the CPU. It's almost like the same algorithms you're using in the regular CPU cache. You're just working on much bigger line sizes.

Sehat: Yes, and it has to be fully set associative versus N-way associative. Set conflicts are a killer. Any conflicts will kill the hit rate, and because we're talking about big data, we can easily create conflicts. If you have 16GB of main memory and only 1GB of cache, you will have severe conflicts if you use a CPU cache algorithm.

If I have a device like a smartphone that already has flash memory, can I just allocate part of that existing flash?

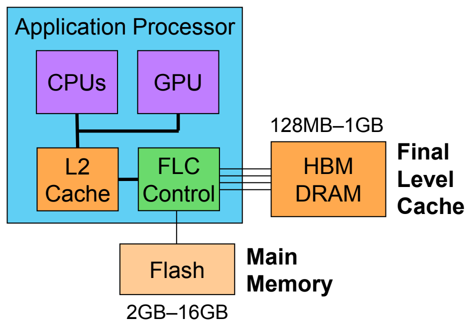

Figure 3. Processor with final-level cache (FLC). The FLC controller uses flash to hold the main memory while keeping the most-recent applications in high-bandwidth memory (HBM) or standard DRAM.

Sehat: That's the goal. Allocate only a fraction of what you have. If you have 32GB of flash, allocate 4GB to the FLC. Nobody will notice. Just call it 28GB of flash; but now you have 4GB of main memory.

Are there any other benefits of FLC?

Sehat: You can potentially get higher performance, because if you use less DRAM, you can put the DRAM next to the CPU. Now you can have much lower latency and much lower power, and you can also build the DRAM to run faster because the connections are so short. Hynix already made a proposal in JEDEC for LPHBM, which is like HBM [high-bandwidth memory] except cut in half. For normal wide-I/O memories, you have I/O in the middle of the DRAM [see *MPR 7/7/14,* "Open Wide to Save Power"]. Our proposal is to cut the DRAM in half, so the I/Os will be on one side. That way, we put the SoC right next to it, and using a simple interposer or high-density wiring, we can get 25GB/s from the DRAM to the processor.

That would be nice.

Sehat: And with FLC, the DRAM is virtualized. Let's say you use Android, and you ask how much main memory you have. It will report whatever you allocate in the flash: 4GB, 8GB. Software does not know that only a fraction of main memory is in the cache. No idea. Just like today's software has no idea what data is in the CPU cache. No difference.

This is actually something that's very hard for people to visualize. I'm no CPU guy; I'm an analog guy. But computers are so crippled, it bothers me. So I spent 20 years on this idea.

Sometimes you need to come from a different perspective to see things that the people who work on it every day don't see.

Sehat: Yeah, I don't have to know how people did it before, because I don't care. I want to build it differently. I just want to solve the problem.

When will we see the first products using MoChi and FLC?

Sehat: We already have engineering samples of FLC. We should have first production to our customers early next year. We are implementing MoChi into products as well, with prototypes planned to be ready by end of year.

Fixing the Design Cost

Most people are worried about transistor cost, but Sehat points out that other costs are rising. Larger transistor budgets increase design costs, and the cost of a mask set is also becoming significant at 20nm and beyond. These fixed costs deter SoC and ASIC designers from using next-generation nodes unless their products have high volume (e.g., smartphone chips) or high prices (e.g., server processors).

For these designers, dis-integration reduces the design cost per chip. Furthermore, a single-function chip might appear in several medium-volume systems, whereas a highly integrated SoC might not generate enough volume to amortize its design costs. MoChi enables several smaller chips to function as a single SoC would. Another approach, however, is to reduce design cost through IP licensing and reuse, enabling efficient use of next-generation nodes even for complex SoCs. The next interview in this series, featuring NetSpeed CEO Sundari Mitra, will address this approach. ♦