A Year of Breakfasts

2018 on Breakfast Bytes

Paul McLellan

Cadence

BREAKFAST BYTES

Edition 1.0

Contents

Introduction

This book is a lightly edited collection of blog posts from Breakfast Bytes during 2018. The true content has not been altered, most of the editing is removing phrases like “last week”, which are obviously no longer relevant.

A post appears every weekday, so there are roughly 250 per year, far too many to include all of them. This is about 10% of what I wrote during the year.

They are not in order of appearance, they are grouped into half-a-dozen of the biggest trends of the year: security, automotive, artificial intelligence, 5nm and EUV, semiconductor in China, and silicon photonics.

I had to decide what to do about links, and I decided just to leave them out rather than put long footnotes that nobody will bother to type. Each section in this book is a post on Breakfast Bytes with the same title, so you can find the originals there.

Obviously, I hope the way that you read Breakfast Bytes is to visit the website every day. It is on the Cadence website, but the easy to remember link is `www.breakfastbytes.com` which will give you a list of the 25 latest posts. But even easier is to subscribe to Sunday Brunch, an email I create each weekend that has a teaser for the five posts from the previous week: the title, the first few lines, and one of the images. If it interests you, then just click on the title to go straight to the post. To subscribe, click on “sign me up” at the end of any Breakfast Bytes blog. If you prefer video, I make a Sunday Brunch video each week too, on the Cadence YouTube channel.

Paul McLellan
San Jose, February 2019

Email: paulm@cadence.com

Chapter 1: Security

Spectre/Meltdown and What It Means for Future Design

At HOT CHIPS, one of the "keynotes" was actually a panel of what I'll call industry luminaries. They were discussing the implications of vulnerabilities such as Spectre, Meltdown, and the recently announced Foreshadow. This is the most important discovery in computer architecture in the last twenty years or so, and will affect how processors are designed forever. Later in the conference, for example, Intel presented their next generation processor, Cascade Lake, and discussed some of the changes they have made as a result. Later in the session, Jon Masters said that Red Hat alone has spent over 10,000 hours on these issues.

I am going to cover the panel in detail. Obviously, it affects processor architects the most. But it affects anyone who uses processors, such as software engineers or SoC designers. Everyone needs to be aware of the implications of this. One takeaway, if this is going to be more than you want to know, is that we don't know how to completely protect against this type of attack without reducing processor performance to a few percent (under 5%) of what it is today.

An Introduction to Speculative Execution

You can do whole advanced Masters-level courses on computer architecture that covers this, so in a few paragraphs, this is going to be the most basic of introductions.

Moore's Law might be limping now, but over the last couple of decades, processor performance improved an enormous amount through a mixture of scaling and architectural innovation. For a decade it was improving at 45% per year. However, off-chip DRAM access did not speed up nearly the same amount. This meant that the processor could execute about 200 clock cycles in the time it took to do a DRAM access. The first solution was to add on-chip cache memory that was much faster. By keeping the frequently used instructions and data in the cache, those 200

cycles could be reduced to a lot fewer. Over time, we went to multi-level caches, with a mixture of small, very fast memories, and larger but no so fast memories. But for this introduction, we don't need to get into those details. We'll assume a fast on-chip cache, and slow off-chip DRAM. In round numbers, a cache access takes 0.5ns whereas a main memory access takes 100ns (hence the 200 cycle number). Most instructions and most data would come out of the fast cache, and so those 200 cycle delays were mostly avoided.

But not all of them. Processor architects realized that the processor could do stuff while it was waiting since often many of the following instructions didn't depend on the value coming from memory, so the processor could get on and execute them anyway. This worked fine for every instruction except conditional branches. When the processor ran into a conditional branch, it could stop and wait for the values coming from memory to arrive, and then discover if the branch would be taken or not. Alternatively, it could take a guess as to whether the branch would be taken, and carry on executing instructions that didn't depend on the values it was awaiting from DRAM. This is known as branch prediction. It is beyond the scope of this little explanation as to how that is implemented, but you win a lot by just following the rule "assume every branch does what it did last time it was executed."

However, there was one big complication. What if the processor guessed wrong? That is why it is called speculative execution since it is guessing whether the branch would be taken, but also doing it in a way that it could clean up after itself if it guessed wrong. After a conditional branch, the instructions are marked as dependent on the branch. If eventually the processor determines that the branch was really taken, then the instructions are retired and the processor moves on. If it turns out that the branch prediction was wrong, then all the instructions that were done speculatively are squashed, and the processor backs up to the conditional branch and starts to execute down the correct branch. To give you an idea how complex this can get, the most advanced processors might get over 200 instructions ahead, guessing that the branch at the end of a loop would be taken and running through the loop many times (before finally discovering that the

loop actually ended many iterations ago, and have to sort out the mess).

This is how all high-performance (so-called out-of-order or OoO) processors have been designed for about the last 20 years. From the programmer's point of view, the processor is executing the program in the order written. The way the processor is built, whether branches are predicted correctly or not, the results are exactly as if the instructions *had* been executed in order like the programmer imagines. Just faster.

For 20 years, nobody saw any problem with any of this. But last year Spectre and Meltdown were discovered. People in the need-to-know groups who had to try and fix these problems knew about it last year. The rest of us found out in the first week of this year. For processor architects, it was not a Happy New Year.

Meltdown is far easier to explain (and fix) so I'll give you a simplified overview of how it works. Let's say you want to read a byte of memory from the operating system that you shouldn't. You train the branch predictor so it will guess wrong that the code I'm about to describe will get executed. Also, you select an area of memory that has never been used so it is not in the cache. Then you do the following: read a byte from the operating system memory, and then use that byte to pick one of 256 locations in the selected area of memory and read the value from there (it doesn't matter what the value is). The processor will soon discover it got the branch wrong and squash all this.

But there is a tiny thing that is different. One of the 256 locations in that selected area of memory is now in the cache (hot), because we read its value. Even though the read was squashed, the cache line is still hot. Since accessing a value in cache is 0.5ns and from DRAM is 100ns, it is not that hard to check the timing of all 256 locations, only one of which will not require 100ns. So we know what was in the byte even though the read itself got squashed, so in a sense "we never read it."

As Paul Kocher said:

These should have been found 15 years ago, not by me, in my spare time, since I quit my job and was at a loose end. This is going to be a decade-long slog.

The Problem

Before going any further, let me emphasize the problem here. This is not a hardware bug in a single processor from a single manufacturer (I'll count Arm as a manufacturer here, although technically they license their designs to the people who actually do the manufacturing). This is a fundamental problem of the way in which processors are designed. Embarrassingly for all the people who work in the area, this is a weakness that has been hiding in plain sight for 15 to 20 years without a single person noticing (well, maybe the NSA and equivalents, who knows?).

Even if you didn't understand my explanation of speculative execution, just take this one fact away. A cache memory access is 0.5ns, and a DRAM access is 100ns. Processor architects use every trick they can come up with to avoid DRAM access, and to find useful things to do during the long delays when they can't avoid it. If we took away these tricks, speculation and caches, then we would have a processor with under 5% of the performance of current processors. No smartphones, no cloud datacenters, and Windows 98 era laptops. *Party like it's 1999* doesn't sound so good in the processor space.

To make things worse, this has arrived as Moore's Law is running out of steam (and processors have hit the power wall too). So we don't even have a 2X factor that we could lose, and win it back with the next node. General purpose processors are simply not getting faster since we've run out of architectural tricks on the architecture side, and Dennard scaling on the semiconductor side.

I'm getting ahead to the panel, but one thing Mark Hill pointed out is that these vulnerabilities are not "bugs" in the sense that the processor does not meet the spec. These processors all met their

spec. The problem is more fundamental still, the way we specify architectures is wrong, since a correct implementation is vulnerable to these side channel attacks.

In the aftermath of the discovery of Spectre and Meltdown, the immediate focus was on how to mitigate the problems with all the processors that were already out in the field. But the next step is to incorporate the knowledge of this type of attack into next-generation architectures. That was the focus of this keynote panel.

The Panel

There were 4 panelists at Hot Chips, chaired by Partha Ranganathan of Google. Each panelist gave a brief introduction, and then they got together as a panel and took questions from the audience.

- John Hennessy, currently Chairman of Alphabet (Google), but one of the inventors of RISC (for which he just shared this year's Turing Award) and co-author of the standard texts on computer architecture (along with his co-Turing-Award-honoree Dave Patterson).
- Paul Turner of Google. Google's Project Zero is one of the groups that discovered these vulnerabilities, and Paul was part of the group tasked with mitigation.
- Jon Masters of Red Hat, the person responsible for fixing up Red Hat's Linux as well as is possible.
- Mark Hill of University of Wisconsin at Madison and also on sabbatical at Google.

John Hennessy: The Era of Security

John kicked off the session pointing out how much the world has changed. There is a lot more personal information online (so we all care more about security). Cloud servers mean that strangers, and even people we might consider adversaries, are sharing the same hardware. Meanwhile the bad guys are getting badder: state actors and cybercriminals are getting more organized and technically adept. Although most attacks are software-based, hardware is now entering the picture.

He gave a brief tutorial on how Spectre and Meltdown work (like mine yesterday). He also talked about NetSpectre, which I hadn't heard of, that allows you to exploit the Spectre v1 hole without running any code, breaking in from a remote machine. It's not a very effective attack, only leaking about 1 bit per minute, but the attack is completely remote.

The big challenge is we can't allow hardware flaws, no matter how much performance could be gained. But it is hard to fix the current flaws and the fixes may cost more than is gained by the hardware optimization. Even next-generation Intel processors probably won't fix Spectre v1 (the hardest of the vulnerabilities to address).

His mea culpa:

> *Lots of us missed this problem for about 10-15 years.*

Paul Turner: The Project Zero Journey

Project Zero is an internal security team founded in 2014 with the goal of reducing the harm caused by attacks on the Internet, with a particular focus on "zero days", which are vulnerabilities that are not known about until the day (day 0) that an adversary attacks using them. Last year, Jann Horn, one of the researchers on Project Zero, discovered this new class of speculative vulnerabilities and, in Google, they became known as SpeckHammer (I think that is a play on speculative execution, and RowHammer, another hardware vulnerability in DRAMs, which is not today's topic).

The CPU tries to hide the big number, the 100ns access to main memory, using caches and speculation. It is very effective, with a low number of cycles per instruction (much less than 1). The flaw in all of this is the assumption that mis-predicted branches have no side-effects. By the definition of the ISA, that is true. But we now know that there this is not true when we look at the bigger picture.

Paul ran through the variants of Spectre, and some of the approaches to mitigation. That's a level too much detail for here. I'll just point out that his slide for "What about Spectre Variant 1"

was blank. There is one attack that nobody has a clue how to prevent without giving up all the gains that come from speculation.

Jon Masters: Exploiting Modern µArchitectures: Software Implications

Next up was Jon Masters of Red Hat. One of the big problems, he said, was that hardware and software people don't talk. In the very old days, pre-IBM/360, there was a much greater understanding (and hardware was simpler). But in the ISA era, there was no clear contract between hardware and software. Programmers assumed sequential execution, which involved various assumptions that were never explicitly clarified. Then we built more layers on top.

It is even worse today, since programming has become much more abstract (Python, Go, Ruby, etc) and many programmers don't even know what a stack or a branch is. Speculation was treated as a magic black box, and the gains were so impressive nobody looked under the hood much. The average programmer has no idea about speculation and out-of-order execution, or branch prediction.

Harold McMillan got re-elected as Britain's Prime Minister in the late 1950s with the catchphrase "You've never had it so good." Jon said something similar:

We are too used to how good we have had it.

Jon's summary:

- The “us” vs “them” became so ingrained we forgot how to collaborate
- Most programmers negatively care about hardware, which is seen as a boring commodity
- Software architects and hardware microarchitects don’t talk ahead of implementing new features, but instead build their view of the world and (maybe) reconcile it afterward
- Previous vertical system model gave way to separate hw/sw companies

- Hardware folks design processors (and interconnects, and other platform pieces)
- Platform-level capability was gradually eroded from outside processor vendors
- The focus on security has actually been a positive from this perspective
- Renaissance in computer architecture brings us a new hope
- Increasing need to understand a vertical stack from hardware to software
- Focus on security has proven the need to understand how hardware works

Modern µArchitectures: Hardware Implications

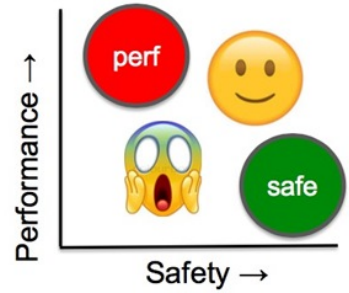
Last of the panelists was Mark Hill. He started with a bit of history:

- Architecture 0.0: (pre-1964) each computer implementation was new, requiring all software to be rewritten (in assembly language, typically).
- Architecture 1.0: (1964 on) the timing independent functional behavior of a computer was captured in an ISA (which would be implemented by more than one design such as the pioneering IBM/360 series), and all microprocessors today.
- Architecture 2.0: what we need next.

The flaws in implementation that Spectre and Meltdown have revealed are not bugs, in the sense that all the affected processors are faithfully implementing their ISA correctly. The flaw is in the 50-year old timing-independent definition of Architecture 1.0. Since leaking protected information can't really be "correct", we need to do two things. First, manage micro-architecture problems like we manage crime, not completely fixing it which would be too expensive. Second, we need to define Architecture 2.0 and change the way we do things.

Some things to consider at the micro-architectural level:

- Isolate branch predictors, BTB, TLBs per process and context switch them. Currently, weird as it seems, branch predictors are shared between all processes, meaning that sometimes it gets the guess wrong due to a different process, which is a trivial problem, but also that one process can train the branch predictor to affect another one, which has turned out to be bad.
- Patition caches among trusted processes (and flush on context switch?)
- Reduce aliasing such as fully-associative caches (use all the bits)
- Hardware protection within a single user address space, such as one browser tab treating another as an enemy
- Undo some speculation where it has minimal performance impact.



Is there a "happy knee" where we get good performance and good safety? Mark fears that there is not. There is a potential to bifurcate, and have cores (or modes) that are fast(er) or safe(r), where some speculation is disabled. This is an extension of what is being done for security, where hardware "enclaves" hold the keys, and perhaps the encryption algorithm implementation. This also plays well with dark silicon, where there is no point in just adding more and more identical cores if we can't turn them all on at once.

But, as Mark pointed out, this is all very esoteric:

> *I'd be just happy if I could stop my Dad executing downloaded code!*

Mark's big point is that we need Architecture 2.0 since Architecture 1.0 is now known to be inadequate to protect information. We need to augment Architecture 1.0 with:

- (Abstraction of) time-visible micro-architecture.
- Bandwidth of known timing channels.

- Enforced limits on user software behavior.

But he admits that none of this seems good enough yet. Another fact of life is the growing use of specialized accelerators such as GPUs, DSPs, neural net processors, and FPGAs. This can actually reduce the need for speculation since the "main" processor is increasingly just doing housekeeping and not running the CPU-intensive algorithms. However, they have timing channels that may be exploitable too. Nobody seems to have looked too hard yet.

Security experts disdain "security by obscurity" in favor of many eyeballs on the code. Only the keys are kept secret. Open source software helps, but even lots of eyeballs on a bad implementation doesn't stop it being bad. Open source hardware is only really getting started, with RISC-V being the most well-known open-source hardware-like thing (it is an ISA, not a hardware implementation, so Architecture 1.0). But as John Hennessy's co-Turing-award honoree, Dave Patterson, said:

> *Most future hardware security ideas will be tried with RISC-V first.*

Discussion

(Note: John is John Hennessy. Jon is Jon Masters).

Question: Who should bear the cost? Today, Intel, Red Hat and Google are paying.

John: Welcome to an industry where the warranty says nothing is guaranteed to work. We have to change how the industry works. As a community, we have acted for functionality over other properties that might be more important. Bill Gates complained to me back in the days when Word still had some competition that people would make checklists and the users would buy the one with the most checks, not the one that worked best. With a processor the first checkbox is how fast it is, not how secure it is. Until a year ago, nobody would have said that they would trade more security for less performance. To be fair, we never asked that question until now.

Mark: We are talking about how to get hardware and software to work in concert, and that will take the next 24 months.

Jon: I'm worried about fatigue. If we get 10 of these per week, we will need to decide which ones to fix, and people will get burned out.

John: This is important, but users accept much greater security issues. People don't create long passwords, different on every system, and change them every month.

Mark: Open source hardware is not a full solution. It is a way to try out security idea and get more eyeballs on it.

Paul: There is so much value-add in the fabrication that it is always going to be secret sauce. it is worth too much money. But it is important to have a spec. The specs today don't address any of this.

John it is good to have an open implementation. In theory you could have an open implementation of an existing ISA, but I don't see that happening for obvious reasons. But with RISC-V people can try things out. You can have a class and get people to implement Meltdown as a teaching tool.

Paul: We need greater isolation but it's a heavy hammer. We need a way to map abstractions at the high level down to abstractions at the low level. Mark the code that is in the sandbox separately from the code running the sandbox.

Question: Better late than never for the era of security. ISA 2.0 first principle could be simple: no access without authorization. It is a challenge for us educators to look at non-quantitative aspects like security.

John: For sure we need to do a better job, but this is not easy. I'm sure a number of you have worked on cache-coherence protocols, and that is really hard. Now think about verifying that you never leak information from a hardware structure. It will require a new set of tools

Mark: I apploud the idea of a simple principle, but that is just what the original architects thought they were doing.

John: Don't dwell too much on caches as side-channels. There are tons of others.

Question: You guys talked about public clouds and paying extra for exclusivity?

Paul: Browsers and cloud providers and operating systems are going to have to find better ways to create more separation.

John: The tree has fallen in the forest and anyone can read it. Isn't the problem that we are not broadcasting who can reference the information?

Jon: The problem is that modern computers share a lot of stuff: the cache, the branch predictors, and so on. These share across boundaries.

Paul: As the user I can control how branches are taken, by training the predictor, but it is impossible for the hardware to know if it was tricked. All it knows is it went down a bad branch.

Question: What about accelerators? Today this has all been about the CPU.

John: Currently accelerators are single-user mode and we currently clear all the state, so that reduces the surface for attack and the rate at which you suck data out. But if these become more pervasive, we'll have to work out how to make them shared, and we'll be back to the problem of having boundaries.

Jon: We don't want to build Spectre accelerators, using FPGAs in the cloud to leak more data faster!

Paul: Northbridge is no longer a separate chip, and so more and more comes under the title of "the CPU".

John: Randomizing page placement, randomizing lots of other stuff, will reduce the bandwidth, but not to zero. But it's like

crime, a temporary fix for now, but not really managing the problem.

On that happy note, the session wrapped up.

Did the Chinese Really Attach Rogue Chips to Apple and Amazon's Motherboards?

Today, Bloomberg's BusinessWeek (BW from now on) published a story *The Big Hack: How China Used a Tiny Chip to Infiltrate US Companies*. The big question is whether they actually did or not. If they did, then this is the most brazen security breach that anyone knows about.

It is worth reading the whole article. Since the article is written by people who don't seem to understand either semiconductors or printed circuit board manufacture, it is hard for me (and probably you) to make your mind up. All the people involved are anonymous people who are supposedly ex-employees of CIA and NSA.

Both Apple and Amazon have denied it in pretty strong terms. Here is Apple's official statement:

> *We are deeply disappointed that in their dealings with us, Bloomberg's reporters have not been open to the possibility that they or their sources might be wrong or misinformed. Our best guess is that they are confusing their story with a previously reported 2016 incident in which we discovered an infected driver on a single Super Micro server in one of our labs. That one-time event was determined to be accidental and not a targeted attack against Apple.*

Apple went further and published an entire rebuttal on their website later in the day. You can read the whole thing. One key paragraph is unequivocal:

> *On this we can be very clear: Apple has never found malicious chips, "hardware manipulations" or vulnerabilities purposely planted in any server. Apple never had any contact with the FBI or any other agency about such an incident. We are not aware of any investigation by the FBI, nor are our contacts in law enforcement.*

Obviously, Apple has been responding to news organizations all day, and the last paragraph says:

> *Finally, in response to questions we have received from other news organizations since Businessweek published its story, we are not under any kind of gag order or other confidentiality obligations.*

Now that's a denial! It never happened, and we are not saying so just because we have been told we have to.

Given all of the facts that came to light as a result of the Snowden disclosures that were previously denied by the NSA, the fact that there is official denial all-round may or may not mean anything. One theory is that the whole thing has been hushed up (too embarrassing?) and made secret, and all the companies are being good citizens and issuing denials as instructed. But the Apple denial in particular goes a long way beyond "no comment" or even "it never happened."

Of course, another theory is the BW got it totally wrong, all the denials are correct. That would imply someone had a motive to create such an elaborate hoax.

What Supposedly Happened?

The basic story is that San Jose company Supermicro makes motherboards for many companies, including (at least in the past) Apple, Amazon, the Department of Defence, The actual assembly of the motherboards is done in China, using a web of subcontractors. The tiny chip was allegedly added to the motherboards, and since it is colored grey it looks more like a surface-mount device. In BW's words:

> *Gray or off-white in color, they looked more like signal conditioning couplers, another common motherboard component, than microchips,*

That was the first version of the hardware hack. BW said that there was an even more sophisticated version:

> *In one case, the malicious chips were thin enough that they'd been embedded between the layers of fiberglass onto which the other components were attached, according to one person who saw pictures of the chips.*

My Opinion

I find the whole story completely implausible. I can believe that it might be possible to sneak a component onto a PCB through a corrupt subcontractor. But for it to do any good, the entire board would have to be re-designed and re-manufactured. The part is small, so it could only connect to very few signals, and those signals would have to all be brought together in a small area of the board. The component is truly tiny (see the picture above from the BB article showing the size against a cent coin). Of course, you can get 100M transistors per square mm, so you can get a lot onto the chip. The problem is getting signals on and off the chip and into the system through the board.

I think it is simple enough to thin a semiconductor die to embed inside a multi-layer PCB. Sony's CMOS image sensor stack thins some of the die to less than 3um. But how would you connect it to enough of the right signals to be useful? Even if you assume, as the article does, that the main function of a chip like that is to allow the hardware to be penetrated and it is the payload so enabled that does the real dirty work, I still don't see how you could do that.

The article blithely assumes that if you can slip a chip onto a motherboard it is simply to fool a Linux system into not requiring passwords using the rogue chip, and only connecting to a handful of signals. It is not enough to connect to them passively (just to listen). But if the chip is doing something active (passing data through and occasionally changing it) then it has to run at speed, all the signal integrity issues need to be addressed, the power supply needs to be clean, and so on.

As Mythbusters used to say "busted".

Extra

One of the few pieces I can find by someone who knows what they are talking about is by security researcher "the grugq" who says:

> *There's not much we can speculate about the modchip because the Bloomberg description of whatever it does is gibberish.*

Most reports are written by journalists who just do the US journalism school thing, where they report what BW said, and then what Apple and Amazon said, and don't attempt to analyze the credibility of any facts, or even try and talk to anyone who might provide any insight.

Despite my feeling that this is complete fiction (or, at best, a dramatic retelling of something that started as true but ended up as "gibberish" after passing through too many people) there is a real problem here. Supply chains might be compromised and there is very little audit that means that something couldn't happen. The "root of trust" for security starts in hardware, which

often means it starts in some semi-anonymous assembly subcontractor in China.

If I was going to attempt an exploit like this, I'd try and hide it on a chip in some gates or IP. When IP blocks are millions of gates, how can you be sure that a few hundred have not been added. I wrote once about a Wally Rhines keynote where he talked about asking "three letter agencies" if they were worried about IP being compromised since verification is all about checking the block does what it is supposed to, and doesn't consider that the block might do stuff it is not supposed to. They apparently laughed, which Wally took to mean that they were doing that, so they assumed the other guys were too.

This story may or may not be fiction. But the basic idea, that the supply chain might be compromised, and we have little protection against it, is not something that is going to go away.

Why You Shouldn't Trust Ken Thompson

I wrote recently about the two exploits Spectre and Meltdown. I think that the most amazing thing about the security weakness exposed is that it has been around for 20 years, in dozens of microprocessors, before coming to light this year. The only equivalent thing that I can remember was when Ken Thompson revealed, in his acceptance for the Turing Award, that "I cannot be trusted."

Unix

Ken Thompson, along with Dennis Ritchie, is the original author of Unix. This is the basis of iOS, MacOS, Linux, Android, and more. So it is the most important operating system ever

developed. It can't have seemed like that at the time, since the reason it got developed at all is that Ken and Dennis "only" had a PDP-11 at Bell Labs and they wanted to get some work done. If you can read C and want to know more about it, I highly recommend trying to find a copy of John Lions' *Commentary on UNIX 6th Edition with Source Code*. It is just what it says, a book containing a full annotated version of the source code of the 6th edition of Unix. He created it to go with his course on operating systems at UNSW, but due to copyright restrictions on the code, it was not allowed to be published and so it was passed along like some sort of samizdat publication. Even today, although the operating system version is obsolete, it is a masterpiece of commentary on a complex program, and it is also a way to see the brilliant way that Ritchie and Thompson did so much with so limited resources. it also created a generation of computer scientists familiar with the UNIX operating system internals.

As a result, Ritchie and Thompson won the 1983 ACM A.M. Turing Award, the rough equivalent of the Nobel Prize for Computer Science. In his acceptance speech/paper, *Reflections on Trusting Trust*, Ken Thompson revealed that (maybe) he had been able to log into any Unix system the world over, despite nothing showing in the source code. There were a lot of Unix systems in the world even then, since AT&T, as part of settling an anti-trust suit, had agreed not to enter the computer marketplace, so it gave Unix away for free since it couldn't sell it. This was before the days of the internet and open source, but was the closest thing to a piece of modern open-source software infrastructure, widely used in academia and, having infected the student population, eventually widely used in industry, too. It is only a slight exaggeration to say that the cloud runs on Unix.

The Unix story itself is for another day. This is a look at what Ken revealed in his acceptance. He says it is not totally original, the basic idea came from an Air Force review of an early version of Multics. The similar sounding name is not a coincidence. Multics was a huge project involving a lot of institutions to develop the ultimate operating system. One of those institutions was Bell Labs, and Ritchie and Thompson worked on it. But AT&T pulled out of it, and Ritchie and Thompson wrote a little operating system for themselves, punned on the Multics name,

and just happened, without any plan, to develop the ultimate operating system anyway.

Ken Thompson Hack

As a result of this lecture over 30 years ago, this is known as the Ken Thompson Hack or KTH. I should point out that Ken, on other occasions, said that they never implemented KTH. But one thing in the security world is that if something can be done, it should be assumed someone has done it. So, for example, recent proclamations that there is no evidence that Spectre and Meltdown have ever been used in the field should be taken with a grain of salt. There would be no way to tell if they have, and a good rule of thumb is that if security researchers discover some weakness, then the NSA (and their equivalents in China, UK, Russia, Israel and more) already knows about it and has exploited it.

If you had control of the Unix source, like Ken did, then one way to make it so you could log into any Unix system would be to add a couple of lines of code to the login command to accept not just the correct password, but also your special magic password. However, that would be really obvious and somebody would soon notice it.

But Ken had control of the C compiler source code, too. It is a bit mind-blowing the first time you come across it, but the compiler for the C programming language is itself written in C. KTH doesn't depend on this though. But a sneakier way to make your magic password work is to leave the source code to the login command unchanged, but fix the compiler so that it notices when it is compiling the login command, and adds your little bit of extra code. This is step one of the hack. If you read the source code of the login command then it is completely correct, and does not contain the security vulnerability.

However, the C compiler contains some obviously wrong code. As he said in his lecture:

> *Such blatant code would not go undetected for long. Even the most casual perusal of the source code of the C compiler would raise suspicions.*

Phase 2 is where it gets really sneaky. In addition to adding code to the C compiler to corrupt the login program, you also add code to the C compiler to corrupt the C compiler itself, by adding both the code to corrupt the login program, and the code to corrupt the C compiler.

When the C compiler compiles the login program, it adds the magic password. When the C compiler compiles its own source code to create a new version of the compiler, it adds both the code to add the magic password, and adds the code to add the code to add the magic password. I know that is a bit complex and hard to get your head around. Ken's acceptance speech goes over the ideas of self-reproducing programs (of which this is sort of example). I remember myself when I went from deciding that a self-reproducing program was "obviously" impossible, to having the aha moment to see how to do it.

But now for the real mind-blowing trick. Once you have created a new version of the C compiler, you go back and take out the code you added. The source code of the login command was already clean, but now, too, the source code of the C compiler is completely clean. And yet, if you make a new version of the login program, it will automatically get your backdoor inserted. Sneakier still, if you make a new version of the C compiler, even though you took the code out, the compiler will insert it anyway.

Implications

As Ken said:

> *The moral is obvious. You can't trust code that you didn't totally create yourself (especially code from companies that employ people like me). No amount of source-level verification or scrutiny will protect you from using untrusted code.*

But that's just Unix. We work in the semiconductor and EDA industries. So the thought experiment is what happens if, instead of corrupting the source code for the login command, you corrupt the source code for a test insertion program? Instead of adding a backdoor password to the login command, if the corrupted tool detects it is adding scan test to a security block, it can add a few

extra gates. Obviously, if you add a million gates to a design it will get noticed. But a few gates in a billion-gate design might well go unnoticed. Nobody has a clue what all those scan test gates do exactly, they just have to make sure the timing is right.

The KTH means that it would be possible to do that not by breaching the security of an IP company, and changing their Verilog, but going upstream to the compiler companies. Once the malicious code is inserted into the compiler (and then removed), the compiler source code is clean, the source code for the test insertion tool is clean, the Verilog for the chip is clean. And yet, there are a few extra gates added to every design to allow the test logic to be used to read out the secret keys (or something equally bad).

Have a nice day!

Passwords: How Even Your Bank Doesn't Know Your PIN

In my predictions for 2018 piece I wrote:

> *The first amazing thing about this is that even when I was studying computer science back in stone age, before PCs and smartphones and the internet, Roger Needham, my lecturer on operating systems told us that passwords should always always always be encrypted with a "one-way function" and not stored in the clear. It shouldn't even be possible to generate a list of passwords for security researchers to study.*

By the way, Roger was a lecturer at the Cambridge Computer Laboratory when I was an undergraduate, went on to become the head of department, then set up Microsoft's UK research laboratory. He died in 2003, young for today at 68.

I realize that probably most people don't know anything about how passwords are stored, and it is quite interesting. So that is today's topic. Before going any further, let me point out that if your job is to store passwords on the internet (or anywhere else) then either you should already know everything I say here, or you need to get professional help from someone who knows what they are doing. I think it goes without saying that you should not make your decisions on security based on something you happened to come across "in a blog on the internet". This is intended to let you know some interesting stuff, and also how to do some limited assessment as to how secure are the systems you use.

Password Database

Somewhere on the server or the computer system, there is a password file. It contains a list of the usernames and a way to verify the passwords (and some other stuff, like whether you are an admin, or what your account number is). On Unix systems, this was traditionally in a file called `/etc/passwd` and this file was readable by anyone. So the first thing that is obvious, is that you can't just store the actual passwords in the file, since then anyone could read the file and find out every user's password. These days the requirement for it to be readable by anyone has been obviated. You can even check this, if you are on a mac. Open a terminal window, you can then type `ls /etc` and you will see a file called `master.passwd` which I assume is the password file for your own computer, but if you try and read it (`cat /etc/master.passwd`) you will get permission denied.

Going back to the original quote from Roger Needham, the passwords should be encrypted with what, in the 1970s, was called a one-way function and is now called a cryptographic hash function. If the passwords were just stored in plain text in the password file, then the login procedure would be something like "take the password typed, and see if it matches the one in the file, and if so let the user log on." With the one-way code, what is stored in the password file (when the user is created or changes their login password) is the password after being passed through the one-way code. So now the login procedure becomes "take the

password typed, run it through the one-way code, and see if that matches the one in the file, and if so let the user log on."

One-way code, or cryptographic hash functions, are ways of taking the actual characters of the password and turning them into a fixed length (but typically long) number, called the hash. The same characters will always give the same hash. Even a tiny change to the characters typed (changing an "a" to a "b") will result in a hash that is completely different. The one-way nature of the hash function is that even if you find out the value of the hash, there is no way to find out the characters of the password (actually, of course, like with those little bicycle combination locks with four digits, you can try them all, which is why systems make you have long passwords and use a rich alphabet that "must contain a letter, a number, and a special character").

Even Your Bank Doesn't Know Your PIN

This is a very important aspect of the way passwords are stored, and worth emphasizing. If you call up your bank and say you forgot your PIN for your debit card, they obviously aren't just going to tell you what it is over the phone. But it is not just that because that would be stupid. The person on the phone with you doesn't have access to your PIN, the computer system won't show it to them. But here is the important part: this is not just because the computer system has been well-written. The bank's computer system doesn't know your PIN either, and has no way to find out. It doesn't need to. When you go to the ATM and type in those numbers, it runs them through the hash function and checks that matches what it has in the database. The one-way function only goes one way.

Cracking Passwords

There are a number of ways that the bad guy can try and crack someone's password. On a computer system or a smartphone, one way to to try and get in is to try all the passwords. Obviously, on a keyboard, this is completely impractical since there are billions of possible passwords. To make it even more difficult, systems start to slow you down once you get several wrong passwords, making you wait ten seconds, then a minute between attempts. On an iPhone, if you have it enabled (which you should), your

phone will actually erase all the data on your phone after 10 failed attempts.

So the only practical way to crack a password is to steal the password file and try passwords offline. There are two main ways to do this. One is known as brute force, meaning try all the passwords in order. So start with "a" then "b" and eventually "aa" and so on. The other is called a dictionary attack. Most people's passwords are very simple, words in the dictionary, often changed a little bit (add a digit on the end, make the first character upper case, change "I' to "!", that sort of thing). There are a lot fewer words in the dictionary (even trying common ways they get changed) than in the exhaustive search. A variation, if your system has been compromised, is to try every word in every file on the system, on the basis that you might have put your bank account (or whatever) password in a file (don't).

In security, one assumption always made is that the security depends only on the keys used in the encryption algorithm. This has been known for a long time and is called Kerchoff's Principle (after the Dutch cryptographer Auguste Kerchoff, who published in 1883—I said it was a long time) or Shannon's Maxim (after Claude Shannon, the legendary information theorist who created information theory). It is thus always assumed that the bad guys know the algorithm being used (which cryptographic hash function) and have managed to get a copy of the password database. That is not to say that you shouldn't keep the algorithm secret, and that you shouldn't make the password file inaccessible to ordinary users. Just that the security of the whole system does not depend on those two things remaining secret. Even if they steal the database, and know which algorithm you use, they *still* should not be able to find out the passwords.

The first defense against this is, unlike most of the time in computing, to make the cryptographic hash function really slow, and expensive (in terms of compute time) to calculate. When you log in to your computer or phone, it really doesn't matter if it takes quarter of a second to check the password, it only has to be done once per login. But for the bad guys, trying all the passwords, this might change the number of passwords they can try in a second from a billion to...four. Of course, the bad guys

can use multiple computers, but even if they use 1000 then they can only try 4,000 passwords per second.

Lookup Tables

There is another approach the bad guys can take and that is to precompute all the passwords, or a big subset of them. They run all the possible passwords through the hash function and then keep all the answers. Now, given the hash of a password, they can see if it is in their table, and if so they know the password. In a sense, at the cost of a huge amount of computation, they have made the "one-way" function (partially) two-way. This is most useful if the bad guys have stolen lots of passwords in either a huge password file, or password files from a lot of systems.

With almost infinitely cheap computation available through the cloud, this becomes feasible, even with a computationally expensive hash function. Plus, just as with Bitcoin mining, if you want it to go really fast you can use GPUs, build a special FPGA encryption engine, or (at least theoretically) a custom chip.

Salt

It is not immediately obvious how to protect against this. At least it took a long time before the idea of a salt was invented. I haven't managed to find out who came up with the concept, and even the name doesn't make much sense ("just add salt").

The lookup table approach works because the same password is always hashed to the same hash. A side-effect of this, by the way, is that even without discovering your password, it is possible to work out if you use the same password on different systems.

What is done is to add "salt" to the start of the password, a random string of characters is added to the start of a password. The random string has to be generated when the password is stored in the password database, and it has to be stored along with the hash. But, and this is the key insight, the salt does not need to be kept secret. The importance of the salt is that it makes building a complete lookup table completely computationally infeasible.

You wouldn't do it this way, but if you added a salt of six decimal digits to the start of each password before encryption, you now have a million different hashes of every password. You now need to have a table a million times as big and it takes a million times as long to calculate it. In the real world, the salt is typically at least as long as the hash, the output of the hash function, usually 256 bits (32 bytes). This is more than the number of particles in the universe, so you are going to have some fundamental problems building that table!

Other Weaknesses

A general rule of being a bad guy in security is not to attack the cryptography. There are other weaknesses.

Social engineering is attacking the people. One famous story I remember, but cannot find online, is a general in the Pentagon during a meeting with a security consultant famous for social engineering (I believe she was called Susan). She had pointed out what I just said, that you don't attack the cryptography, you attack elsewhere. The general pointed out his password was completely unguessable. Susan picked up his business card, dialed the number of his office in LA, and his assistant answered. She said she was an assistant to the person he was visiting in the Pentagon and was with the general, but he'd forgotten his password. Within 30 seconds she had his password. This sort of attack works since we don't expect to be attacked. The general's assistant knew he was at the Pentagon, knew who he was with, and the call probably showed up as coming from the Pentagon. What's to be suspicious of?

Another big weakness is there may be other ways to get into the system. So-called security questions are an obvious one. Working out my long password by brute force is clearly not going to be feasible. But finding out my mother's maiden name? That may be more than a Google query away, but it is not that hard to find out. The town where I was born? That's not exactly a deep secret.

Here's something you might not have thought of if you want to find out people's mother's maiden names (or the name of their first pet, whatever). Set up a juicy website ("secrets of the Hollywood stars" or "daily bargains Amazon doesn't want you to

know") and make people register for it. Then get them to put in some security questions in case they forget their password, like...what is your mother's maiden name?

Two-Factor Authentication

Another way to make passwords more secure is called two-factor authentication. Security is around factors like:

- Something you know (password, mother's maiden name)
- Something you have (smartphone, special dongle)
- Something you are (fingerprint, facial recognition)

Normal passwords just use something you know, so just one factor.

Using two of these is called two-factor authentication. Using an ATM is two-factor: something you have (your ATM card), and something you know (your PIN). Your PIN, as 4 or 6 digits, is nowhere near secure on its own (you can't use it to log in for online banking, for example).

Usually, for two-factor authentication, the password is strengthened by adding the requirement that you have your smartphone and can receive a text message with a special code. On any important system you use, if you can, turn on two-factor authentication.

If you are a Cadence employee, you will know that Cadence has switched to two-factor authentication off the Cadence network. You need to use an app on your smartphone to confirm you are trying to log in. In fact, to use the app requires your fingerprint, too, so I believe it is three-factor authentication.

Red Flags

If you need to reset your password, usually because you have forgotten it, then here are two red flags:

- The procedure to reset your password should be at least as hard as logging on, otherwise the easiest way to get in for the bad guy is to reset your password. In particular, if you need two-factor authentication to log in, you must require

two-factor authentication to change your password too (the bad guy needs not just your mother's maiden name, but also your smartphone).

- If the system emails you your forgotten password, run away fast. Remember how earlier I pointed out that the bank doesn't even know your PIN? If the system can email you your password, that means they store the passwords in plain text and don't use any of the stuff in this book. If someone steals the password database, they have everyone's passwords. Game over.

Fooling Neural Networks

One of the things that I mentioned in passing about models was over-training. This is where the model is made so accurate, often by adding additional parameters, that it matches the training data perfectly. However, on other data it doesn't do as well as it seems it should. This was my experience modeling Scottish Highland deer populations, where more complex models would match the training forests better and better, but at the same time didn't improve (or got worse) against the test forests. Global warming models are similar, and can be made to match the existing data almost arbitrarily close (known as hindcasting) only to diverge increasingly badly once the parameters get locked in place and time passes to find out how accurate the predictions were.

When training neural networks, a similar phenomenon has been observed. In some ways it is more insidious. When adding another parameter to our deer model, we knew we were doing it. When the neural network is being trained and starts to give weight to a new parameter, which is pretty much the same thing, it is invisible.

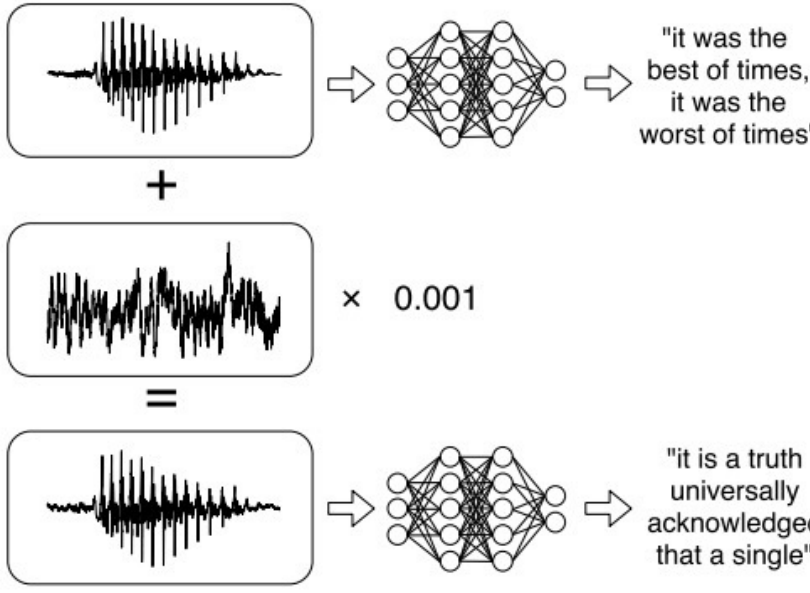
This leads to networks that are not, to use Nassim Taleb's word, anti-fragile.

Sound

I came across a recently (January 5, 2018) published paper *Audio Adversarial Examples: Targeted Attacks on Speech-to-Text* that did something similar with waveforms:

> *We construct targeted audio adversarial examples on automatic speech recognition. Given any audio waveform, we can produce another that is over 99.9% similar, but transcribes as any phrase we choose (at a rate of up to 50 characters per second). We apply our iterative optimization-based attack to Mozilla's implementation DeepSpeech end-to-end, and show it has a 100% success rate. The feasibility of this attack introduces a new domain to study adversarial examples.*

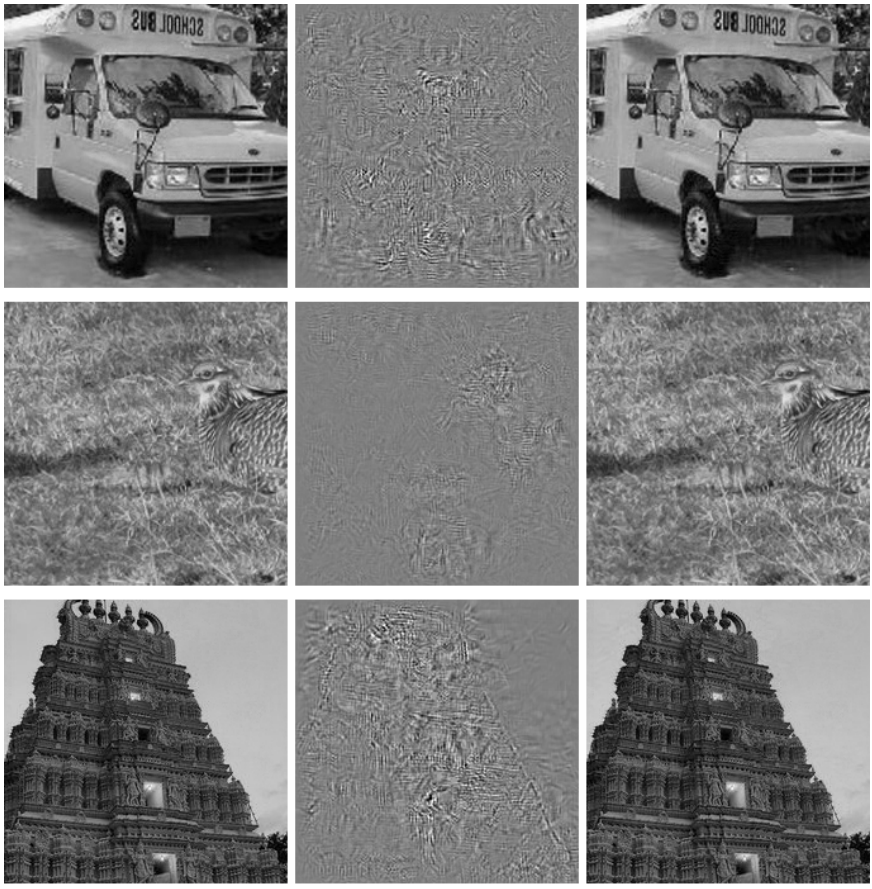
Here is the first figure from the paper, changing "it was the best of times, it was the worst of times" into "it is a truth universally acknowledged that a single" (pop quiz: where are those two quotes from?).



Vision

The first major paper on anti-fragile neural networks in vision was published in 2014 with the rather anodyne title of *Intriguing Properties of Neural Networks*.

The most astounding examples, I think, are the pictures below. The column on the left shows the original pictures, all correctly identified by AlexNet. The column in the middle shows the small amount of carefully constructed noise. When the noise is added to the original pictures, you get the pictures in the right column. These look identical to the original to the human eye, and certainly if we can identify the image in the left column, we can identify the corresponding image in the right column. AlexNet identifies all the images in the right column as ostriches!



Here is another famous example. On the left is Reese Witherspoon. By adding some carefully constructed glasses we get the picture in the middle that looks like...Reese Witherspoon with glasses. But the neural net identifies it as Russel Crowe (on the right).

Okay, those are amusing rather than scary. But, in another paper, a team of researchers from several universities worked out how to fool neural nets in the physical world with just a few pieces of tape:

> *Starting by analyzing the algorithm the vision system uses to classify images, they used a number of different attacks to manipulate signs in order to trick machine learning models into misreading them. For instance, they printed up stickers to trick the vision system an autonomous car would use into reading a stop sign as a 45-mile-per-hour sign, the consequences of which would obviously be very bad in the real world.*

Summary

What is surprising to me is just how little the input data needs to be distorted to cause the neural networks to misidentify things. The stop signs with a few pieces of tape on are clearly just that to a human—a stop sign with a few pieces of tape. The images on the right in the 3x3 grid above look nothing like ostriches. These are not borderlline cases, like the signs in the traffic sign database in fog, for example, where even a human struggles to decide if it is a 30mph or a 50mph speed-limit sign. I've not listened to the sound examples, but I'm sure that they sound normal to the human ear since they have just 0.1% additional noise.

So a bit scary, to say the least!

Pop quiz answers: "It was the best of times..." is the opening sentence of A Tale of Two Cities by Charles Dickens. "It is a truth..." is the opening sentence of Pride and Prejudice by Jane Austen.

A Computer Scientist Takes a Look at Mechanical Security

One of the exhibits at The Tech in San Jose shows you how a cylinder lock wors (and even how lock picks works, since you could use them to open the lock without using a key). I mentioned an academic paper about master keys that caused some pushback from locksmiths. Rather like the Yogi Berra remark that nobody went to some bar anymore because it was too crowded, half the locksmiths attacked the paper as being common knowledge, and half attacked it as letting locksmith secrets out to the bad guys because nobody knew it.

The paper is *Cryptology and Physical Security: Rights Amplification in Master-Keyed Mechanical Locks* by Matt Blaze. The title couches everything in the terms that we use in the computer security world. Rights amplification is doing something like having a password for a Linux system, and then somehow transforming yourself from a normal user to root, the Linux superuser who can do anything. In the same way, this paper

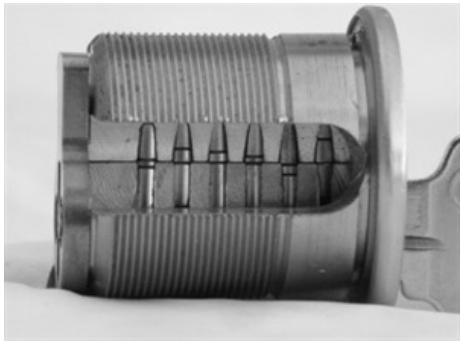
shows how having a key to a single office can allow you to create a master key that will open any office. This assumes that the office building has been keyed for a master key in the usual way, but most are since security (and the janitorial service) doesn't want to carry hundreds of keys.

The paper dates back to 2003 but is still quite important. In their AT&T labs report on the paper, they summarize the situation well:

> *We describe weaknesses in most master-keyed lock systems, such as those used by offices, schools, and businesses as well as by some residential facilities (particularly apartment complexes, dormitories, and condominiums). These weaknesses allow anyone with access to the key to a single lock to create easily the "master" key that opens every lock in the entire system. Creating such a key requires little skill, leaves behind no evidence, and does not entail engaging in recognizably suspicious behavior. The only materials required are a metal file and a small number of blank keys, which for many locks are readily available. Needless to say, the ability for any keyholder to obtain system-wide access represents a serious potential threat to the security of master keyed installations. Individuals and institutions that depend on such locks to protect their safety and property should be aware of these risks and consider alternatives to eliminate or reduce their exposure to this threat.*

How a Cylinder Lock Works

If you look at the above picture, taken in The Tech in San Jose, you can see the internals of a normal cylinder lock. Obviously, a normal lock is not this big, and it has more than four pins. But the basic concept is the same. Those black and grey cylinders are the pins. In a real lock they are spring loaded but in this demonstration, they are just pulled down by gravity. Each "pin" consists of two sub-pins, a grey one at the bottom, and a black one on top. The length of the grey ones is what determines which key will open the lock. The key has to push up each grey pin just enough that the gap between the grey and black cylinders, known as the shear-line, is exactly on the boundary between the lock cylinder and the outer case of the lock. If any pin is pushed up too far, then that grey pin will block the lock turning. If any pin is not pushed up far enough, then its corresponding black pin will still be penetrating the cylinder and prevent it rotating. In the picture above, the girl has put the wrong key in, since the 3rd pin from the left is not lifted high enough to push its black pin fully out of the cylinder.



The Tech also has a set of giant lock picks to show how that works. There are two tools. One is L-shaped and goes in the entrance of the lock and is used to apply pressure (even though the cylinder will not turn more than a tiny bit). The pick itself is inserted in the lock and used to push the pins up. At least one will lock in place due to the tolerances—this is not precision engineering. Keeping the pressure up with the L-shaped tool, that pin will not drop back while you find another that you can work up. Eventually, all the pins align and the lock opens.

If you ever get locked out and get a locksmith to come to your house, they will either open it using tools like that, or with a "bump key." That is a key with all the notches cut to maximum depth. It serves as both tools at once. It is put in the lock, pressure is put on to try and turn the lock, and then the key is hit with something like the handle of a screwdriver. It jumps the pins up and they will catch.

Master Keys

If the building has a master key, then in addition to each lock having a key that opens it, the master key must open it, too. This is done by having a second shear-line in each pin. There are other complications, since there might be sub-masters that only open some locks, but to keep the explanation simple, the assumption is that each lock can be opened by its own key or by the master key. The key that only opens each office lines up to one set of shear-lines, the master key lines up to the second set. It doesn't matter if, for a pin, the shear-lines are at the same point, that pin will just have one shear line and both keys will line it up. If you look at the cutaway photo to the right closely, you can see the multiple shear-lines on each of the pins.

So how do you make a master key? As it happens, I wrote a book (well, about half of it, I never finished) in which a couple of

engineers break into an office using the technique described in Matt's paper. Here's how the conversation in the book took place (it doesn't matter that you have no idea who any of these characters are):

"So how do we find out where the breaks in the pins are?" Kali said. "We can hardly sit outside Yong-Jun's office all day tomorrow filing keys until we get one to open the door. We'd look pretty suspicious, to say the least."

Peter laughed. "We're going to do something easier. We're going to make a master key."

"Surely that's more difficult, not easier. Isn't the master key more secure or something?"

"No, the important thing about a master key is that it is a master key."

"That's a bit too zen for me," said Kali.

"Because it's a master key, it opens all locks in the building."

"Yes, I know that. That's just the definition of a master key. If I wanted to be geeky, I'd say that was a tautology."

Peter was enjoying dragging out the secret. "Since the master key opens all locks in the building, we don't need access to Yong-Jun's office, like we would to make a key that just opens his office. We can make do with any lock in the building. The breaks in the pins for the master key are the same everywhere."

Kali's engineering background finally started to work as she realized the implications. "So we could use the lock for the janitor's closet or something. A lock where no-one can observe us."

"Yes, but we need to use a lock where you already have a normal key. That way we can investigate the pins one at a time. We already know where at least one break is in each

pin since your key opens it. So we make a key with all the pins except one set to the height of the non-master key. Then we file the last notch down gradually until the key turns. We have then found the depth for the master key notch. We make another key, just missing the next pin, and then repeat the process. When we have done every pin, we know exactly how deep to make each notch to make our own master key. Once we make that key, we can open any office in the building. Including Yong-Jun's."

It's surprisingly straightforward to make a master key.

Safes

Having found vulnerabilities in master-keyed locks, Matt Blaze moved on. His paper *Safecracking for the Computer Scientist* gives his review of how safes work, and how to get into a combination lock safe without the combination. However, even using his techniques, physical security tends to be much better than computer security in some ways. As he puts it:

Few weaknesses in physical security admit the kinds of catastrophic failures common in computers and networks, in which a low-risk, low-cost attack can yield a high-value and easily replicated benefit. Even the most sophisticated attacks against safes, whether involving force or lock manipulation, almost always entail at least some risk of exposure.

Google's Titan: How They Stop You Slipping a Bogus Server into Their Datacenter

At the recent HOT CHIPS conference, Scott Johnson of Google talked about some challenges that Google has. There have been stories about hackers infiltrating malware into the supply chain. Given the stories about the NSA intercepting Cisco router

shipments and adding trojan loggers, this is not pure paranoia. As Scott put it, "how do we even know it is our equipment?" The solution is to tag and verify every device. Cloud companies like Google have numbers of servers measured in the millions, so you can't just go round and check them all visually.

Next problem is verifying the boot chain. When a server (or even a smartphone) is powered on, it first runs what is called the primary bootstrap. usually out of ROM (which can't be changed). Its function is to find the real bootstrap, sometimes called the secondary bootstrap or the bootloader. This checks various stuff and then finds the real code for the operating system and transfers control to it. Google worries about whether the bootloader is truly their code, and then whether the operating system code is truly Google's operating system. Remember, Google is not worried about some teenager in their basement, they are worried about national organizations and organized crime. The solution is to sign and verify all boot code.

They rapidly came to the conclusion that they need a silicon root of trust, and built on that they can move up to the datacenter hardware, then to the software infrastructure (operating system etc), and then up to the cloud software. They wanted this to have four important properties:

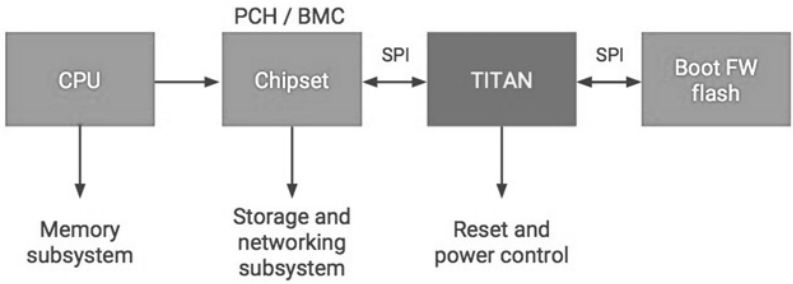
- Every element in the datacenter should be securely identifiable, what Scott calls "cryptographic attestation."
- The first code executed should be cryptographically signed and verified firmware, live-monitored for protection.
- All activities in the datacenter should be monitored and logged in a tamper-resistant manner.
- Own and/or verify every piece of the stack from transistors up to critical firmware.

So they decided to create a chip to do this. In turn, the above requirements led to a set of requirements for the chip itself:

- On-chip verified boot.
- Cryptographic identity and secure manufacturing.
- Boot firmware check and monitor.

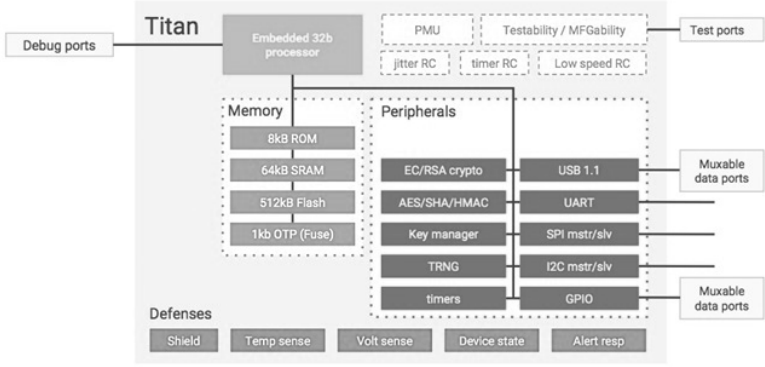
- Silicon physical security.
- Transparent development, full stack.

Titan



The chip they built is called Titan. It sits low down in the system hierarchy as you can see from the above diagram. Titan is a secure low-power microcontroller designed with cloud security as a first-class consideration. But it is more, not just a chip. It also involves a supporting system and security architecture, and a secure manufacturing flow.

Their motivation for doing their own chips was partially that there wasn't anything existing they could use. But also that they wanted complete ownership, auditability, and to build up local expertise in the area and not depend on 3rd party security experts. Also, new attack vectors arrive all the time and so they wanted agility and velocity. If it is their chip, they can respond faster.



The above diagram shows the architecture of the chip.

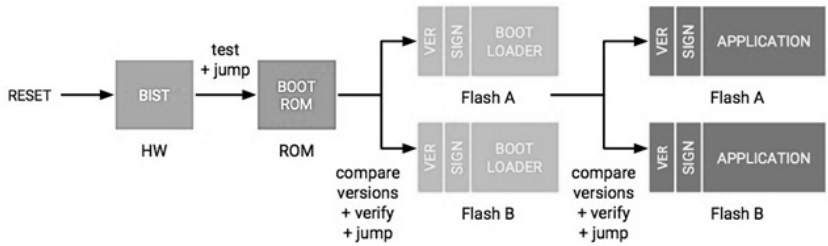
The blue boxes are memory: 32b microcontroller core, boot ROM, flash for instructions and data, SRAM scratchpad, and one-time programmable fuses (more about these later).

The green boxes contain cryptographic acceleration, key management and storage, and (true) random number generator, along with the usual mix of peripherals.

The red boxes are physical defenses, live status checking, and hardware security alert response.

Let's take a look under the hood.

Verified Boot



The verified boot progresses as follows, with each stage verifying the next. There is duplicate flash code so that it can be updated live, and the system is still in good shape if it fails during the update. Code signing is taken seriously, and though it was beyond the scope of this talk, Scott said that there are multiple key holders, offline logs, playbooks for who can do what, when.

The boot works like this:

- LBIST (logic built-in-self-test) and MBIST (memory BIST) are run. If either fails, the system stays in reset. If all is OK, the system jumps to the boot rom.
- The boot ROM compares the two bootloader (BL) version and chooses the most recent.
- The bootloader signature is verified. If that fails, try and verify that other one. If that fails too, freeze.
- Next, the bootloader compares the two firmware (FW) versions and chooses the most recent.
- The FW signature is verified. If that fails, the other one is tried. If that fails too, freeze.

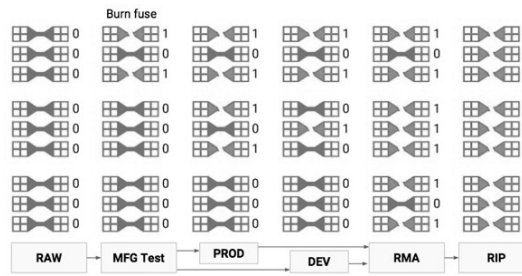
- Execute the successfully verified FW.

Trusted Chip Identity

Trust is established at manufacturing. Each tested device is uniquely identified with an assigned serial number (unique but not secret), and it then generates its own cryptographically strong identity key. This is done using multiple silicon technologies (ROM, fuse, flash, logic) all of which need to be defeated to compromise the chip. This identity is registered in an off-site secure database. Parts are shipped and then put on datacenter devices for production. They are then available for attestation, proof that the servers are Google's. The boot ROM is locked down at tapeout, so it has to be small and bug-free since there is no way to change it.

Life-Cycle Tracking

After manufacturing, there is a continuing need to guarantee authenticity. So Titan is in one of six states, and moves irreversibly from one to another by blowing OTP fuses. The 6 stages are:

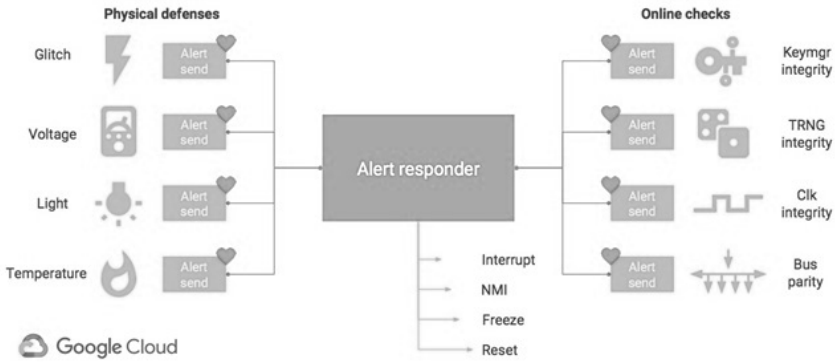


1. Raw: no features enabled, deters wafer theft
2. Test: enables test features only, no production features.
3. Development: enables production features for lab bringup.
4. Production: final production features, no testability, unique keys.
5. RMA: re-enable testability but disable production.
6. RIP: after RMA or manufacturing failure, permanently disable the device.

The above diagram shows the fuses used for each stage. Note that due to the choice of fuses, a given chip can only go from left to right, and a development chip (for playing in the lab) can never be enabled for production.

Physical and Tamper-Resistant Security

Scott admitted that some of this is overkill for a datacenter that is already protected by armed guards. If you manage to get into a datacenter, you are probably not going to use lasers to attack the Titan chips, but they wanted to learn what it would take and, in the future, Titan or similar chips might be used in less secure environments like smartphones.



- Attack detection (power supply glitch, laser, thermal, voltage).
- Fuse, key storage, clock and memory integrity checks. The clocks are generated on-chip, so you can't attack them directly.
- Memory and bus scrambling and protection.
- Register and memory range address protection and locking.
- TRNG (true random number generator) entropy monitoring.
- Boot-time and live status checks.

In the event tampering is detected, Titan responds by one of: an interrupt, a non-maskable interrupt, freezing the system, or performing a full system reset.

Chapter 2: Automotive

In Other News, 100 People Were Killed by Cars Driven by People

You probably heard that a woman was killed in Phoenix by a driverless car. In 2016, 37,461 people were killed on US roads. So if that day was typical (and it probably varies by weekday versus weekend, at least) then about 100 people would have been killed by cars with human drivers.

As I write this, the precise details of what happened are not clear. The police in Phoenix has already said that the accident is probably not the driverless car's fault. Other reports seem to imply it was a complicated junction that the software wasn't prepared for. When I first heard about it, it sounded like a "cyclist" was killed. But it was a homeless woman, with a lot of bags on a bike she was using as transport, at night. She was pushing the bike, not riding it. She was not crossing the road. She apparently stepped into the road so suddenly that the first the safety driver knew of her behavior was when the car hit her. There is video, so you can make your own mind up. Presumably, they will test whether the woman had alcohol or drugs in her blood.

By the way, the NTHSA definition of "alcohol-related" is if anyone, driver, non-driver, or passenger has been drinking or believed to have been drinking. I think they do this since it makes the drunk driving statistics look more dramatic, because the casual reader assumes "alcohol-related" means "alcohol-caused". But if a drunk pedestrian is hit by a sober driver, or a drunk driver is rear-ended by a sober driver, or a passenger has been drinking, it all counts as an alcohol-related accident. NTHSA has been criticized for this since, at the least, it makes the statistics misleading.

It Will Happen

Naturally, Uber has suspended their testing of autonomous vehicles until what happened is clear, and some of the other companies with driverless cars have done the same.

Anyway, my point is not to blame or exonerate the Uber vehicle in this particular case. At some point, someone will be killed by an autonomous vehicle due to a bug or malfunction, whether or not this woman was that accident.

My point is a different one.

On average, 100 people are killed on the roads in the US every day. In round numbers, roughly a million people are killed on the roads around the world per year (1.25 million in 2013 according to the WHO). In the US, this is 12.5 deaths per billion miles driven. The French economist Frédéric Bastiat always discussed the seen and the unseen. When the baker's window is broken, the seen is the work made for the glazier. The unseen is the meat the butcher didn't sell to the baker because he had spent his money on glass. In the same way, the seen are the people killed by autonomous cars. The unseen are the people killed by non-autonomous cars. If all the autonomous vehicle programs are suspended for a period due to the death in Phoenix, I think it is likely that at least one more person will be killed by a human driver due to all the extra miles driven by regular Uber vehicles in the meantime. I've seen an analysis of 9/11 that showed many people were killed in the months after 9/11, when TSA made airline travel very inconvenient, because so many people drove instead of flying. Driving is much more dangerous than flying by any measure. In 2017, there were no commercial passenger jet deaths worldwide, so you can't even work out just how much more dangerous driving is than flying without dividing by zero (it seems there were some non-jet and non-commercial accidents).

If autonomous vehicles reach the point that they "only" kill 10 people per day (scaled up to the number of vehicles on the road), that would be a reduction of 90% in traffic fatalities. If some technology, like magic airbags, could reduce deaths that much, they would immediately be mandated. Twenty years ago, we mandated those extra little red brake lights in the rear window

of cars by law because they were shown to reduce rear-end collisions by 10%, most of which are fender-benders in which nobody is injured, let alone killed.

My point is that if we don't let autonomous vehicles on the road until nobody is killed, then the seen is the people saved by the regulation. The unseen is the 100 people killed every day by normal driving. The US, in particular, is a very litigious society. Except in unusual circumstances, people who cause a fatal accident do not get sued. Partly because it is accepted that some accidents will happen, and partly because even insured people don't have deep enough pockets to pay the sort of multi-million dollar awards that attract trial lawyers. Autonomous cars are not like that, and Waymo and Ford and Tesla risk getting sued since they do have real money. Oh, and it turns out that about half of all fatal accidents are single vehicle, such as driving into a tree or rolling over.

The risk is that lawsuits drive (!) autonomous vehicles off the road, or postpone their introduction by years, condemning thousands of people to death.

Vaccine Court

Something similar happened to the vaccine industry. There is not a lot of money in vaccines since you only need a vaccine once in your life (there are only around 4M kids of any given age). The result was that if one in a million kids had a complication, sued, and won a big settlement, then the entire vaccine industry was not viable. As a result, companies withdrew from providing them. However, like autonomous cars, a lot of the good things about vaccines accrue to other people (people not killed in epidemics, pedestrians not killed by driverless cars). Economists call this a positive externality. Since not having any vaccines for US children would be a bad thing, in 1986 the US set up the National Vaccine Injury Compensation Program, funded by a 75¢ surcharge on every dose of vaccine. There is a "vaccine court" (officially The Office of Special Masters of the U.S. Court of Federal Claims, which sounds more like something out of Game of Thrones) who administers compensation. The thing that makes it all work is that it is mandatory. You cannot sue vaccine makers, you have to go through the compensation scheme.

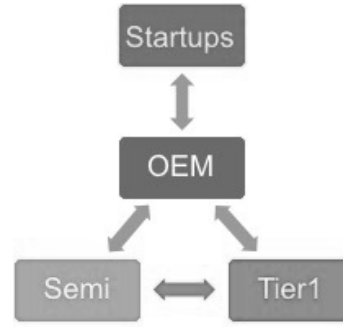
I think that we may need to set up something similar for autonomous vehicles. We want to reduce the nearly 40,000 people killed every year on US roads. If we could instantly reduce it to zero, then there would be no need. But if we can reduce it to 10,000 people per year, that is such an enormous reduction that it should be something to celebrate. But those thousands of people, and thousands of lawsuits, and thousands of multi-million dollar awards would otherwise risk making autonomous vehicles as financially non-viable as vaccines were becoming, and leave us stuck with non-autonomous vehicles. And 40,000 deaths per year.

Any economist will tell you that the problem with positive externalities is that they are undersupplied, it is a form of market failure that should be addressed (negative externalities, such as pollution, are over-supplied, which we typically fix with regulation). Autonomous vehicles are an improvement on what we have once they are safer than humans, even if they are not perfect. We need to make sure that we don't prevent their introduction.

CDNDrive: ISO 26262...Chapter 11

Automotive manufacturers are under some conflicting constraints to deliver vehicles that are both efficient electric vehicles (driven by the requirement to reduce emissions) and, at the same time, safe autonomous vehicles. This is not going to be achieved with a supercomputer in the trunk of every car, it requires advanced-node, low-power SoCs. As it happens, Cadence is a company that can help you there!

It is also breaking down the traditional hierarchy of OEMs, who assemble the vehicle and build the engine, tier-1s, who supply the complete electronic control units (ECUs), and tier-2 and tier-3 who supply components, such as semiconductors, to the tier-1s. The OEMs are not just challenged technically, their business differentiation has historically been based on excellence in engine design ("the ultimate driving machine") but that is going away with electric traction.



Robert went on to point out that connectivity of the vehicle cannot be relied on for autonomous driving. It is annoying to get "buffering…" when you are watching a movie, rather more serious when your car is trying to decide whether light is red or green. This means that the artificial intelligence (AI) that is at the heart of autonomous driving must be done at the edge. Since general purpose processors (CPU) and specialized graphics processors (GPU) are too power-hungry, that means specialized processors optimized for the problem, but still retaining programmability to keep current as technology advances.

One big area of dispute is whether to do local processing at all the sensors (video camers, radar, Lidar etc) or instead feed all the raw data to some super-powerful central processing unit. The problem with doing local processing is that something important might be filtered out, and the problem with the central approach is that the network bandwidth may be too high, and the computational power required may exceed what it is possible to deliver.

In the automotive world, ASIL safety certification is only done on complete systems. So semiconductor IP cannot be "ASIL certified" by definition. Instead, the buzzword is SEooC. A Tenslica processor (or any IP, from Cadence or elsewhere) is a "safety-element-out-of-context."

The high-end Tensilica processors are ASIL-B ready, designed with a process that is ASIL-D compliant against systematic design faults. For ISO 26262 there are safety manuals and certified tool flows. The Tensilica organization also maintains ISO 9001 certification.

ISO 26262...Chapter 11

Next, Robert introduced Steve Williams of the Tensilica IP group, to do a deeper dive into what ASIL-ready processor IP really means, and what is in the new ISO 26262 part 11.

ISO 26262 is a functional safety standard for passenger vehicles (it will soon cover motorcycles too, which have different standards—nobody will come out and say it explicitly, but motorcycles are so inherently dangerous that it doesn't make sense to over-engineer all the electronics). Parts 1-10 address hardware, software, and tools. Part 11 (new! Improved!) addresses how IP suppliers and integrators work together.

Severity Class	Probability Class	Controllability Class		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

Metric	ASIL B	ASIL C	ASIL D
Single Point Failure (SPFM)	>= 90%	>= 97%	>= 99%
Latent Fault Metric (LFM)	>= 60%	>= 80%	>= 90%
Probabilistic Metric for random HW faults (PMHF)	$< 10^{-7}$ per hour	$< 10^{-7}$ per hour	$< 10^{-8}$ per hour

ISO 26262 defines injury severity (from no injury, to survival uncertain), controllability (from easy to uncontrollable) and probabilities of happening (from probable to "incredible"), and then has a table showing what ASIL level is required for each combination. In turn, the ASIL levels map to requirements for things like single-point-failures. The tables above show the combinations, and the ASIL requirements.

One key aspect of ASIL is that judgment is applied at the item level. Of course, "item" is a technical term but means something such as a braking system (including sensors, analyzers, and actuators). In fact, there is a whole hierarchy of common words that are technical terms. Inside an item are components, like an SoC. Inside that are parts, like a processor. A sub-part might be

an ALU, and an elementary sub-part might be a standard-cell. Phew. Don't tell them about transistors!

ISO 26262 goes on to define FMEDA, Failure Modes, Effects, and Diagnostic Analysis. As an example, think of a light (like a brake light, or a dashboard warning light) controlled by an ECU. How might it fail? How might we address those failure reasons and mitigate them?

FM			E
ID	Failure Mode	Failure rate (unmitigated)	Safety Goal Violation
FM1	Lamp burn out	0.1	Lamp state is not control-able
FM2	Driver failure	0.001	
FM3	Resistor failure in lamp ground path	0.01	
FM4	Switch failure	0.1	
FM5	Resistor failure in switch pullup	0.01	
FM6	Incorrect request from another ECU	0.001	
FM7	Corrupted request from another MCU	0.001	
FM8	Microcontroller failure	0.1	

D (Safety Mechanisms)	
ID	Safety mechanism to detect/mitigate the root cause
SM1	Briefly toggle state and read sense voltage
SM1	Briefly toggle state and read sense voltage
SM1	Briefly toggle state and read sense voltage
-	None
-	None
AoU1	Assumption of Use: source ECU SW design will keep errors below this rate
SM2	CRC or similar on command packet
SM3	Run self-test

The table above shows an analysis, with reasons in the table on the left, and possible mitigations on the right. These range from adding error detection codes to network packets, to not being able to address the problem (the switch fails completely). Which mitigations should we use?

Of course, the requirements for this "light control IP" depend a lot on what light it is controlling. The cabin light doesn't matter much, brake lights matter quite a bit, and the airbag problem light matters a lot (especially since you'd can't even notice, unlike the brake lights). Some mitigations cost nothing, and others are costly or technically impossible. So which to apply varies with the light.

Even that simple, almost trivial, example goes to show that ASIL FMEDAs are complex. Tensilica IP already has a lot of basic safety mechanisms built-in to detect problems and either correct them or react to them. These range from ECCs (error-correcting-codes) on memories, to watchdog timers (I've seen these called COPs for computer-operating-properly).

So what do you get with Tensilica IP to put it in your "item"?

- ASIL-B for random failures

- ASIL-D for design processes against systematic design faults
- ISO 26262 safety kits (safety manual, FMEDA and verification reports tailored to each configuration)
- Software development tools for ISO 26262
- ISO 9001 certified development process
- …and we have the certificates to prove it

Automobil Elektronik Kongress 2018

Once a year for 22 years the electronic divisions of the European automotive industry converge on the small town of Ludwigsburg just outside Stuttgart. Stuttgart is a big enough city, you've probably heard of it. In the automotive world, it is home to both Mercedes-Benz and Porsche. I am sure that it was a fairly sleepy conference 22 years ago, since automotive electronics was sleepy too. But with ADAS and autonomous vehicles, it is one of the best places to put a finger on the pulse of what is really going on. Since it is held in a small conference center in a small town, it always sells out.

Germany is really the heart of European automotive, home to not just Mercedes and Porsche, but BMW, Audi, Volkswagen, Opel (which used to be GM, but they sold it to Peugeot), Bosch and even Renasas (which is obviously not German but bases its automotive activities there).

Like most conferences these days, there is an App. Among other things the App shows all the attendees. There are about 600 of them. They include lots of very senior people. For example, Bosch has 15 people attending including senior vice presidents and the CEO—the sort of people who meet with Lip-Bu Tan when he is in Germany.

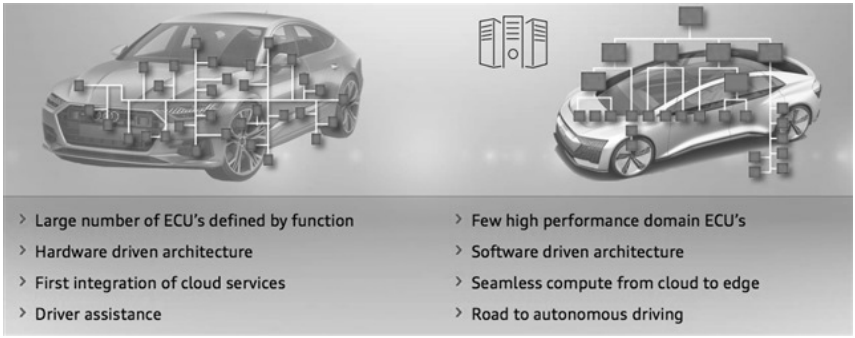
To give you an idea of how importantly the suppliers take this conference, despite only having 600 attendees (versus 150,000

for CES), Intel presented on the first day. But they didn't send the European marketing guy, or even the head of automotive. Brian Krzanich, Intel's CEO, made the presentation himself. At CES, Brian gave one of the keynotes, but at Ludwigsburg, he's just one of the non-keynote presenters. Last year, Jensen Huang, the CEO of NVIDIA attended and presented.

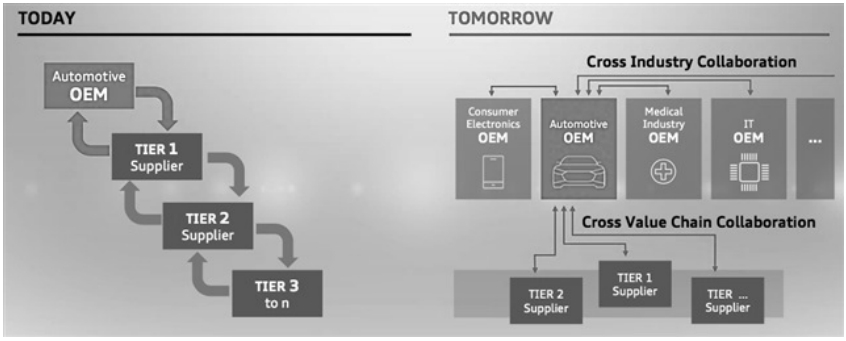
Overall Impression

When I came last year, the mood was very much that the automobile industry needs to get its act together, and cannot go on with business as usual. One of the core competences of the big automotive companies is mechanical design, especially engines. I hadn't really thought about it too much until Tesla has been struggling, but clearly another core competence is knowing how to do high volume vehicle manufacturing. What is, for sure, not a historical core competence is developing software.

Historically, the automobile industry has been divided into layers. At the top are the car manufacturers (BMW, Audi, Ford etc). They are known as OEMs, and they seem to use the same term to refer to people on the top of the pile in other industries such as medical. They generally don't design any of the electronics in the car, they buy that from the next level down, known as Tier-1. These are companies like Bosch, Delphi, and Desnso. They never designed their own chips, they would buy those from the semiconductor companies, such as NXP and Infineon, which are known as Tier-2s. That structure is all changing. Indeed, during the two days of the conference, there were several companies that presented very similar slides, except for the logos of the companies involved.



The slides that were common were two-fold. The first slide would show how cars used to be designed, with electronic control units (ECUs) spread all over the car and without any central coordinating processor. The ECUs were all purchased from the Tier-1s, and the chips (mostly analog or microcontrollers) purchased from semiconductor companies. The new architecture is to have a big central ECU that controls all the advanced driving and communicates with other ECUs over networks, typically automotive Ethernet if forward-looking, or CAN-bus for older ECUs that don't need much performance (adjusting your seat or mirrors, for example). The above image comes from Audi, but is almost identical to several others.



The other slide was showing how the OEM, Tier-1, TIer-2 system is breaking down. All the big German manufacturers seem to be developing the software for these central ECUs themselves. Some seem to be doing chip design, or at least coordinating with semiconductor manufacturers directly. The Tier-1s are also designing similar central ECUs and doing their own chip design. There is a realization that advanced driving and the in-cabin user-interface (often called the HMI, for human-

machine interface) need to be differentiating. The above slide is Audi again, but it could have been any of the presentations by the main German manufacturers (OEMs).

Two things clearly worry the car companies. One is that they need to make sure that they continue to own the customer, even though "there will be less interaction since electric cars don't need oil changes." The other is a longer-term worry that young people don't want to own cars and just want the advantages of a car anyway, meaning ride-sharing. This can, in time, lead to a shrinkage of the car market as has already happened in the bicycle market in China due to bike sharing. However, there is a worse fate, which is nobody cares about brand in a ride-share or a taxi. If Uber eventually uses self-driving cars, then you will think of it as using an Uber (and maybe the car will even have Uber logos on the back) rather than a Toyota or a BMW. It's not a lot of good delivering the ultimate driving machine to people who don't want to drive the vehicle themselves. My daughter works in the whisk(e)y business and is effectively not allowed to drive by her company. She is perhaps a harbinger of the future since she takes Uber everywhere (she has a four-figure Uber bill each month). Her company picks that up, but even if it did not, it is probably cheaper than owning, insuring, maintaining, and gassing a car.

A challenge everyone talked about was hiring. One of the presenters said that in round numbers, it is 1000 engineers for infotainment, 1000 for chassis, 1000 for connectivity, 1000 for security, and 1000 for driver assistance. Different companies broke it down differently. But there aren't that many engineers in Germany. As Daimler said:

> *We can’t just be here. We need to be in Seattle, Sunnyvale, Berlin, Bangalore, Tel Aviv, Beijing. We need the talent to do it.*

Trends, Technologies, and Regulations in China's Auto Market

At "Ludwigsburg", officially the International Autombil Elektronik Kongress, there was a push this year to be more international. In particular, attendees last year had asked for a deeper perspective on what is happening in the largest automotive market of all. There were several presenters who covered different aspects of the market and I learned a lot from very basic facts (all interenal combustion cars in China have to be kept off the road one day per week) to more technical (self-driving cars in China assume that there will be a 5G network that can be relied on).

The most comprehensive presentation was by Dr Volkmar Tenneberger who works for the VW Group in China, and I stole his title for the title for this section.

It is easy to fall into the trap that the market you know best is the leader in some sense, and China, still being relatively poor, is not yet that significant. But here are a few statistics Volkmar presented to show just how big markets are in China. The per-capita GDP may be lower than the US or Western Europe, but there are a lot of capita. All data in this table is 2017, so it is pretty much up to date.

Market	Global	China
Vehicles	25.9M vehicles in USA, Japan and Germany	25.8 milliion vehicles
Ride Sharing	Uber gave 4B rides in 78 countries	DiDi gave 7.4B rides (just in China)
ERetailing	Amazon took in $6.6B on Cyber Monday	Alibaba took in $25.7B on the "11/11" festival

Regulations

The Chinese government is pushing very hard to encourage electric vehicles, in their terminology NEV for "new energy vehicle". I think there are two reasons for this. The first is that there is an urgent need to reduce pollution, especially in Beijing. The second is that electric vehicles are a discontinuity in the market, giving China a chance to be the world leader in electric vehicles in a way that would be impossible for ICE (internal combustion engine) vehicles. Overall, the Chinese market is very regulated compared to US or Europe. Even when you have bought a car, you don't automatically get a license plate, you either need to get in line or get in an auction. The authorities limit the number of vehicles by limiting the number of license plates available.

There are multiple incentive programs in place:

- National subsidies of up to 666 K RNB, equivalent to €8.92K (a bit over $9K). There are also often local subsidies.
- Unlike for an ICE vehicle, there is no waiting time for a license plate and it is free (in Shanghai they are auctioned and a plate was €12K in the May auction)
- There are no driving ban days, whereas every ICE vehicle is banned for one day per week in Beijing
- There is tax exemption on NEV (and plug-in hybrid, PHEV, except in Beijing where only NEVs qualify)

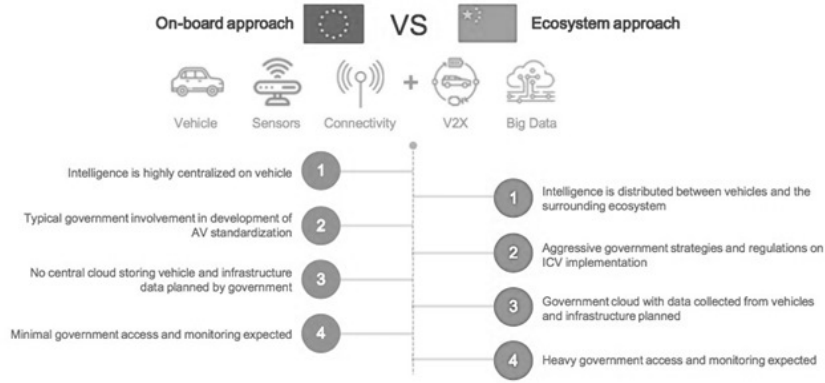
Volker did an interesting calculation and took the NEV subsidy and worked out how many batteries it would pay for, coming up with 90kWh. "In Germany, we say the battery is too expensive, but in China the subsidies cover it." The sum of all the incentives means that you don't have to be a technology enthusiast to buy an NEV. This year there will be more NEV sold than ICE. "They are everywhere, nobody nudges their neighbor and says 'look a Tesla'."

There is also a quota system a little similar to the CAFE system we have in the US, that operates across a company's (like VW's) whole fleet. Volker said that "for one EV I can sell two ICE at

present." There is pressure to keep the cars small too. "Chinese people love big cars and given full freedom they will buy big SUVs...but we have to keep the fleet targets on the correct part of the curve as the numbers tighten over time.

A big difference between China and Europe is that China is not pushing for all the intelligence on the vehicle to be on-board. In fact that shows up in the names used: autonomous vehicles in Europe, Intelligent connected vehicles (iCV) in China. In China, the intelligence is distributed between vehicles and the surrounding ecosystem. This obviously depends on a very reliable network. Of course, 5G has not been rolled out yet, but I have to say that in all my trips to China, I have never once noticed that I had no signal: not in the countryside, not in underground parking garages, not on the subway. I can't say that for any other country.

In Europe, governments are typically only involved in autonomous vehicle standards. but no central cloud infrastructure is planned by governments, and minimal government monitoring is expected. China has aggressive government strategies and regulations on iCV., including (planned) running of the cloud infrastructure with data collected from vehicles. There is expected to be heavy government access and monitoring.



Megatrends

Nine cities so far have published regulations for road-testing autonomous vehicles. 13 companies haveacquired licenses from 6 of the 9 cities, and more companies and cities are expected. The

big tech companies all have test licenses. There are already 27km^2 in Shanghai and will be 100km^2 next year.

Another difference from the US and Europe is the dominance of WeChat (WeiBo) as a sort of "super app". In the rest of the world there is Facebook, WhatsApp, GoogleMaps, Tinder, Yelp, Paypal, Uber, games, and more. In China, these functions are all part of WeChat. Despite the name, it's not like WhatsApp, with just chat and voice. It has navigation, social, video, photo, payment, dining, and more.

As a result, mobile payment is a part of everyone's everyday life. 84% of Chinese respondents are comfortable not carrying any cash, since you can always pay with your phone. "Whatever, I can use mobile paymentsAs a result, in the last 2 or 3 years, mobile payments have shot up and last year reached over 200 trillion RNB (about $35T).

Summary

- China is serious about its "Made in China 2025" with ambitions to become the lead market for ICVs (and for NEVs too).
- The Chinese market for electric vehicles is policy-driven more than customer-driven.
- The digital ecosystem is almost entirely within China (behind "the great firewall") and centrally controlled, but more widespread than in the West. Literally everyone, from the poorest peasant to the techies, is online all the time, and the society is becoming cashless.

Automotive Summit: The Road to an Autonomous Future

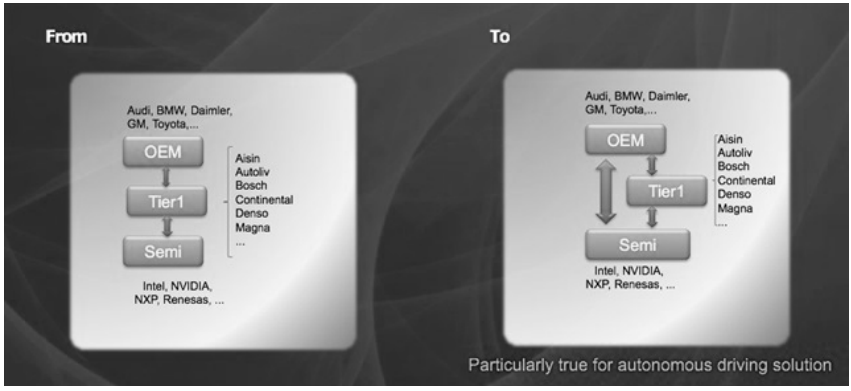
Before Thanksgiving, Cadence held an Automotive Summit. I was going to dive into some of the detailed material presented, but it occurred to me that it might be a good time to step back and

take a look at where we are in the industry as a whole. I've written a lot about automotive. Don’t forget it was just 14 years ago that DARPA had the first "Grand Challenge" for driverless cars over a 120-mile route in the Mojave desert. But the furthest any vehicle got was 8 miles. A lot has happened since then.

I'll use Raja Tabet's opening presentation as an organizing theme, but drop in other stuff. He's Cadence EVP of Emerging Technologies, of which automotive is the main one.

Automotive Semiconductor

There are a demand-side reason and a supply-side reason for all the interest in automotive semiconductors. The demand side is that ADAS (automatic driving assistance systems) and eventually autonomous driving require a big change in the requirements for semiconductors. Historically, automotive semiconductors have been built in ten-year-old processes, with lots of characterization data, lots of analog, and small microcontrollers. In the future, the processing requirements mean that they will be built in advanced processes, such as 16nm and 7nm FinFET. The supply side is that the wind is that the overall semiconductor industry s growing at a CAGR of 5.2% whereas automotive is almost twice that, at 10.6%, and autonomous driving silicon over twice that again, at 23.6%. The means that the average semiconductor value per vehicle is going to double from about $250 today to $600 in 2022. It is expected to triple with the deployment of level autonomous vehicles.



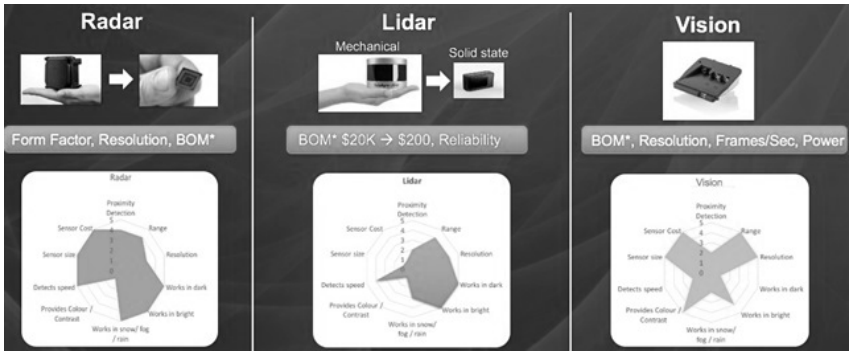
This is leading to a shake-up in how the automotive industry is structured. I was at the Ludwigsburg conference earlier in the year and practically every company had a version of the above picture with just the company name changed. Historically the car companies (OEMs in automotive-speak) like Ford, Toyota and BMW would deal with major automotive suppliers (tier-1s in automotive-speak) like Bosch, Denso, and Continental. The tier-1s would deal with the semiconductor companies (the big ones in automotive being NXP, Infineon, Renasas, STM, TI). In particular, the OEMs didn't really have relationships with the semiconductor companies, and they didn't really understand semiconductor design (nor even create their own software).

That is all changing. The OEMs realize that they need to be in control of at least some of their own silicon and a lot of their own software, otherwise their cars will be just like everyone else's. It reminds me of the transition in mobile where the market leaders, starting with Apple, all realized that they needed their own application processors at least. The tier-1s also needed deeper semiconductor relationships or they would be bypassed.

I talked about the demand-side (automotive) and supply-side (semiconductor) above, showing both sides need each other. But it goes deeper. The automotive companies have no experience of designing high-performance SoCs in advanced processes. But the semiconductor industry has no experience of trying to deliver automotive reliability in high-performance SoCs. It has been a learning exercise on both sides.

Big change like this always attracts new entrants. There is a lot of VC investment in new players, and a race to create working solutions. Meanwhile, as the diagram above showed, there is vertical integration with both OEMs and tier-1s seeking their own SoCs, while companies like NVIDIA and Intel/Mobileye provide standard solutions.

Sensors



A key part of autonomous driving is making the vehicle "see". There are three main technologies for this:

- Vision: cameras of the type developed for cell-phones with (typically) CMOS image sensors.
- Radar: this has the big advantage that it can "see" in the dark, in rain, fog, snow, and other conditions where vision is poor.
- Lidar: this uses laser pulses to build up a picture of the vehicle's environment. Those are the big spinning things on the roof of first generation test vehicles, but for mass deployment they need to be solid state.

Sensor fusion sounds like a type of music, maybe next-generation techno, but in fact it refers to combining the information from all those sensors to get a single picture of what is going on around the vehicle, as a first step for deciding what to do next (keep driving, turn a corner, brake, and so on).

Communication

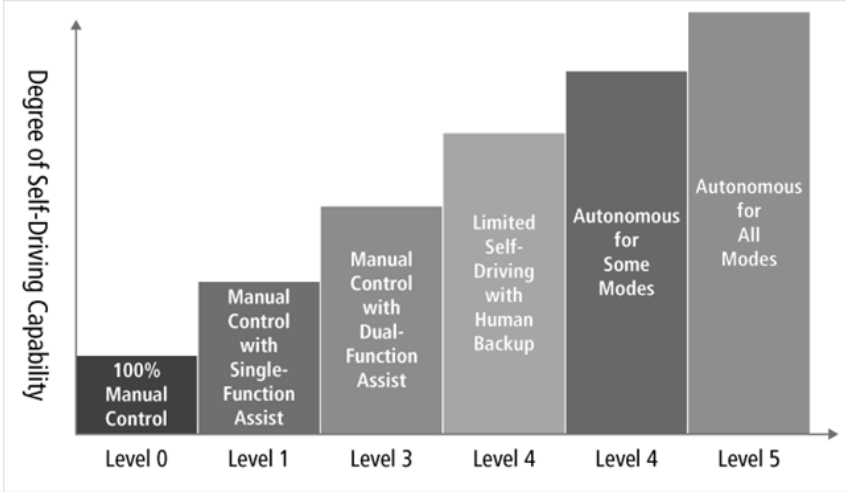
Automotive vehicles require communication. At the very least, mapping and road condition data needs to be uploaded from the cloud to the vehicle. That requires a network connection, and it is seen as one of the drivers for 5G since it has high bandwidth and low latency. But vehicles need to operate even if the connection fails, or is congested: you can't go to the cloud and back to decide if a traffic light is red.

But there are potentially other forms of communication, which are known as V-2-X. The X is usually either another V for V-2-V (vehicle to vehicle), where vehicles can potentially signal to other cars what they are doing, or that they are approaching a blind intersection or whatever. The other choice I for Infrastructure so V-2-I (vehicle to infrastructure), where things like traffic lights can communicate with vehicles to more intelligently decide when to change, or perhaps signal to vehicles what they will do soon.

I would say there are three modes of thought about communication (I leave it to you to draw analogies between how cars will work and how society is run):

- Chinese: autonomous cars will be always connected, and a lot of coordination will be done in the cloud in a centralized manner. It helps that I have never once had no signal on my phone in China, even in subways and basement parking garages.
- Self-contained: cars will contain all the sensors required to drive themselves, and the connectivity is only used for things like road condition updates that are not time-sensitive. Vehicles will not wait on the government to be a partner in getting to autonomy (except for legal and regulatory framework).
- Self-contained and V-2-X: All the V-2-X stuff is driven by the traditional automotive industry and government bureaucracy, especially in Europe. But I have a feeling it might all just get overtaken by the first two. By the time traffic lights can communicate with vehicles (remember, there are about a million in the US alone) the vehicles will just read the light, and perhaps even do what I do and read the pedestrian crossing countdown too.

Levels



I think I mentioned driving levels above without explaining what they are. The Society of Automotive Engineers (SAE) has defined a 5 level progression (6 if you count level 0, which is no automation at all). They are:

- Level 0: manual control
- Level 1: driver assistance
- Level 2: partial automation (feet off)
- Level 3: conditional automation (hands off)
- Level 4: high automation (eyes off)
- Level 5: full automation (mind off)

Most cars today are level 1. Cars with things like lane-following and automatic emergency braking (AEB) are level 2. The first real "self-driving" comes in at level 3. Full automation, level 5, where you can imagine a car that no longer needs a steering wheel, is extremely demanding and likely a long way in the future. I noticed on a fully automated rail system, the London Docklands Light Railway, that there is still a full control dashboard underneath a normally-locked panel—and trains are simple, they can only go or stop. Cars will have manual controls for extreme conditions and breakdowns for a long time, I think.

Automotive Sensors: Cameras, Lidar, Radar, Thermal

Camera

Radar

LIDAR

One of the first non-Cadence presentations of the day at the Cadence automotive summit was Manju Hegde, the CEO of Uhnder. It turned out to be a blessing that his company is still mostly in stealth mode, since it meant that his presentation had none of the "my company's solution is the best" aspect. Instead, he gave a wonderful overview of sensors, focused on what he calls the core sensors: camera, radar, and lidar. He did say that he was doing radar, but admitted that when he was raising money, most VCs said to him "Why aren't you doing lidar? Radar is a solved problem."

Requirements

Menju had a top ten list (actually eleven) of critical aspects of a sensor:

1. Range. sufficient to accommodate the vehicle speed.
2. Field of view (FoV): sufficiently wide to encompass the whole scene. Some of this is driven by regulation for automatic emergency braking: AEB 2018. But 2020 and 2022 are more stringent and require avoiding collisions at intersections, requiring a wider FoV.
3. Angle. Angle detection and resolution sufficient to detect relevant features. One issue is that human being are relatively weak targets for radar (hey, so are fiberglass sailing boats, so they have a special reflector at the top of the mast—I propose special hats).
4. Velocity. Measure the speed of and resolve moving objects. Radar can do this, but most lidar can't do it

directly (they can do it indirectly by measuring the difference between frames over time).

5. Classification. Radar does this poorly. The US military has radar signatures of every enemy tank and plane, and use machine learning. It is difficult to identify object type (tree, pedestrian, car etc) at range.
6. Color. It's less important than you might think; colorblind people drive perfectly well. But it is very important for traffic lights in particular. I had a colorblind friend who told me he can't tell if a light is red or green until he is close enough to see if it is the top light or the bottom light, which means that at night he has to assume all lights are red until he is close.
7. Processing overhead. There are large amounts of signal processing and image classification involved in getting from raw sensor data to "it's a child" or "it's a fire-hydrant".
8. Operation. The big challenge is full functionality in all lighting (day and night) and in #9 which is...
9. Adverse weather: Rain, fog, snow. Even humans have trouble when it gets bad.
10. Interference. There are two types, environmental, and from other sensors. A camera gets dazzled more easily than an eye. In radar and lidar there are multiple units on the vehicle. Radar sends a signal out that deteriorates with the usual square of the distance, and the reflection that comes back also deteriorates at the square of the distance (from the target), so a total of a fourth power. So a car coming towards you is flooding you with radar at R^2 and you are detecting at R^4, so it is easy to get "dazzled." Then add that you might have 4-6 radars on each vehicle.
11. Cost. The predictions of when self-driving cars are going to be available depends partially on cost. Google/Waymo can run a few cars in Phoenix and not care about cost, but low sensor cost is critical for volume production. Eventually this needs to be technology that we can put in a $20,000 car.

Sensors

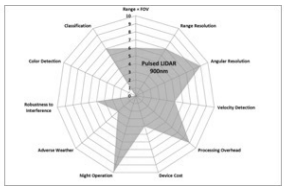
The three sensor types are vision (cameras), radar, and lidar.

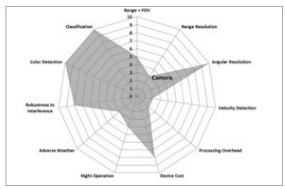
First vision. The CMOS image sensors (CIS) needed for automotive are different from consumer electronics (which is mostly focused on making images Instagram-good). Automotive needs high dynamic range, improved low light sensitivity through larger pixel sizes, lower resolution, faster response time, and performance needs to work at higher and lower temperatures (smartphones live in our pockets, cars live in Minnesota and Arizona). Obviously, one big disadvantage is that like humans, at night they are limited to the vehicles headlight illumination. Cleaning is an issue, the lens needs to be kept clear of muck.

Next Lidar. There are a huge number of different types of lidar (even more than the number of ways to capitalize lidar). For automotive use, during the current prototype phase they use mechanical scanning, but the focus in the longer term is on solid-state lidar, either using MEMS mirrors, or an optical phase array. One subtle issue is that the lowest cost comes from near infrared 850-940nm wavelength light, but this is limited by eye-safety requirements, and solar background. The higher performance is 1550nm which has 5 orders of magnitude (500,000X not 5X, although a later speaker said "just" 120X) more allowed energy, 10X less solar background, but is expensive. Its big plus is great angular resolution, but has a problem with very non-reflective objects and fog/rain/snow. Cleaning is an issue.

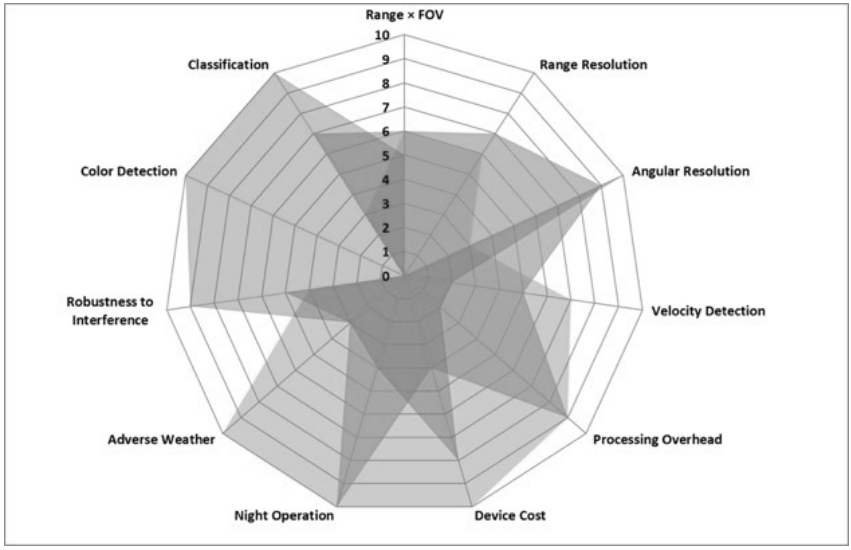
Radar. This is current radar since Uhnder is doing next-generation radar which Menju gave some sneak information about later. All current radar uses analog FMCW (frequency modulated continuous wave). Transmit a chirp, mix with the reflected wave received, downshift to baseband, use a low pass filter. One issue is that you have multiple radar units on a vehicle, but only one can be transmitting at a time, so it is necessary to do time division multiplexing (basically, rotate around each radar one at a time).

Strengths and Weaknesses





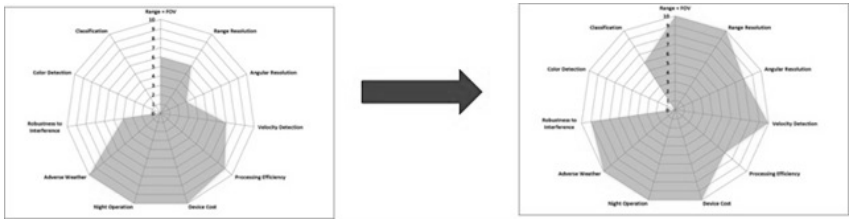
We can put each sensor type onto a spider diagram as above. Radar on the left, lidar in the middle, vision on the right.



The reason that these are all important is that if we overlay them, the strengths of one make up for the weaknesses of the other. Radar can't tell if a traffic light is red or green, but vision can. Vision can't see well in fog but radar can. And so on. But there are stil substantial gaps.

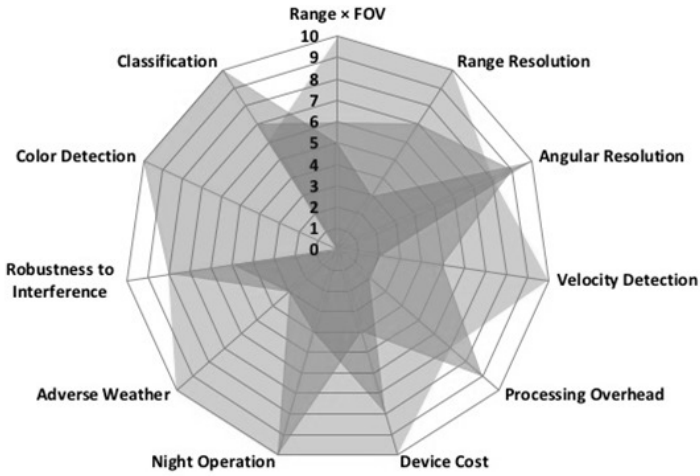
Next-Generation Radar

What if we transformed the radar sensor? Today's radars were all designed for detecting large targets. Lots of companies are working on this. Needless to say, it's what we're doing.



What if we added more range, more resolution (both vertical and horizontal), very sensitive velocity detection, better bright to dark

target ratio, and more interference-resilience. That would transform radar roughly as in the above spider diagrams.



More importantly, you fill the gaps. In fact, they are so well filled in that lidar is of marginal value. There will still need to be lidar on the front for fast driving to detect things like a piece of wood in the road that is not radar reflective and can't be detected by a camera in time, but you will need a lot fewer lidars.

Of course, Menju is still in stealth mode. But it is clear the attraction of what Uhnder is doing (if it works, and it works better than the competition, yadda, yadda) is that next generation radar is a great solution, and from a cost point of view, adding it to a vehicle can be "paid for" by removing many of the lidar sensors.

Thermal

Chuck Gershman of Owl Autonomous Imaging added a fourth sensor type to the mix: thermal. I won't cover everything he said, a lot was a similar analysis of the strength and weakness of various sensor types. But the big advantage of thermal is that it can detect living objects, and it works the same in day and night and has all-weather operation.

Lidar alone is not good enough since it sees but doesn't understand. Plus lidar range is seriously reduced in bad weather.

905nm lidar, in particular, goes blind in fog and rain. Owl has a focal plane array (FPA) 2-color detector that they have delivered to the Air Force.

As Chuck said, wrapping up:

> *Unlike in the Sixth Sense, we see live people.*

The Wall Street Journal on Sensors


THE WALL STREET JOURNAL.

The Bumps Ahead for Autonomous Vehicles

How soda cans, bad weather and snakes present challenges to the latest technology

As it happens, The Wall Street Journal just ran a piece on automotive sensors. I say "a piece" but it was more of a weird animated graphic It has some odd misunderstandings in it, or at least an odd way of putting things:

> *Cameras aren't as effective at capturing the environment during low visibility conditions, including after sundown. Lidar and radar are unaffected by darkness, however, because they collect information about the environment from electromagnetic wavelengths higher and lower than that of visible light.*

Well, that's true. Except for the "because". It's true about eyes too. We normally address that by putting high-intensity visible wavelength light generators on the front of our vehicles and seeing what comes back. These are known as headlights. Lidar and Radar are not unaffected by darkness due to the wavelengths used, but because they already do something similar, putting out pulses of electromagnetic radiation and then seeing what comes back.

Another quote:

> *Driving on uneven surfaces can compromise the calibration of lidar and cause excessive wear on the ball-bearings that stabilize the sensor atop the car. The more*

> *often a vehicle encounters these conditions, the more frequently the lidar sensor will need to be replaced.*

I assume this is true. But those big spinning Lidars on the vehicle roofs are not seriously proposed as a solution to enter commercial volume production. They cost more than the rest of the car. These are experimental platforms. Lidar will require solid-state solutions for economically-viable commercial production, as discussed above. Solid state lidar does not have ball-bearings and all the rest.

The WSJ's final sentence:

> *And however flawed humans may be, they're still the best drivers on the road.*

And who could be against motherhood, or apple pie? These "best drivers on the road" killed 40,000 people in 2017 in the US alone (1.25 million globally in 2013, the most recent number I can find). Actually, despite the appalling numbers I just mentioned, humans really are pretty good drivers and only have 1.18 fatalities per 100 million miles, so we don't have enough data one way or another to say whether humans are the best drivers on the road. And, having had two teenage drivers in my family, I can say the bar can be pretty low. But to be fair to my kids, I don't think it was so much that they were 16, but that it takes a couple of years to build up automatic reflexes—I'm sure we've all had the experience of finding our foot on the brake pedal before we consciously realized why. New drivers don't do that.

Chapter 3: Artificial Intelligence

Deep Blue, AlphaGo, and AlphaZero

There are three major events in computers learning to play board games at a very high level:

- In 1997, Deep Blue defeated world chess champion Gary Kasparov
- In 2017, AlphaGo defeated #1 world Go champion Ke Jie
- In 2017, AlphaZero defeated Stockfish, then the world leader in computer chess

You have probably heard a lot about the first two. But the third is probably the most important. But let's start with Kasparov's defeat.

Chess

In 1997, IBM's computer Deep Blue beat Gary Kasparov, the world champion at the time. It was the first defeat of a world champion by a computer under tournament conditions. In some ways, it wasn't that Deep Blue cracked some secret about how to play chess, just that it was the world's most powerful computer at the time, and if you throw enough computer power at a combinatorial problem then you can use simple algorithms. It was perceived as a victory for brute force rather than a breakthrough in artificial intelligence. If you want to know more, there is a video *Game Over: Kasparov and the Machine*. As a sign of just how fast technology was advancing, this was the second matchup of Kasparov and Deep Blue. There had been one the year before, in 1996, which Kasparov had won.

It is worth pointing out that chess has been played for a long time. There has been a huge amount of analysis of chess, especially openings and endgames, which are well understood. Deep Blue had all that built in, of course. There are also thousands (millions?) of high-level chess games available for

analysis and training. There are also chess computers. So a young person can leverage all this and can train against chess computers that are rated higher than any human player. As a result, chess grandmasters are getting younger, and Bobby Fischer's 15½-year-old record, which stood for 33 years, has since been beaten by dozens of young players. Now, instead of having to spend a lot of time in book learning studying great games, openings and so on, young players can just play against chess programs on a PC that already embody all that prior knowledge. Even so, it takes nearly 10 years from first learning the basics to getting to a grandmaster title. For example, Magnus Carlsen, the current world champion, became a grandmaster at just 13½, having been introduced to chess at the age of 5, and playing his first tournament at the age of 8.

Go

Having defeated the world champion, there wasn't a lot more that could be done in computer chess (although we'll come back to chess later), and the man versus machine arena moved to Go. This is considered a more difficult game for a computer. Indeed, at the time of Kasparov's defeat, there were people saying that Go was too complex for the foreseeable future. Go has simpler rules than chess, but many more moves at each point in the game, and is considered more intuitive and less analytical than chess.

Traditionally, computer games use a technique called minimax. The basic idea is to look at all the moves you can make, followed by all the moves the opponent can make, then all the responses you have. This explodes combinatorially so is not practical for games like chess or Go. The name comes from the fact that you want to pick the move that maximizes the minimum position you end up with, that is it guarantees that you end up in the best possible place even if your opponent plays perfectly. But it requires an algorithm that can assess how good a given position is. This is somewhat of a black art since a good position is one you can win from.

It reminds me of a lot of EDA algorithms. For example, the definition of a good placement is one that can be routed well—but routing is far too expensive to run as part of placement so other heuristics need to be used. One approach to pruning the

tree, known as alpha-beta pruning, is to cut out branches of the tree when you have already found better alternatives. But that assumes that the evaluation function is accurate and doesn't miss moves that superficially look poor but in a few moves time lead to good outcomes.

Chess and Go are both full of moves like that, so the naive approach won't work. So under the hood of AlphaGo are two neural networks, one for assessing the value of a position, and one for deciding which moves to continue to analyze and which to prune. These are called the value network and the policy network. AlphaGo was trained on a database of around 30 million moves, initially with the goal of getting it to make the same moves as humans would make. Then it was further trained by reinforcement learning by playing itself.

AlphaZero

Alpha Zero (and a forerunner, AlphaGo Zero) is a different version of the program. The "zero" in the name indicates that it starts from nothing. It gets given the rules of the game. It can play chess, Go, or a sort of Japanese chess called Shogi. The training is entirely done without any knowledge of previous games, human play, tactics, or strategy. The implementation has a lot of power, using 5,000 first-generation Google Tensor Processing Units (TPUs) to generate games for analysis, and 64 second-generation TPUs to train the neural networks.

There is some debate as to whether the version of Stockfish that AlphaZero beat was optimal, running with 64 threads and a gigabyte of hash. But that is a wrinkle. AlphaZero beat Stockfish, the best computer chess program in the world, after just nine hours of training.

Not Just Stockfish—Its Programmers

This is amazing in so many ways. I said above that a human with access to the best computer chess programs in the world, and access to centuries of analysis, takes eight or ten years to get to grandmaster standard. AlphaZero did it from a standing start in less than a day. Yes, chess is an artificial environment and this is not general intelligence. But it is so far beyond human performance as to be hard to believe. The nearest any human has

come to trying something similar can be read in *A Chess Novice Challenged Magnus Carlsen. He Had One Month to Train*, which ended as you would expect. In fact, since Stockfish can beat human players already, AlphaZero didn't just, in effect, beat a novice who trained for a month, it beat all the grandmasters who trained for decades.

More significant, perhaps, is that AlphaZero didn't just beat Stockfish. It beat Stockfish's programmers. That is, in a few hours it beat the years and years of work that has gone into making computer chess as good as it can be, and better than humans. Everything that has ever been done in chess forever, in both human chess and computer implementations, was no match for half-a-day training on AlphaZero's (admittedly extreme) hardware.

What's next? Maybe given the rules of arithmetic, a few hours later it will discover differential and integral calculus, matrix algebra, group theory, non-Euclidean geometry, and all the other facets of advanced mathematics that is the accumulated knowledge of millions of people over hundreds of years.

Of course, the basic approach only works in a closed domain with some simple rules where a lot of contact with the real world is not required. Discovering how DNA works, or astronomical dark matter, or the relationships of subatomic particles, require observations, not just running AI algorithms against themselves. But the whole world is actually built up from simple building blocks. Cells, molecules, atoms, particles, quarks.

Even if we don't understand the details, we know that, by definition, consciousness and intelligence in humans are emergent behaviors of atoms and chemistry (or just physics if you go lower still).

Accelerating AI: Past...

SiFive does a quarterly series of tech talks, not necessarily directly to do with SiFive or even RISC-V. This quarter it was Krste Asanović on *Accelerating AI: Past, Present, and Future*.

If you know Krste's name at all, then it is probably as one of the leaders of the team that created the RISC-V ISA and as its primary evangelist. In one presentation I saw him give last year, he threw out a line that his PhD is not in instruction set architectures, but in neural networks. "Bad sense of timing, I was twenty years too early." As you know, if you haven't been on sabbatical in Timbuktu, neural networks are one of the hottest areas in computer science, semiconductor, automotive, vision, artificial intelligence. But having been working in and around neural nets his whole career makes him the ideal person to tell the whole story.

Krste opened by pointing out that we have applications, things we want to do. And we have technology, the stuff we have to do it with, which today is primarily silicon. Computer architecture is what comes in between the applications and the technology. Today, the hardest application we are dealing with is AI, and so there is a lot of investigation going on into what is the best architecture. As Krste indicated in his throwaway remark about being too early by twenty years, he has been in this field for a couple of decades. But it actually goes back a couple of decades further than that.

The Prolog Era

Let's start in the 1980s with Japan's fifth-generation project. MITI (the Japanese ministry of industrial trade and industry) had been very successful in directing a large part of the economy to improve quality and also identified semiconductor as an area for special investment. Today, when Japan is famous for its high quality, it is hard to remember that in the 1960s it was apparently a joke. I grew up in England, and "made in Hong Kong" was a similar joke (and, I only realized later, meant made in China and somehow exported through Hong Kong to avoid trade barriers).

Anyway, MITI decided that the future was artificial intelligence (AI), and logic programming was the way of the future, and that programs would all be written in Prolog. It was then just a question of building optimized fifth-generation architectures to run them faster and faster.

(In a weird coincidence, the primary textbook for learning Prolog was *Programming in Prolog* by William Clocksin and Christopher Mellish. When I first went to Edinburgh to do my PhD, I had to find somewhere to live from 400 miles away (this was before email, Internet, etc) by phoning the University Housing Bureau. They found me a room in Morningside, famous for its accent. In Muriel Spark's book *The Prime of Miss Jean Brodie,* she lives there. If you've seen the movie of the same title, her accent (well, Maggie Smith's) is a Morningside accent. The joke is that in Morningside "sex [sacks] are what you put your potatoes in". Another guy who had done the same thing from much further afield, Washington State, was Bill Clocksin. So for a year he and I were roommates, although I think this was before he was doing much with Prolog since I don't recall coming across it until I was already in the US. In that era, Edinburgh's departments of AI and CS were separate, and miles apart, so we didn't work together at all.)

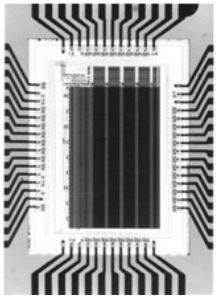
If you don't know anything about Prolog, you can think of it as a language that makes it easy to express constraints (parts of the Specman *e* language for verification, and some parts of formal verification have things in common). One example that is easy to explain is Sudoku. A valid Sudoku board has each number in every row, column, and 3x3 square. The code below expresses that. Unlike an imperative or even functional programming language, this can run in a number of ways, trivially testing if a given solution is correct, but also by generating all valid Sudoku squares, or completing partial squares (which is what we think of as "solving" a puzzle):

```
sudoku(Rows) :-
 length(Rows, 9),
 maplist(same_length(Rows), Rows),
 append(Rows, Vs), Vs ins 1..9,
```

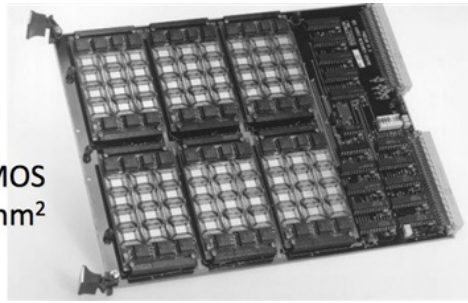
```
 maplist(all_distinct, Rows),
 transpose(Rows, Columns),
 maplist(all_distinct, Columns),
 Rows = [As,Bs,Cs,Ds,Es,Fs,Gs,Hs,Is],
 blocks(As, Bs, Cs),
 blocks(Ds, Es, Fs),
 blocks(Gs, Hs, Is).

blocks([], [], []).
blocks([N1,N2,N3|Ns1], [N4,N5,N6|Ns2], [N7,N8,N
9|Ns3]) :-
 all_distinct([N1,N2,N3,N4,N5,N6,N7,N8,N9]),
 blocks(Ns1, Ns2, Ns3).
```

It is hard to remember how threatened people in the west felt by Japan's Fifth-Generation project. MITI seemed invincible. They had declared that Japan should get into automobiles, and suddenly they were of higher quality than Detroit and imports were starting to shake up the market. They had declared that Japan should get into semiconductors, and they had driven the US out of the memory market. They declared that Japan would build AI computers, and there wasn't any obvious indicator you could point to that indicated how totally the program would fail.



1.2µm CMOS
5.8 x 7.9mm^2
8 MHz



At the time, Krste was an undergraduate in the UK and thought, "maybe they know something." He went to work on a project with Padmavati at GEC Hirst Research Center in 1987-89. They were doing a machine for natural language processing. It had 170,000 processors, with 148 processors per chip. The problem was that each processor only had 36 bits of storage. In was built in 1.2um (1200nm) CMOS and ran at 8MHz and was called

SPACE. In some ways, it was the ultimate in computing-in-memory since every memory bit had logic attached. Everything was done by associative lookup. It was designed to accelerate Prolog unification and LISP associated arrays. However, Krste and others soon discovered that if you programmed on the bare metal, and bypassed Prolog and LISP, it was over a hundred times faster.

Lessons that Krste took away from that:

- The language needs to match the application domain
- You need high memory capacity to hold the problem state, or else you waste too much effort swapping parts of it in and out
- Fine-grained computing-in-memory is very inefficient since only a tiny fraction of the memory is involved in any compute step
- Bit-serial arithmetic is very inefficient, most of the machine cycles in practice are sequencing multi-bit adds

Since the world is not programming everything in Prolog today, something else happened in AI, and that something was neural networks. They had actually started back in the 1950s with single-layer perceptrons, with limited success.

The ICSI Era

In 1989, the International Computer Science Institute (ICSI) in Berkely was working on The RAP Machine (Ring Array Processor) for fast training of "big dumb" networks for speech recognition. it was based on TI DSPs, with 4 per board, and up to 10 boards. It was fast and flexible, but at $100K+ (in 1989 dollars) was too expensive, you couldn't give one to each researcher.

Krste, now a naïve grad student said:

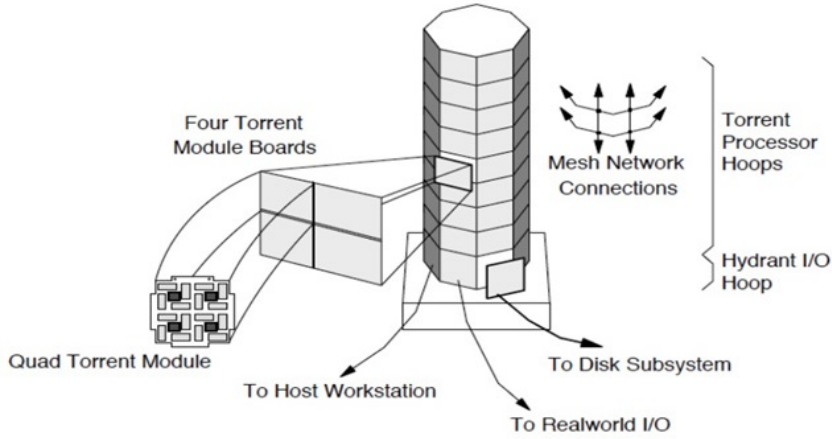
> *I joined the group to design custom chips for neural nets, sounded cool.*

In those days, MOSIS had a "tiny chip" program where it was $500 to fab a 2.2mm x 2.2mm chip in 2u (2000nm) CMOS. They

used this to build various bits of the Highly Pipelined Network Trainer, or HiPNeT-1.

But before they got finished, the language people came up with a new architecture. But it needed a different architecture, and so completely different chips. Like any computer scientist when faced with that sort of problem, people who keep changing the specification, the answer is to make it programmable. So they built a VLIW/SIMD (sorry, but if you don't know what these stand for already, just telling you the words won't help you understand what they mean) machine, with a vector unit and a scalar unit, similar to many embedded DSPs today (Tensilica's basic architecture is similar, although with more of everything).

The SQUIRT test chip from 1992, was built in 1.2um CMOS with 2 metal layers and ran at 50 MHz, consumed 400mW at 5V (yes, remember we used to have 5V power supplies), and was 8x4mm.They then, in 1992-95, they built the CNS-1, the Connectionist Network Supercomputer.



One of the things that they had learned from watching earlier work at Thinking Machines was just how important the physical enclosure was. The benchmark program was to evaluate a network with a million units and a thosand connections per unit, 100 times per second. This required about 200 GOPS (for comparison, the neural engine in the iPhone X decades later is 600 GOPS).

Then there came a realization. Wait, we haven't finished SPERT and we're doing to do another processor...who's going to write all the software? It turned out that the correct answer was to scrap the whole thing, since VLIW is a major pain, with no upward compatibility. Even the scalar compiler was complex, and the parallel compiler was harder than complex. Even assembly was next-to-impossible to write.

It's the Software, Stupid

The big realization: it's all about the software. So don't build something that is impossible to program. Instead, use a commercial RISC, add a vector unit, and extend a standard compiler. Basically, put a Cray on a chip. They called it Torrent-0 or T-0.

The system architecture choices were SPARC, HP-PA, PowerPC, Alpha. They picked MIPS since there were good tools, desktop workstations they could use for development, and even a 64-bit extension. However, there were no soft cores, this was still before the age of synthesis. Verilog was just showing up. So they decided to build their own MIPS core. After all, how hard can it be?

They tried to get a license to the MIPS core in 1991. This is just to use the architecture. But it was $2M. That was just for "their blessing and some test vectors".

> *If you want to know where RISC-V came from, then this was one of the formative experiences*

So they cheated. They went ahead and built it anyway, and left out the few bits that they knew were patented and that they didn't need. It filled the reticle completely. HP built it (they still had their fab in Boise in those days). It had 8 MACs per cycle, a 32-bit datapath, 16x16 multipliers. It was 16.7mm square in 1um CMOS and ran at 40MHz.

The chip, SPERT-2, worked. 35 boards were built in 1995, shipped to nine international sites. It was used as a research platform for nine years (which is more a century in computer years) and was last powered up in anger in 2004.

The Wilderness Years

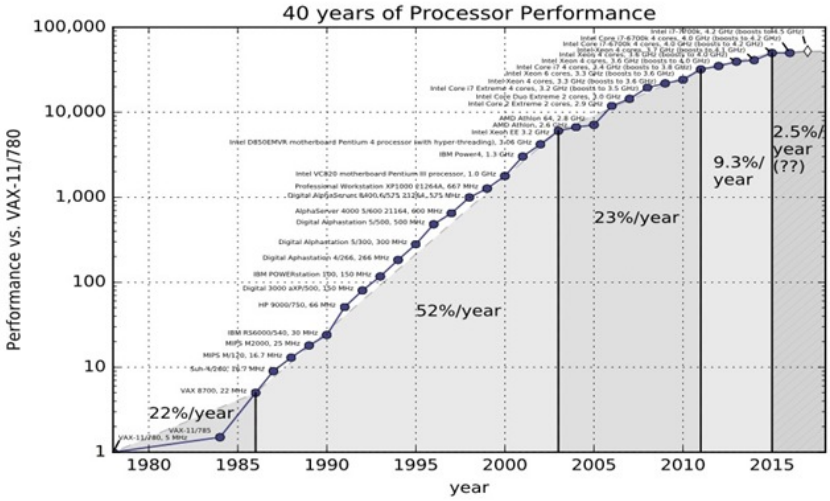
Neural networks faded in popularity and became a niche. In 1992, Intel introduced MMX, which added narrow fixed-point arithmetic and data parallelism, primarily for MPEG video decode. So that meant that multimedia had almost all the capabilities needed. If that wasn't enough, Moore's Law was still in full swing and a few years later your code would just run faster without your having to do anything.

Graphics Processing Units

GPUs were originally dedicated fixed function units, cramming workstation-like graphics for PCs. Gradually, in the early 2000s, more programmability was added. The architectures was naturally massively parallel, but with a very constrained programming model. However, people found out how to do general purpose computation by mapping input and output data to images, and computation to vertex and pixel shading computations. But it was next to impossible to program, even with the rudimentary languages that started to be developed. The big change came in 2006, when NVIDIA introduced the GeForce 8800 and the new programming language CUDA (for Compute Unified Device Architecture). The industry as a whole subsequently pushed for OpenCL, a vendor-neutral language with similar capabilities to CUDA (CUDA is still proprietary to NVIDIA).

The idea was to take advantage of GPU computational performance and memory bandwidth to accelerate some general purpose computing. It was also using the "attached processor model" with a general purpose host CPU to handle the housekeeping, and a specialized data-parallel processor to do the functions that could be highly parallelized. In particular, over time, it became the fastest way of performing neural network training, short of designing a specialized chip. These became known as GP-GPUs (general purpose graphics processing units).

Attached Processors



This table, from Hennessy & Patterson shows how processor performance has improved, first riding Moore's Law and then running into the power wall. From 1985 to 2003, performance improved 52% per year (about 0.8% per week). Then until 2011 it increased 23% per year. But since then, performance increase fell to 10% per year. Looking forward, the number is anywhere from zero to 2%. This means, in effect, general purpose computers will never get (much) faster, every architectural trick we know has been used, and the semiconductor tricks are limited by power. Going from serial to parallel (with effectively as many cores as you want) is a one-time kicker though.

The one flaw in the argument is "general purpose". We know how to make faster processors for almost any given function by optimizing down to the silicon and building a specialized off-load device. As Krste pointed out in his talk, there are 50-60 specialized neural network accelerator startups, and "the list would be out of date by the time I finished writing it." Or, as Chris Rowen likes to say, an AI startup is any startup created since 2015.

Algorithms Change Quickly but Patterns Endure

There is a famous 2006 paper about the Berkeley Dwarfs, officially *The Landscape of Parallel Computing Research: A View from Berkeley* which Krste is actually the lead author of by dint of his name being first alphabetically. A dwarf is an essential element of any computational problem, such as sparse and dense linear algebra, convolutions, spectral transforms, graph traversal, dynamic programming, and more (13 in all). If you build parallel hardware that can run the dwarfs efficiently and fast, then they can run any future program fast. They are the periodic table of parallel algorithms.

Or, as Krste put it:

> *I don't know what future AI algorithms will look like, but they will use these patterns. So design for flexibile composition of instances of these patterns*

Moore's Law is Dead but Amdahl's Law Lives

Amdahl's Law was an observation by Gene Amdahl that the speedup a parallel computer can get is limited by the part that cannot be parallelized. For example, if 5% of the code cannot be parallelized, then the maximum speedup in the limit is 20X (the parallel stuff runs infinitely fast, leaving just the 5% that could not be parallelized). Amdahl's lesson was that scalar performance in a vector processor is really, really important.

The CNS-1 proposal, back in 1992, included:

> A connectionist accelerator can at best speed up an application by a factor of $1/(\textit{fraction of non-connectionist computation})$.

This remains true 25 years later: whatever you don't accelerate will constrain your performance/energy efficiency. The faster the acceleration, the greater this effect. So it is important to work hard on scalar performance and control latencies.

Software Matters Most but You Can Never Finish It

Domain-specific languages such as TensorFlow are a big boon to new hardware, but you still have to actually map the backend onto your hardware. You have to do this without having access to all the software since maybe less than 1% of the software will be finished before tapeout. There is a tendency just to code the kernel 1% and not the other 99%, but then Amdahl's Law can bite you.

But the bigger message is this:

> *If the system is difficult to program, then you will not have software. And if you don't have software, then you don't have an accelerator*

RISC-V

A SiFive presentation by Krste would not be complete without a mention of RISC-V. Originally this was designed as the basis for custom accelerators "so we didn't have to beg MIPS not to sue us." But you can simplify the software by having just one ISA for all cores, whether out-of-order, interrupt responsive, vector extension and so on. Every core will then have the same memory model, same synchronization primitives, same compiler tool flow (with stuff like C-struct packing the same down to the bit level), debugger, tracing, and more.

> *The benefit of doing something slightly better when you lose all this is just not there.*

Krste's Top 3 List

So, distilling an hour of presentation into three bullet points, if you are designing some sort of accelerator for AI then:

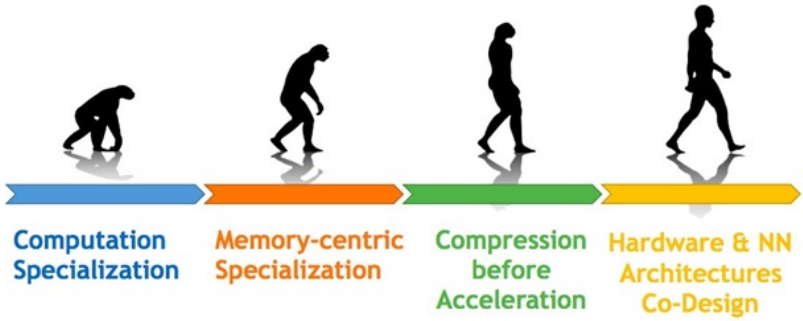
- Design for flexible composition of instances of the dwarf patterns
- Work hard on scalar performance and control latencies
- Make it easy to program or nobody will

HOT CHIPS Tutorial: On-Device Inference

The Sunday of the annual HOT CHIPS (the 30th!) conference is tutorial day. In the morning, it was the Blockchain, which I missed due to other commitments. in the afternoon it was Deep Learning. This was divided into 3 parts:

- Overview of Deep Learning and Computer Architectures for Accelerating DNNs.
- Accelerating Inference at the Edge
- Accelerating Training in the Cloud

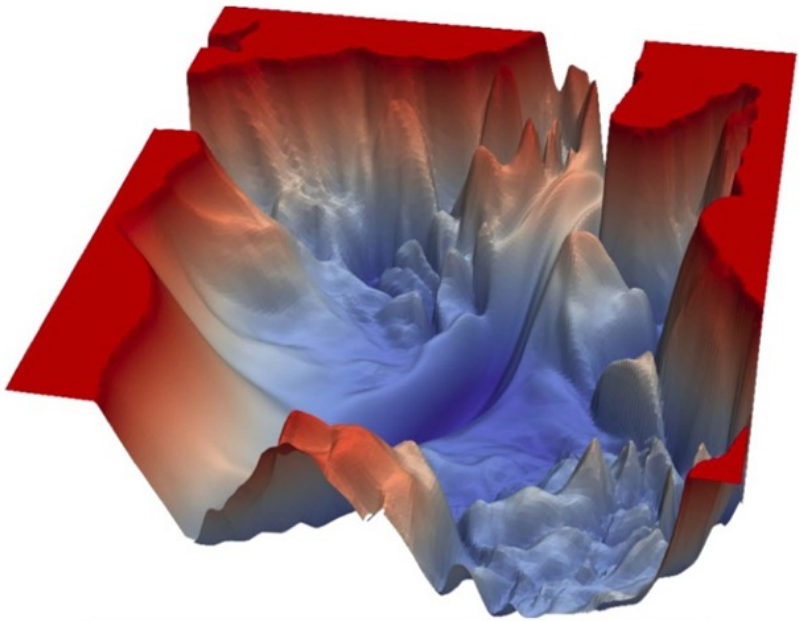
I am going to focus on the on-device inference section since that is most relevant to applications for the higher end entries in Cadence's Tensilica processor portfolio. I'll also pull in some information from the cloud-based segment on benchmarks. This was presented by Song Han of the MIT Han's Lab. Han's Lab is not just using his name, the H stands for High-performance, high-energy-efficient Hardware. A is for Architectures and Accelerators for Artificial intelligence, N is for Novel algorithms for Neural Networks, and the S is for Small Models (for inference), Scalable Systems (for training), and Specialized Silicon.



Obviously, one way you can do a better job of on-device inference is to build a special on-device inference engine. Indeed, during the two main days of HOT CHIPS several were presented from Arm, NVIDIA, Xilinx, and DeePhi...except that a few

weeks ago Xilinx acquired DeePhi, so that one was Xilinx too. But there's more. All the server processor presentations had optimizations for neural network programming. Even the next generation Intel processor, which is called Cascade Lake SP for now, has new extensions to the ISA adding a couple of instructions specifically for evaluating neural nets faster than is possible with the regular instructions. But that is a topic for another day (or two).

Training a neural network almost always takes place in the cloud, using 32-bit floating point. There is a lot of research that shows that you need to keep the precision during training even if eventually you plan to run a reduced model. If you reduce too soon, you miss getting stuck in local minima, or ending up in something that does not converge. Usually, when you see a graph showing a surface representing the space that the training algorithm is exploring, it is a nice smooth saddle where nothing can go wrong. But the picture below is actually more representative:

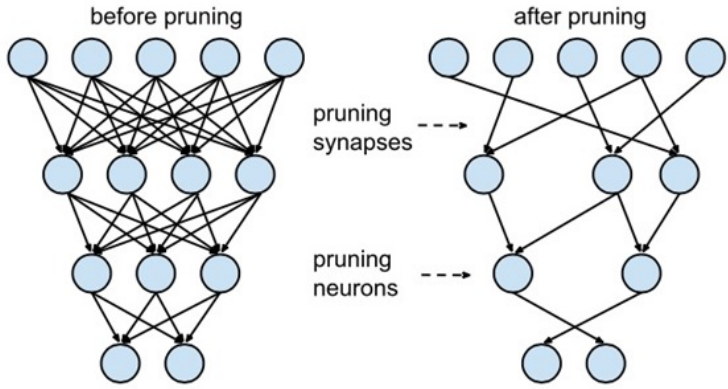


Deep Model Compression

I first saw Song Han speak at Cadence. Back then Song was still doing his PhD on compressing neural networks. Somewhat to everyone's surprise, it turns out that you can compress neural networks a lot more than anyone expected.

Pruning

The first optimization is pruning. The network as it comes out of training has a lot of connections, and many of them can be removed without any loss of accuracy. Once they are removed, the network can be retrained with the reduced connectivity, and the accuracy is regained by retraining and recalculating all the weights. The process of pruning and retraining can be iterated until there is no reduction without too much loss of accuracy.

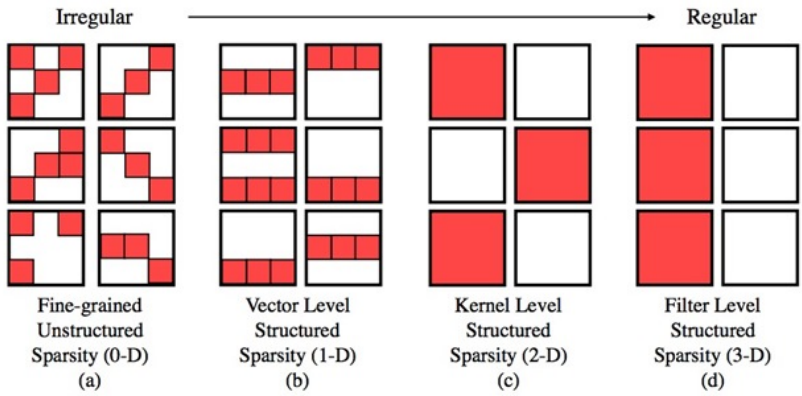


It turns out that the human brain does pruning too. A newborn has 50 trillion synapses, this grows with the brain until there are 1,000 trillion synapses by the time a baby is one year old. But that gets halved back down to 500 trillion synapses by the time that baby is an adolescent. Pruning the neural network this way has a similar effect, and sometimes the pruned and retrained network is not just smaller than the original but has increased accuracy too. Using this approach on AlexNet, the convolutional layers can be reduced by 3X, and the fully connected layers by 10X.

Sparsity

The next technique is sparsity. There is obviously a straightforward optimization when a zero weight is fed into a multiplier since we know that anything times zero is zero. So not only do we not need to feed the zero into the multiplier, we can hold the other input at its old values and save both a memory access and power from toggling the bus. When training, a lot of weights are barely participating in the inference and are close to zero. By setting them exactly to zero, the matrix becomes sparse and all sorts of optimizations are possible.

The sparsity can be unstructured, or it can be structured, as in the diagram below. By using sparsity, a network that looks like it can deliver, say, 1TOPS can deliver 3 TOPS (if you count all the operations involving zero that were never actually executed).



Quantization

Quantization in this context means reducing the width of the weights from 32-bit floating point, to 16-bit, 8-bit, or even lower. It seems surprising that you would not lose a lot of accuracy by doing this, but deep compression really works. Song actually did a lot of the research in this area as part of his doctoral thesis, and found "you could be significantly more aggressive than anyone thought possible."

Putting It All Together

If you do all of this, you get compression ratios as high as 50X. If that number is surprising, then more surprising still is that every

one of the benchmarks that Song talked about had increased accuracy with the compressed networks. Compression is not a compromise, there is clearly a reason mother nature prunes our brains too.

Network	Original Size	Compressed Size	Compression Ratio	Original Accuracy	Compressed Accuracy
LeNet-300	1070KB	27KB	**40x**	98.36%	98.42%
LeNet-5	1720KB	44KB	**39x**	99.20%	99.26%
AlexNet	240MB	6.9MB	**35x**	80.27%	80.30%
VGGNet	550MB	11.3MB	**49x**	88.68%	89.09%
Inception-V3	91MB	4.2MB	**22x**	93.56%	93.67%
ResNet-50	97MB	5.8MB	**17x**	92.87%	93.04%

But wait, there's more...the pruned models accelerate image classification and object detection. This is because the limit on speed (the so-called "roof line") is the hitting the memory bandwidth limit, not hitting the computational limit. By reducing memory accesses, the computation units can be kept busier. This is almost independent of what engine is being used to perform the inference. There really does seem to be only upside to compressing the network: smaller, faster, more accurate.

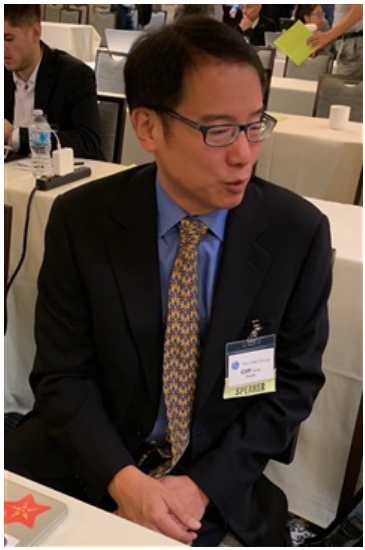
Designing Hardware

Based on the presentations of specialized neural network processors over the following couple of days, I would say that the lesson that everyone has taken away from the work of Song Han (and others) is:

- Train in the cloud at full precision
- Compress the network using the techniques above
- Optimize the inference hardware for sparse matrices, avoiding representing zeros.
- Optimize for MAC operations where one input is zero, and suppress the operation, and the access to the non-zero operand.
- Reduce the precision to 8-bits (or maybe 16-bits) and built lots of 8-bit MACs.

- Don't use caches, you are just wasting area. Be smart about ordering the operations so that values fetched from memory are re-used as much as possible, rather than moving on and coming back to reload the same value later (of course, you can't avoid this completely, but you can be smart, or rather your compiler can be).

Inside Google's TPU



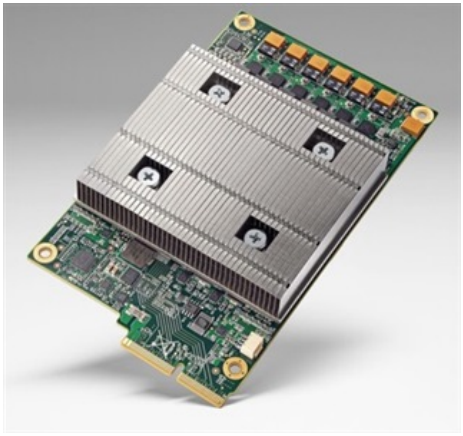
At the Linley Fall Processor Conference recently, the keynote on the second day was by Cliff Young of Google titled *Codesign in Google TPUs: Inference, Training, Performance, and Scalability*. The TPU is Google's Tensor Processing Unit. It is actually a family, since, as Cliff detailed, it is now on its third version. One of the things that makes Google's TPU so interesting compared to other designs is that it is deployed at scale in Google's datacenters. When you use Android or Google Home and say "OK Google" or "Hey Google" then the voice processing is done by TPUs in a production environment. One of the motivations for Google developing TPU in the first place was apparently a calculation that they would need dozens more datacenters if they did all the anticipated voice processing on regular datacenter servers.

Cliff started out by pointing out that every aspect of Google's business, including hardware, is being influenced by AI. There are three generations of the TPU (so far).

As Cliff said:

> *We are building our own chips but we are building them at scale. That's a Google thing, we believe in warehouse computing.*

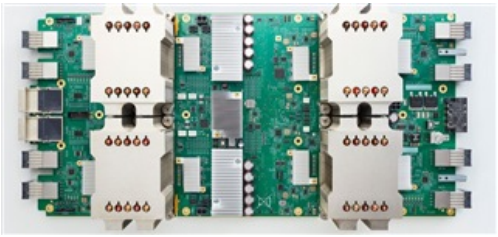
TPU v1



TPU v1 has been in datacenters for 3 years now, since 2015. Google disclosed it after a year in 2016, at Google I/O after AlphaGo had beaten the world go champion. The initial chip was only to do inference to address the potential disaster of speech overwhelming even Google's datacenter capacity. It used 8-bit arithmetic, with a fallback to 16-bit if that turned out not to be enough. There is no floating-point. The original paper talked about 6 product applications, but there are way more than that now. It is also powering research inside Google, so it's not just the speech stuff that is most visible.

ASIC designs like TPU are potentially on a spectrum as regards programmability. On one end you can license Arm and build a sea of cores, and at the other end you can build a fixed-function block. Google picked a place in the middle for TPU, with what they thought was a good balance between programmability and performance. In particular, in a field where the ground is shifting almost daily, there needs to be the right amount of flexibility to handle future models, about which little is known. The TPU v1 was designed in 15 months and it has 15-30X the performance of contemporary CPUs, and 30-80X the performance per watt of contemporary CPUs and GPUs.

TPU v2

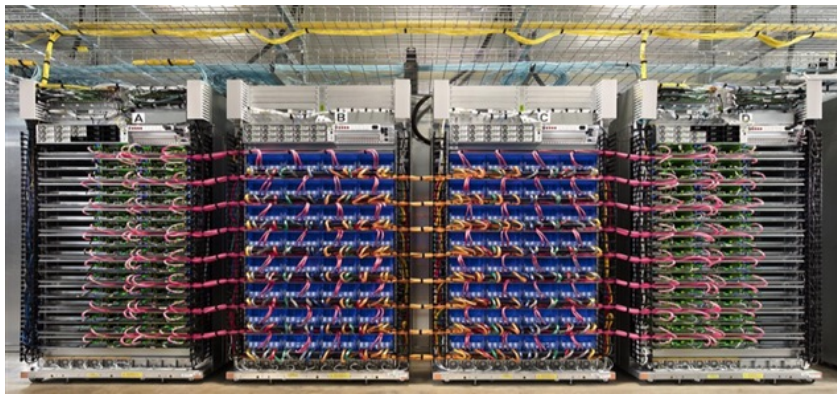


The TPU v2 is generally available, in the sense that you can rent time on it through Google cloud. It has 180 teraflops of computation, 64 GB of HBM memory, 2400 GB/s memory bandwidth. It is designed to be connected together into larger configurations.

It also is designed for both training and inference. It still has 8-bit but also has floating point. Whereas TPU v1 was designed like an old-style GPU to be a co-processor to a host CPU, v2 is designed to be networked (although still connected to a host server).

They kept the matrix multiply unit from v1 but also added a general purpose scalar unit and a general purpose vector unit. Cliff said that in some ways it is like an old Cray supercomputer. Cray had the best performance scalar units on the planet since, due to Amdahl's Law, a major limit on overall performance is the part of the code that cannot be vectorized.

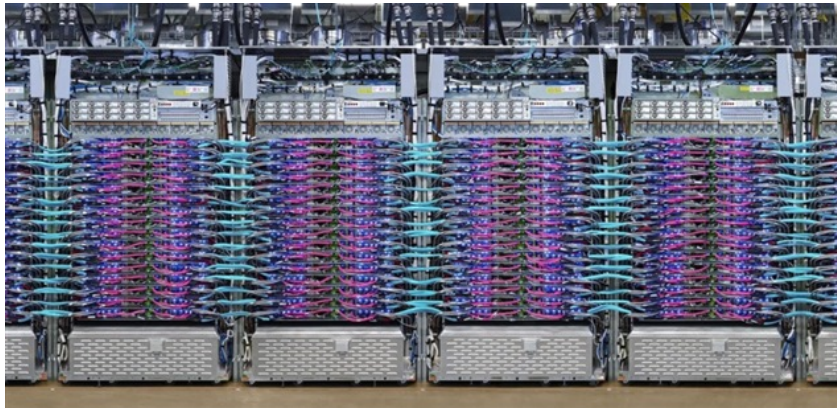
The v2 chip has 2 cores, with 22.5 TFLOPS per core. There are 4 chips per 180 TFLOP Cloud TPU unit. Mostly it uses 32-bit floating point. But it also supports bfloat16 (the "b" stands for "brain"). Normal IEEE fp32 has an 8-bit exponent and a 23-bit mantissa. The IEEE fp16 half-precision has just a 5-bit exponent and a 10-bit mantissa. It was designed with graphics in mind, and is good for high-definition rendering. But for machine learning, it is the wrong tradeoff, since the sum of tiny differences is important. So Google invented a new floating point representation, bfloat16, which has the 8-bit exponent of fp32 but just 7 bits of mantissa. In the MACs, the multiplies can be bfloat16 and the addition fp32. In the Q&A, it was pointed out that Intel has committed to support bfloat16.



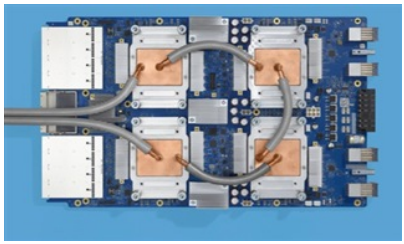
The cloud version of v2 has been available since last year, the TPU v2 pod is now in alpha, built into supercomputer configurations. The TPUs are in the blue section in the middle of the above picture. It has 11.5 petaflows, 4 TB of HBM, and a 2-D toroidal mesh network.

As I said above, the TPU v2 was designed for training as well as inference. Training is about 3X the computation of inference: forward propagation (as in inference), back-propagation, and weight update. There are also much longer data storage lifetimes, which puts more pressure on memory capacity and bandwidth. There are huge training datasets, and more regular changes to algorithms and model structure, which requires even more flexibility.

TPU v3



Cliff could only say "a few things" about TPU v3. It is liquid cooled (as you can see from the picture). The pod size is increased to 8 racks, as you can see from the above picture revealed at this year's Google I/O. It is currently in beta. In the Q&A later, Cliff was asked whether the connections were optical. His carefully worded response was:



> *I don't think we've disclosed that so I'm not going to say. But if you looked at the box at Google I/O I can say they were electrical connectors. How's that?*

The performance is 420 teraflops (per unit) with 128 GB HBM. The v3 pod has over 100 petaflops, 32 TB of HBM.

In the Q&A someone asked how important it was to scale beyond 256 nodes. Cliff said:

> *We already have. TPU v3 is more, I think we said 1024. We do look at different network topologies, all the stuff in supercomputer design, and even datacenter design. But demand from the brain side is "a teraweight machine". There is a collision between how many pods we can fit in a datacenter and how many datacenters can we buy.*

Edge TPUs

Google has announced their edge TPUs. However, they haven't announced the specs yet, and Cliff wasn't talking. They are the tiny chips on the cent coin in the photo below on the right.

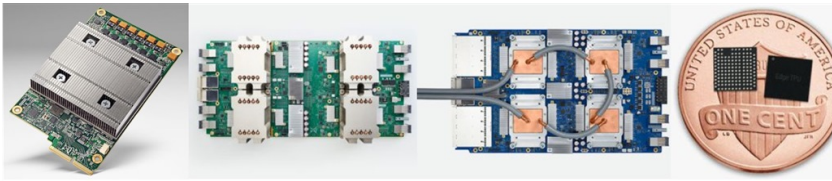
One their website, all Google says is:

> *Edge TPU is Google's purpose-built ASIC designed to run AI at the edge. It delivers high performance in a small physical and power footprint, enabling the deployment of high-accuracy AI at the edge.*

Summary

Google have been making "relentless progress":

- TPU v1, deployed 2015, 92 teraops, inference only.
- TPU v2, cloud TPU 2017, pod 2018, 180 teraflops, 64 GB HBM, training and inference, generally available. 11.5 petaflops in a pod.
- TPU v3, cloud beta 2018, 420 teraflops, 128 GB HBM, training and inference, beta. >100 petaflops in a pod.



Codesign

Cliff started talking about instruction set architecture (ISA) which is a contract between the hardware and the software. A classic tradeoff from the 90s was whether the scheduling should be done in the compiler (at compile time, obviously) or in the hardware (at runtime). The compiler approach led to VLIW machines like Itanium, the hardware approach led to out-of-order (OoO) execution. The answer really turned out to be both: build OoO hardware but also do careful instruction scheduling.

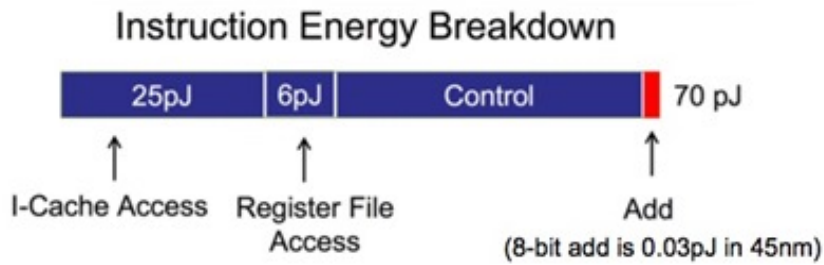
> *The codesign debate ranged for 15-20 years, and that was just a single interface. Today, we have domain-specific areas. We probably want a compiler. We probably want libraries since the people who build the machines know best how to use them. We need to codesign from the physics all the way up to the applications, which is hard.*

Cliff said that there are three big codesign themes in TPU design:

- Systolic arrays.
- Reduced-precision numerics.
- Pods as tiled architectures at datacenter scale.

Cliff said that systolic arrays are a two-dimensional generalization of a pipeline. They have entirely local communication "which is great, since we can't even get through 1mm of interconnect within one clock cycle." The basic systolic technology was invented in the 1970s but "they only got to have

8 ALUs, whereas today we have 64K. It just wasn't a good match for the implementation technology of the time." Like a heartbeat, systolic arrays have alternating computation and communication phases.



Systolic arrays go a long way to address computing's energy problem. An 8-bit add is just 0.03pJ in 45nm, yet an add on a general purpose CPU takes 70 pJ, which is 2,000 times as much.

I pulled Cliff's discussion on reduced precision numerics, in particular, bfloat16, into the earlier section about the hardware implementation in TPU v2. But he emphasized that bfloat16 is a good example of codesign, requiring support all through the system from the hardware, through the compiler, libraries, and perhaps up into the neural network frameworks.

Pods require TPUs to be designed from the beginning to be networked. The TPUs are tiled within the chips but those chips are then tiled to build supercomputers. The big challenge is data parallelism, taking the weights and sharing them across multiple TPUs, which can kill you if you get the interconnect wrong. There is an upcoming paper at NIPS 2018 discussing this, with the goal of handling billion and trillion parameter models, exploiting the fact that there are 4TB of HBM per pod. Then run the same single-program-multiple-data (SPMD) on each core. The approach apparently shows accuracy improvement.

Performance

There is a challenge measuring and comparing performance. Benchmarking is not where it needs to be. As Cliff put it, people tend to say something like:

> *Hey I got a resnet-50, might not be the same code as yours but we'll call it resnet-50 anyway, and here are our numbers.*

Machine learning performance needs more, and need reproducible benchmarks via open-source implementations.

DAWNBench was proposed by Stanford as both a benchmark and a competition, with new results in April. It covers both training and the cost of training.

The Cloud TPU on one benchmark can train in about 8 hours (not uncommon to be multiple GPU *weeks*) at a cost of under $50 by the DAWNBench cost metric. That has improved over the last 6 months, and is now down to $35 (some due to improved software, some due to lower rental costs). For pre-emptible cost it is $11. Pre-emptible cost means you might get thrown off the TPU if someone higher priority comes along and needs it. That's nice, but as Cliff put it:

> *That the same hardware can get twice as fast in a 6 month period means we haven't quite worked out how to best use our machines.*

The pod numbers are very fast. A single TPU v2 can train ResNet-50 at 3250 images per second, for a full training time of 9 hours and a coast of $59. That is down to just over 8 hours today, for as little as $13 pre-emptible cost. On a TPU v2 half-pod, it can process 77,392 images/second, for a training time of just 30 minutes (24 minutes without checkpointing). With some clever tricks that have come along since the DAWNBench competition, they have got ResNet training down on a TPU v2 from $40 to $17, and with pre-emptible pricing down to $5.

MLPerf

MLPerf is a broad ML benchmark suite for measuring performance of ML frameworks, ML hardware accelerators, and ML cloud platforms. It is driven by researchers from Harvard, Stanford, UC Berkeley and more, supported by over 30 companies such as Google, Intel, NVIDIA, Arm, AMD...and I

see from the website, EDA. All three of Cadence. Mentor and Synopsys are supporting companies.

The basic idea is to build SPEC-type benchmarks for machine learning. The philosophy is:

- Agile development because ML is changing rapidly.
- Serve both the commercial and research communities.
- Enforce replicability to ensure reliable results.
- Use representative workloads, reflecting production use-cases.
- Keep benchmarking effort affordable (so all can play).

Takeaways

Danger: the end of all three of Moore's Law, Dennard Scaling, and standard CPU performance architectural tricks. The limits of the CMOS are in sight, and Intel's 10nm challenges and GF's 7nm exit are signs.

Opportunity: there is a revolution in machine learning, with economic demands for ML accelerators, architectural and codesign experimentation. Perhaps we can use ML to design better ML accelerators.

Irony: exponential demand for ML computation just at the end of Moore's Law. Efficiency will thus matter a lot. There are huge opportunities for HW/SW codesign in building TPUs and other domain-specific-architectures (DSAs).

Chapter 4: 5nm and EUV

If It's Tuesday this Must Be Belgium. My First Visit to imec

Outside of semiconductor, Belgium is famous for three things: beer, chocolate, and fries (which they eat with mayonnaise). Inside semiconductor, it is famous for imec. It's also famous for the 1969 movie *If It's Tuesday, This Must Be Belgium.* And it is, indeed, Tuesday.

After CDNLive in Munich (on Tuesday, to keep with tradition) I flew to Brussels and took the train to Leuven. After 35 years in the semiconductor business, it is hard to believe that it was the first time I visited. I've been to ITF, the Imec Technology Forum, a couple of times, but they hold that in downtown Brussels, not out at Leuven.

Getting There

In case you ever have to visit imec, here's the easy way to get there. Fly to Brussels. If you need to stay overnight, stay in the Sheraton which is literally across the street from top level of the airport terminal. There is a train station in the basement of the terminal. You can buy a ticket at the machines to Leuven (although I had to type it as Louvain, its French name). They run roughly every half hour. At Leuven, take a taxi from outside the station to imec. If they ask you where at imec you want to go, tell them "the tower" since that is where the visitor reception is.

Imec History

When I lived in France, imec existed. It was created in 1984. However, it was more European-focused than it is today, with its main partners being the three big European semiconductor companies:

- Siemens, who spun their semiconductor operations out to create Infineon and Qimonda (RIP)
- Philips, who spun their semiconductor operations out to create NXP

- ST Microelectronics (although, if you go back far enough, SGS and Thomson Semiconducteurs pre-merger)

Back in that era, most semiconductor companies had a research fab, often called the TD fab since the generic-sounding "technology development" has the specific meaning of developing process technology inside a semiconductor company. But as the cost of fabs, fab equipment, and the investment to create a process rose, there was a need for a research environment with a fab that could allow semiconductor companies and equipment companies, to cooperate in the pre-competitive time-frame. There can only really be one roadmap for the industry since the equipment and material suppliers have to line up behind it. That's not possible if each semiconductor company develops its own esoteric flavor or processes requiring different things from its suppliers.

Imec is in Belgium because that's where it started. But it is neutral ground, in the same way as it is a neutral country for the headquarters of the European Union. If it had been in the US, Taiwan, Japan (and later Korea or China) then it would not have worked. SEMATECH was a similar operation in its early days, except its mission was explicitly to regain US competitiveness after Japan had driven companies like Intel out of the memory business, and looked like it might dominate the entire semiconductor business. But even later when it threw its doors open to non-US companies, it was still in the US and so not seen as independent.

Today, all the leading-edge semiconductor companies in both logic and memory have joint programs with imec. They each send a couple of dozen engineers, over 500 "guest" engineers in total, to work on programs there. Cadence has a couple of people there, as do many of the equipment vendors, and other contributors to the ecosystem.

My Visit

I met first with Luca Matti, who somehow has three hats, working at imec, while finishing his Ph.D. at Braunschweig University of Technology, and being part of the Cadence Academic Network. His work is on DTCO, Design-Technology

Co-Optimization. He was also involved in the recently announced Cadence-imec 3nm testchip. Luca and Diederik Verkest gave me a bit more detail. The chip was not a whole chip, just a large block containing an Arm processor. However, the whole block was laid out and went through timing closure. That way, every aspect of the block was realistic. However, only a few masks were made since the main purpose was to investigate manufacturability of the tightest layers.

Diederik gave me imec's view on what the future roadmap for the semiconductor industry will be. Since the leading semiconductor companies who all have staff at imec, this is really the consensus roadmap.

I was taken on a fab "window tour". That means I got to see into the fab from outside, but I didn't suit up in a bunny suit and go inside. They have just extended the fab with an expansion area which is now being filled with equipment. The total investment is over €1B and I suspect that mostly covers the shell, air conditioning, water conditioning, and so on. Most of the equipment is also part of research programs.

The most interesting piece of equipment was, of course, "the beast", the ASML 3300 EUV stepper. It was especially interesting to see since it was undergoing maintenance and so most of the covers were off revealing the innards. I'd love to show you a picture of it, but it was too interesting—I wasn't allowed to take photos of that part of the fab.

Memory

Arnaud Furnemont gave me a presentation about where imec sees the future of memory. His summary was "it's all 3D NAND and DRAM". That's for standalone memories. For embedded memories, some other technologies are attractive. Arnaud said that a few years ago it looked like RRAM was going to take over the world, with performance close to DRAM and non-volatility. But the current required to make the filaments stable seem to be too large and so we have three memory technologies for standalone:

- 3D NAND. The number of layers will continue to increase, but nothing dramatic is expected to change.
- Storage class memory, which is the generic name for Intel/Micro 3D Xpoint. But that has been slow in really getting to market, and there are no other suppliers of similar technology.
- DRAM. It will continue to improve slowly at about 1.1X per year. Adding on-chip ECC is helping since it makes it easier to manage the fact that some of the transistors on a memory die will be a long way from the mean since there are billions of them.

The big memory challenge is that demand for memory, especially from mobile and IoT, is increasing faster than the memory technology. Looking at the big picture, that means that the only way to scale fast enough to keep up is to build more fabs. China is building lots; in particular, a NAND flash, a NOR flash, and a couple of DRAM fabs. But even looking at the entire world, we can't get onto a sort of Moore's Law of fabs where we have to build twice as many memory fabs each year to satisfy demand. But there don't seem to be any breakthrough technologies on the horizon even in research, which means nothing in volume manufacturing for five to ten years.

In embedded memories, MRAM is the most promising technology, although RRAM also has a place. eFlash is too expensive in terms of masks, so both MRAM and RRAM are attractive. They don't involve so many masks, and they live in the BEOL so it is easy to move from one transistor-level technology to another. This fits with what the foundries seem to have on their roadmaps.

SEMICON 5nm: 7nm Is Just a Dress-Rehearsal

Usually I don't go to the last day of SEMICON West since not much happens that day. But they have got smart, and two of the most interesting sessions took place in TechXPOT (I think you pronounce that Techspot) on Thursday. In the morning, it was *Lithography at 5nm and Below*.

This session featured:

- Eric Hoster of GLOBALFOUNDRIES on *EUV Lithography: The Next Generation*
- Michael Lancel of ASML on *Lithography for the 5nm Node and Beyond*
- Stephen Renwick of Nikon on *Advanced Nodes with 193i Lithography*
- Christopher Progler of Photonics on *EUV Mask Insertion: Confident or Compromise?*
- Mary Ann Hockey of Brewer Science on *What is Really Happening with DSA?* [directed self-assembly]
- Neeraj Khanna of KLA Tencor on *Addressing Process Control Challenges for the 5nm Node and Beyond*
- Angelique Riley of TEL on *Novel Patterning Schemes and Technologies for the Sub-5nm Era with a Focus on EUV*

There was obviously more material from that array of experts than I can possibly cover in a single section (or even several).

The first presentation, by Eric Hoster, who is GF's EUV Lead Technologist, is where I'm going to focus. He works for a foundry so is a comparatively independent observer. The next two presentations, by ASML and Nikon could be summarized as "everything is going to be okay", and "we've got your back if they're not." I'll drop in appropriate statements from other speakers, but Eric's sober assessment of how much further we need to go to make EUV work for 5nm, even given that it's working(ish) at 7nm was fascinating.

Eric Hoster

Eric opened with a great quote:

> *We stand on the precipice of EUV insertion at 7nm. But 7nm is just the dress rehearsal for the next generation, which will be incredibly complex.*

He recalled the i-line to KrF transition, which involved the introduction of chemically amplified resists (CAR), arc lamps changed to excimer laser, and new pellicle materials. It was disruptive. The transition to the second generation of KrF lithography was much more evolutionary. But he warned that the transition to the second generation of EUV will not be like that.

As he carried on to say:

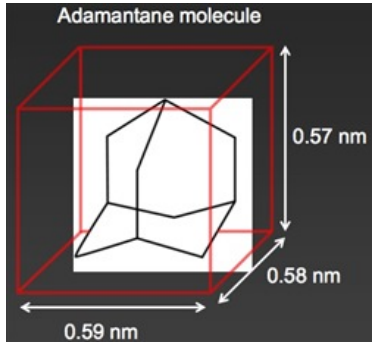
> *We've solved a lot of challenges over the last 30 years to get to this point. Now we need to look at what we need in the future.*

2014	2015	2016	2017
1.Reliable source operation with > 75% availability	1.Reliable source operation with > 85% availability	1.Reliable source operation with >85% availability	1.Reliable source >250W operation with >90% availability
2.Resist resolution, sensitivity & LER met simultaneously	2. Resist resolution, sensitivity & LER met simultaneously	2. Resist resolution, sensitivity & LER met simultaneously	2. Resist resolution, stochastics, and sensitivity met simultaneously
3.Mask yield & defect inspection/review infrastructure	3. Mask yield & defect inspection/review infrastructure	3. Keeping mask defect free	3. Keeping mask defect free
3. Keeping mask defect free	4. Keeping mask defect free	4. Mask yield & defect inspection/review infrastructure	4. Mask yield & defect inspection/review infrastructure

2014	2015	2016	2017
1.Reliable source operation with > 75% availability	1.Reliable source operation with > 85% availability	1.Reliable source operation with > 85% availability	1. Resist resolution, stochastics, and sensitivity met simultaneously
2.Resist resolution, sensitivity & LER met simultaneously	2. Resist resolution, sensitivity & LER met simultaneously	2. Resist resolution, sensitivity & LER met simultaneously	2. Reliable source >250W operation with >90% availability
3.Mask yield & defect inspection/review infrastructure	3. Mask yield & defect inspection/review infrastructure	3. Keeping mask defect free	3. Keeping mask defect free
3. Keeping mask defect free	4. Keeping mask defect free	4. Mask yield & defect inspection/review infrastructure	4. Mask yield & defect inspection/review infrastructure

The grid on the left shows the ranking of the issues over the years for the introduction of the first generation of EUV (which is turning out to be 7nm, although everyone assumed it would be much earlier). The top worry was getting a reliable source, which seems to be on-track. But the chart on the right shows the ranking of the issues for the next generation EUV for 5nm, where the big challenge is to meet sensitivity, stochastics, and resolution simultaneously. Stochastics are pretty much the new name for line-edge-roughness (LER) since there are other things involved now. But the main tradeoff remains the same: sensitive resists, and high doses, cause big sochastic problems. Low doses, or low sensitivity get lower stochastics, but the throughput is uneconomical. If you want to go fast, you are going to have a stochastic problem.

The problem for 5nm and beyond is that as you increase the resist dose, you drive down the defect density. But as you go to smaller resolution, the same number of photons are in smaller area so we need to double the sensitivity of the photoresist. If we can't do that, and we probably can't do it enough, then, "we're gonna need more photons". The effective dose has to increase a lot at every node: 40 mJ/cm^2 at 7nm, 60 mJ/cm^2 at 5nm, and 120 mJ/cm^2 at 3nm.



As Eric pointed out, "we are victims of our own success since the molecular nature of matter is now relevant". For example, the photoacid generator (PAG) molecules in CAR have a spacing of 1.7nm in a high PAG-loaded resist. The variability can lead to notching and scumming. The adamantane molecules in the resist are 0.5nm cubes, in a process where the overlay budget (alignment error between different masks) is just 2.5nm. There are some hopes for the future, such as metal oxide resists and silsesquioxanes, which are very regular.

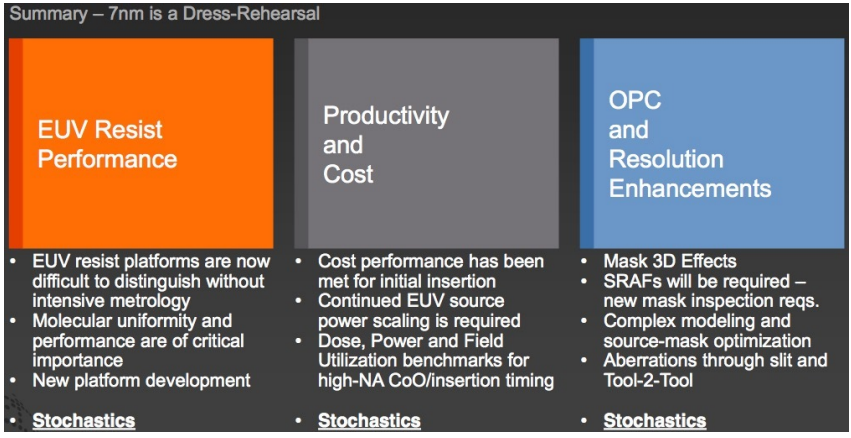
Another resolution issue is known as scanner fleet variation: at these resolutions, every EUV scanner is a little different due to aberrations. For example, the overlay between one layer and the next is typically much better if both layers are exposed on the same machine than on two different ones, but in practice in HVM you don't have that luxury.

Next Eric moved on ot look at productivity and cost of EUV. EUV is more cost-effective than optical triple-patterning for now. The next generation high-NA EUV machines will have very high cost-of-ownership (and they are two stories tall, so also impose costs on the fab construction). The tradeoff is between EUV LELE (litho-etch-litho-etch, ie double patterned EUV) versus high-NA next generation EUV, which might survive with single patterning for another node. He had a lot of detailed tables that I'm not going to reproduce here, but his conclusions were:

- <70 mJ resist dose is needed
- <350 shots per wafer is needed
- >350W source power is needed
- lithography performance will remain of paramount importance

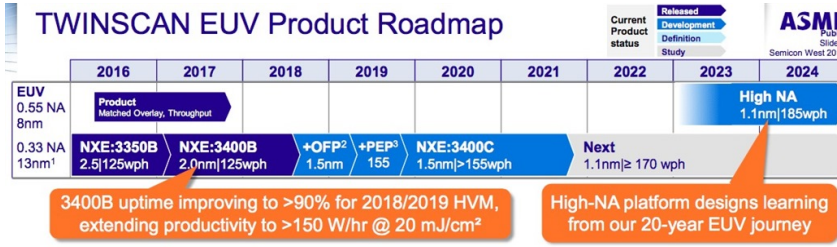
Finally, he moved onto OPC/RET (optical proximity correction, resolution enhancement technology, basically working out how to put different patterns on the mask so that it prints better). This is going to get lot more complex. RET has actually got a lot simpler, first with double patterned 193i which required less than the last generation of single patterned, and EUV required even less. But we can't get to 5nm and beyond without SRAFs, sub-resolution assist features. It will requires "OPC on steroids." However, model-based implementation of SRAFs has proven difficult even in optical lithography due to computational cost, shape complexity, and output file size. EUV further adds complications of minimizing pattern displacement error and best focus shifts.

Eric's finally summary slide. Just like the three most important things in real estate are location, location, location, so the three most important things in next-generation EUV are stochastics, stochastics, stochastics.



Michael Lercal

I won't cover all of Michael Lercal's presentation. But he works for ASML, the only company to manufacture EUV products. So their future roadmap is very important. Here is the one timeline summary:



TSMC Technology Symposium 2018

This week it was the TSMC Technology Symposium in Silicon Valley. Dave Keller, president of TSMC North America was the MC for the day.

Dave kicked off by giving a few statistics about TSMC's business in North America. In Q4 last year, North America was 71% of TSMC's global revenue. The region consumed 6.1M 12" wafers (or equivalent—since it works by area an 8" wafer is half a 12" wafer, and a 6" wafer is a quarter).

Dave introduced CC Wei who is currently co-CEO of TSMC (the other co-CEO being Mark Liu). He is in transition to being the full CEO. This was all announced along with Morris Chang's full retirement to take place at the annual shareholders meeting in June this year. At that point, Mark Liu will become chairman of the board, and CC Wei will become CEO. Morris will no longer be on the board, nor have any role in management. I will just say, since it is a theme that ran through the whole day, that TSMC sees the two big trends that drive everything as being 5G and AI. Obviously mobile, but also automotive, IoT, and HPC, which are

the four focus areas that TSMC has talked about for a couple of years now.

All through the following sections, you will see comparisons of processes by speed and power. If I say that a process is 10% faster, 35% less power then that means either you can take the improvement as performance, or you can take the improvement as power-saving, or as some mixture of the two. But you don't get to have the full 10% increase in performance at the same time as 35% less power.

YJ Mii

YJ is the head of what is usually just called TD, technology development. This means semiconductor process technology development. It seems to be obligatory to have a very short name to head up TD—when I was at VLSI Technology our head of TD was Ho Yu, and it's hard to have fewer than 4 characters in your first and last names together.

YJ said he would talk about 5 subtopics: N7, N7+, N5, EUV, and research.

N7

N7 is, of course, the first generation of TSMC's 7nm process. YJ said that it has passed all qualification, is in volume production now, with over 50 tapeouts planned by the end of the year, roughly a third in mobile, 2/3 in HPC, and a couple of automotive.

Compared to 16FF+ it has a speed increase of 35%, a power decrease of 65%, a routed gate density 3.3X higher, and SRAM cell size 0.37X.

N7+

N7+ is a further density boost, with 20% logic density improvement and 10% power reduction. This comes about since N7+ uses EUV for some layers. More details on EUV came later in YJ's presentation. N7+ is already demonstrating the same SRAM yield as N7, and has passed all wafer reliability checks. Customer product tapeouts are expected in 2H 2018 with risk production starting in Q3.

N5

N5 will be the best technology in 2019 "further extending TSMC's technology leadership". Compared to N7, it is 15% faster with the normal transistor, or up to 25% faster with a new ELVT transistor (Extremely Low Voltage Threshold). Power is reduced by 30%. Logic density is increased by 1.8X. It is a full-fledged EUV process, not manufacturable by immersion, with 30% fewer masks than N7.

The 256Mb SRAM yield has already reached double digits (without repair), which is ahead of schedule.

Risk production for 5nm is planned in 1H 2019 primarily for HPC (and high-end mobile).

EUV Progress

TSMC are very confident to deliver EUV for high-volume manufacturing in 2019. There are multiple EUV layers implemented in both N7+ and N5. They are continuing aggressive installation of NX3400 steppers (the current state-of-the-art ASML product) for volume production.

TSMC is running the light source at 145W in daily operation since the start of the year. They demonstrated 250W just two weeks before in April. This is a huge milestone, since for several years everyone has said that a 250W light source is necessary for volume manufacturing, since otherwise the wafer throughput is too slow. The other aspect of throughput is the sensitivity of the resist. TSMC has significantly reduced resist dosage and is on-target for 1Q19 production schedule.

Another EUV challenge has been pellicle transmission. Without a pellicle on the mask, any contamination risks messing up a lot of wafers before it is detected and cleaned. The pellicle is more complex with EUV for two reasons. First, since it is reflective optics, the light path goes through the pellicle twice. Second, the 13.5nm EUV light is absorbed by almost everything (even air, which is why an EUV stepper is in a high vacuum) and so there are not very many materials that are practical. YJ said that they now have pellicle transmission at 83% with a target of 90% in 2019.

Once the EUV lithography is used to create structures, it has better process control (smaller variation), better CD uniformity, and better pattern fidelity. They are getting good yields in both N7+ and N5 with multiple EUV layers. A single pattern EUV can replace 4 immersion litho layers.

Research

Beyond FinFET, TSMC is looking at nanowires, in particular stacked horizontal nanowires. YJ had a micrograph of a gate with 8 separate wires running through the gate. There is also the opportunity to go to nanosheets, where the wires are elliptical not circular, which makes it possible to fine-tune the device widths. This is important technology for pushing transistor scaling to 2nm and beyond.

Then, beyond silicon itself, is Ge (germanium). Germanium enables lower power at the same performance compared to a silicon transistor. They demonstrated for the first time Ge CMOS-compatible gate-dielectric and world-record low contact resistance. A TEM image showed the capability to produce Ge nanowires.

Future transistors might exploit 2D material. These exhibit high mobilities down to sub-nm thickness. For regular materials, line-edge-roughness (LER) is an increasing problem. What currently seems an attractive 2D material is MoS_2 (molybdenum di-sulphide) which has good carrier mobility down to the nanometer range.

For interconnect, they are looking to extend copper with giant grains. YJ showed a cross section with Cu grain 10X normal size. This can reduce metal resistance by more than 50%. They can also do selective dielectric-on-dielectric (DoD) deposition, enabling full self-aligned Cu vias to increase reliability, and a 50% improved via-to-line breakdown voltage.

Ultra Low Power

TSMC have several processes that go under the ULP (ultra-low-power) naming, and also ULL that extends the ULP process with ultra-low-leakage transistors. For the advanced (post 28nm) nodes, there is N22 ULP and ULL with a wide range for adaptive

voltage scaling from 0.9V to 0.6V. He showed a graph of a design at 90LPP using 35 units of power (presumably not watts!) at 1.2V, going to 40ULP at 0.7V where the units went down to 10, and then to 22ULL at 0.6V cutting power to 3 units, so from 90 to 22, a 90+% power reduction.

The ULP technology will be extended to 12FFC_ULL using a low-power FinFET. Note that 12FFC is an optical shrink of 16FFC, so is a sort of half-node between 16 and 7.

Memory

In emerging memory, they are looking at all three of MRAM, RRAM, and PCRAM (magnetic, resistive, and phase-change). These are all built in the back-end (in the metal stack). I will say more about memory when I cover the afternoon sessions, since there was more detail of what will be available when.

Of course, flash is not completely forgotten. They have the leading 28nm eFlash for high-end Grade-0 applications, built on top of 28HPC+. Grade-1 qual is planned to complete in 1H 2019, and Grade-0 qual in 2H 2020.

Both RRAM and MRAM have made great progress. 40RRAM is ready for production, as a flash alternative for IoT. It has completed qual. Unlike adding flash, it is just two extra masks.

22MRAM will be ready in December 2018 as an eFlash alternative for mobile, HPC and automotive. It has superior performance to eFlash, with 3X faster write, 10 years retention after a million cycles at 150°C.

MEMS

TSMC has been in MEMS for several years, starting in 2016 with a motion sensor (over 350M units shipped), then in 2017 with a capacitative pressure sensor. In Q3 2018 they will have a WLCSP sensor SoC.

High Bandgap

TSMC is focusing on GaN (gallium nitride) as opposed to SiC (silicon-carbide). YJ said they have "GaN on silicon leadership" with GaN power 100V and 650V driver integration by 3Q 2017

with better performance, smaller form factor, and more effective power control.

They also have GaN RF with a 30V D-MISFET qualified for RF switching in 2017, 100V D-HEMT to be qualified in 2019 for 4G/5G RF power amplifiers.

Advanced Packaging Technology

There was a lot more discussion on advanced packaging in the special session in the afternoon. But YJ's summary was that:

- CoWoS is in the 7th year of volume production
- InFO PoP is in 3rd year of volume production

CoWoS is moving to larger interposers and finer Cu bump pitch to house more functionality and performance. CoWoS with 1X reticle size and 130um bump pitch will be in production later this year, going to a 2X reticle size interposer in 2019.

There is also innovation going on in the InFO and chip-level 3D IC solutions to enable heterogeneous integration with small form factors for mobile, IoT and HPC. In particular InFO_MS (memory on substrate) on a 1X reticle size fanout with 2um RDL interconnect. This will be qualled by end of 2018.

For mobile, there is InFO PoP with or without backside RDL and optional BSFDL to match DRAM package footprint (over the top of the SOC) and including integrated passive devices (IPD) for better performance.

Multi-stacking got its own acronym MUST (MUlti-layer-STacking). This is TSV-free 3D stacking for small form-factor and enhanced performance. You can have an SoC on the bottom, with one or more on top, and TIV going down around the base SoC for connections, without any TSVs going through the base die.

For radios, there is also InFO_AIP (antenna-in-package) with two different types of antennas available depending on whether it is millimetre-wave or lower frequency.

Finally, YJ introduced SoIC (system-on-integrated-chips). This has innovative stacking of multi-chips with very fine pitch (10um) using a wafer bonding process. It is compatible with many package types such as flipchip, InFO and CoWoS. He showed one picture that had two chips of identical size, but also one with two small chips on a larger chip (which obviously cannot be done with two whole wafers).

Samsung Foundry Forum: 10, 8, 7, EUV, 5, 4, GAA, 3...

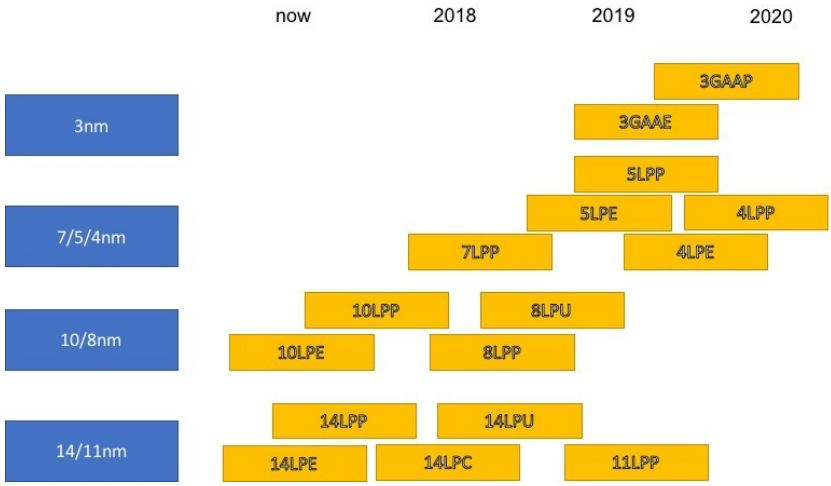
Last week was the Samsung Foundry Forum. Almost exactly a year ago, Samsung reorganized so that foundry was a standalone business, a part of device solutions (along with memory, and system-LSI). As you might expect, US contributes the most revenue to foundry. There are two new fabs, S3 and S4, in operation, and an EUV lithography line under construction. They passed all their customer audits in the last year, 20 of them, with 9 in automotive.

If I had to sum up the day's presentations, it was that Samsung have a very ambitious process roadmap, and also put a lot of effort and investment into SAFE, the Samsung Advanced Foundry Ecosystem, to ensure that IP, tools, and flows are available when the processes are available.

Samsung, like other foundries, won't allow us to take any pictures and don't give us any handouts. I have tried to recreate from memory and notes the master process roadmap slide (for FinFET, FD-SOI has its own roadmap). First, a word about naming. Samsung take most processes through several generations through "smart scaling". The first generation of the process has a suffix E (for early). Then P (for perfrormance), then the third generation has C (for cost-reduction), and finally a fourth generation with U (for ultimate). All four versions don't always exist, but that's the decoder ring to understand the process names

that this section is flooded with. The business strategy is that the first process is introduced for high-volume mobile customers (that's the E process). Then it is performance enhanced for HPC/networking (that's the P process). Then it is cost-reduced for consumer markets (the C process). Finally, the performance is enhanced again (to the U process). Often, there is also a shrink, to a slightly lower number. Most of the design rules remain unchanged (apart from the shrink factor).

The Master Roadmap and Summary



The left-hand edge of the boxes is roughly when preliminary PDKs should be available. The right-hand side doesn't mean anything (the boxes are just the same size). In particular, the 3nm GAA processes are scheduled for risk production in 2021.

Specialty (some 12" but mostly 8") is expanding from logic to eFlash, power, display driver, CMOS image sensor (CIS), fingerprint sensor, power discrete, and MEMS.

EUV is going into fab S3 in Korea. It will be ready for risk production in the second half of 2018. They have their own EUV mask-debugging infrastructure. Their wafers per day is over 800, on track for 1000 by end of year. They have demonstrated the illuminator at 250W, so they are on track for 1500 wafers per day (in 2020), which is what is the goal for economic volume

production. They will initially do EUV production with no pellicle, but they are working on pellicles and will eventually use them (my guess is that they can get away without a pellicle for contact/cut masks, which are mostly dark, but not for lines). Other aspects of EUV such as single line open, and improved photo-resist, are all on track too.

The 5 big fabs are S1, S2 (Austin TX) running 65-14nm, S3 running 10-7nm with a dedicated EUV line. S4 is dedicated to CIS production. There is a lot of assured capacity in place. Currently, 28nm is 30% of capacity. They are adding 28 FD-SOI too. 28nm is the last node before FinFET so is seen as a very, very long-lived node. 14/10nm and 10/7nm are each over 20% of production.

They have a lot of advanced packaging: FOPLP-PoP (fanout PLP) higher performance, thinner, and smaller. I-Cube (2.5D) for four HBM2. 3D SIP (ready in 2019) for homogeneous integration.

Samsung is also looking towards application solutions for HPC-AI to drive the 4th industrial revolution. This combines 7nm and below, ASIC service, Samsung DRAM and 3D NAND, 3D packaging. Because of their extensive memory portfolio, they can be a one-stop-shop. Similarly for connected applications (mobile, automotive, IoT).

They explicitly said that 10/8nm will be a long-lived node, as is 14/11nm (and going way back, 28nm). Note that last year they said that GAA would come at 4nm, but now they have pulled 3nm in and so GAA will come there, and they can extend the lifetime of 4nm with FinFET. 7nm is EUV (there is a 3:1 difference in price so economics mean that single-layer EUV is cheaper than triple patterning but more expensive than double patterning).

FD-SOI was not mentioned in the morning (and does not appear on the roadmap slide above), but was covered later in the afternoon.

The Details

All the above information came from the 40 minute press briefing that we got given before lunch. The foundry forum itself, with a huge audience, was in the afternoon. There was a lot more detail than in the press briefing, and they ran through each process node in turn. When I say "performance a%, power b%" that means you can take the better process to increase performance or to lower power, but not hit both numbers at the same time.

SD Kwon, SVP of the Logic PA Team (device architecture) said that the big innovations come at the big inflection points: 32nm HKMG, 14nm FinFET, 10nm Multi-ArFi, 7nm EUV, 3nm GAA. The way they develop a process is to evolve a mature baseline process with plug-in modules.

Pure pitch scaling (reduce CPP and MxP) on its own is not enough, so smart scaling is also required. At 14nm, they got 8% from that. This covers things like reducing the number of dummy gates, and adding smart diffusion break (SDB). Another is what they call "flexible contact placement" which I believe is what is usually called COAG, contact-over-active-gate. The pictures showed the contact moving from the central area of the cells to where the fins are, anyway. Next is adding mixed diffusion break (MDB) which "breaks the performance limitations of FinFETs".

Some smart scaling in the roadmap diagram above. From 14 to 11 there is Mx scaling. 8LPP to 8LPU has low power cells with single fins, and Mx scaling with EUV, and MDB. From 7LPE to 5 uses the flexible contact version 1, then down to 1 fine, then to 4LPE with next generation of contact, and Mx scaling. Then to gate-all-around.

11nm

This is the "one ring to rule them all" technology, with the best PPA in the class of 11-16nm processes. Performance 1.07X, area 0.82X, power 0.71X from 14LPP. There is an optional Vt for higher Fmax (+5%). 48nm pitch for m2.

10nm

This is the most advanced high-volume-manufacturing technology, inherited from 14nm.

8nm

The most optimized and economical node. This is the 3rd generation of the 10nm family, reusing all the 10LPP IPs (just scaled). 10% area reduction. 44nm m2 pitch.

8LPU is faster, and has lower power consumtion. With the Fmax boost from uLVT and MDB the performance is close to 7nm, 15% better performance. Power reduction is 0.44X for single fin devices versus 3 fin.

7nm

7nm uses EUV lithography for some layers. It is a very aggressive pitch-scale (presumably due to EUV) with a 46% area reduction (just like the old days), 17% speed up, 48% power reduction from 10nm.

5nm

5LPE is an easy migration from 7LPP. "Smart migration." Performance is 1.05X, power 0.82X, area 0.77x, versus 7nm. Key features are MDB, flexible contact placement (phase 1), and 1-fin device (for power reduction).

4nm

4LPE is further scaling, with extreme scaling of MOL and BEOL. There is flexible contact placement (phase 2), additional cell area shrinkg of 10% enableing M1 27nm/40nm.

3nm

3nm is gate-all-around (GAA). Performance 1.15X to 1.2X, power 0.5X compared to 7nm. Easy migration since, despite the very different transistor architecture, it is mostly a FinFET compatible process. The width of the nanosheets can be varied, like with planar transistors, so this escapes the quantization of FinFET (where you can have 1, 2 or 3 transistors etc but not vary their size). So this means that you can have variable drive (and probably means analog design is more feasible than with FinFET, I presume). The pictures all showed 3 nanosheets per transistor.

FD-SOI

Gitae Jeong, SVP of the LSI PA team talked about "mainstream tehnology". As he said, these technologies consume more wafers than the most advanced technologies (mostly 28nm I'm assuming). But the minastream roadmap then goes to FD-SOI instead of FinFET. The SOI process architecture is especially attractive for RF, and especially millimeter wave for 5G. Their process naming is "FDS" for the FD-SOI processes.

28FDS is on track for high volume manufacture with the first 5G mmwave RF product verified. There are 17 products planned in 2017, and more customers coming all the time. mmwave is up to 100GHz, fmax>400GHz.

28FDS+eMRAM will be available this year, with full manufacturable e-NVM solution. They have successful demos of test chips, 2 major companies engaged, and 10 more under discussion. Industrial temperature range from -40°C to 125°C. 94% yield (8Mb). Solder refloex feasibility achieved. So there are no technical barriers to mass production.

28LPP+eFlash is targeted at IoT (apparently Samsung is the only company with 29nm eFlash memory).

28LPP is a million wafer seller, also with RF and eFlash, ready for security, IoT etc.

28FDS+RF+eMRAM is also working, and is being extended to 18FDS+RF+eMRAM. 18FDS has the same BEOL as 14nm and is the migration path for MCU and IoT applications. PDK will be in September 2018 as planned. Performance is up 22%, power 37-55% down (versus 28). PDK 1.2 with eMRAM will be December 2019. 18FDS metal pitch is 64 using double patterning (28FDS was 90nm single patterned).

8” Specialty Technology

Samsung is expanding capacity over the next few years. They were at 190K wpm and currently are at 250K wpm, and will go up to 350K wpm. There are increasing numbers of customers with new applications (10 already in 2017).

CMOS Image Sensor (CIS) needs back trench isolation for the future, and a large microlens.

Power IC technology: they have shipped over 1M wafers with a unique device architecture. Automotive grade 1 is available. eFlash option is available with a very small flash macro.

They have two types of fingerprint sensors, one capacitive (built in 180nm) and one optical (built in 90nm), and this can be used to eliminate the home key since it can be on the screen.

Packaging

Dae-Woo Kim, VP package development team, laid out the packaging roadmap.

The biggest surprise to me was that they have PLP (panel-level package) technology. Instead of using a 12" wafer as the substrate and attaching die to that, it is done the size of big flat-panel displays using equipment that Samsung already uses for display technology. It has a fine pitch RDL and no chip bonding process. There are embedded passives devices in the RDL (such as capacitors and inductors). There also can be a backside RDL too, allowing memory to be stacked on the top. Panels, being rectangular, don't lose die slots near the edge like a circular wafer, plus there is the large panel size. Panel level package is in mass production as of 2Q 2018 (currently just samples).

The focus of the mobile package roadmap is to enhance i-PoP with 130um interposer in the middle of PoP. FOPLP-PoP is a high performance solution for premium mobie, that is thinner and smallergoing forward. It also can take thicker silicon (despite being thinner) and has better power dissipation than i-PoP.

AI/server/HPC package roadmap is all about big packages. This is 2.5D silicon interposer. There is a complex roadmap that seems to run out to 2023 or 2025, involving lots of HBM. 30um bump pitch with package size over 100mm x 100mm.

They have two different interposer processes, CoS and CoW (chip on substrate, and chip on wafer). They thin die to 100um for COS. But CoW handles the entire wafer without cutting, then attach the devices, mold it, attach to PCB substrate. CoS is

cheapest (by about 20%) but its size is limited to 1200mm^2. If you need four or more HBM then you have to use CoW. 2.5D Si-interposer. Mass production readiness is Q4 2018.

Samsung has advanced packaging that they haven't announced, called Chip Last Mult-Die Fanout Package, that has lower cost and larger size. It is not Si-interposer-based, they are developing a different technology. I didn't quite understand what was being disucssed here, but I think we were getting a sneak preview of something, so we weren't meant to understand it all.

Wafer bonding technology is mature. W2W and CoW bonding have been used for CIS since 2014: grab the sensor wafer, and attach it to the logic wafer. There is package development to go to the next stage and put CIS, logic, and DRAM in the same package, as in the Galaxy 9. They are working on Cu to Cu hybrid bonding for W2W and D2W, and when it is ready they will be able to support a bond pitch of less than 10um.

But his big news was that FOPLP mass production has started, and 2.5D Si-interpower mass production is ready.

Panel

The day wrapped up with a panel session on the SAFE ecosystem (that's what the "E" stands for). I won't try and recap what all the panelists said. By and large they were all emphasizing the way that Samsung Foundry had partnered with them, and how the whole ecosystem has come together. The panelists were:

- Jaehong Park, Samsung Foundry
- Gus Yeung, Arm
- Babu Mandava, Cadence
- Deirdre Hanford, Synopsys
- Joe Sawicki, Mentor

At the end, the panelists were asked what challenges they see coming in the future, especially with respect to GAA at 3nm. I'll leave a brief summary of what each person said:

Gus: We we need to do DTCO [design technology co-optimization] early. It can't be lip service. What we see from Samsung is commitment.

Deirdre: DTCO is a really critical partnership model. GAA will require new extraction, new timing models, gigantic designs. There will be impacts on placement and synthesis. This will be tough.

Joe: We end up feeding ourselves with the previous generation of chips. I can guarantee on 3nm side we will be using AI. And thank goodness we say goodbye to quantized gates again.

Babu: Hopefully we can make the cycle of change quicker, availability of tool flow, packaging. We can't take as long as we took from 16nm to 7nm. The money is in the middle of the pack, we all need that.

Jaehong: Achieving Moore's law every day is not easy. We need DTCO, DFM, EUV. There are many hurdles as everything gets smaller.

Chapter 5: Semiconductor in China

SEMICON China: Me and 70,000 of My Closest Friends

If you want to know why I made the trek from California to Shanghai to attend SEMICON China, it is well summed up in the press release that SEMI put out on opening day:

> *SEMICON China 2018 opens today at SNIEC in Shanghai with a record 70,000 visitors expected with 3,600 booths from more than 1,000 exhibitors bringing the latest innovations in the electronics manufacturing supply chain. SEMICON China kicks off as China's fab construction spending is forecast to grow from a record high $6.2 billion in 2017 to $6.8 billion in 2018, accounting for over 50 percent of worldwide construction spending. China is expected to significantly expand investments in wafer fabrication equipment this year to become the top-spending region in the world.*

Opening the Show

A procession of the people who have made the Chinese semiconductor industry and SEMICON China what it is today gave a perspective. They ranged from Lung Chu, the head of SEMI China, to Ajit Manoja, the CEO of SEMI globally, to Wang Ning, the head of CECC, and the head of the Shanghai chamber of commerce.

SEMICON China has become the biggest and most influential industry exhibition in the world. This year is the sixth consecutive year that SEMICON China has been the biggest SEMICON (it is three times the size of the San Francisco-based SEMICON West). It is also the 30th year of SEMICON China. To give a comparison, when SEMICON China started, there was 60,000 square feet of exhibition space (6000m^2). This year, there is 730,000 square feet. As it says above, there are over 3,500

booths from over 1000 enterprises (I don't quite know how that counting works, since that means lots of companies have more than one booth). Also running is FPD China (FPD stands for flat panel display).

The challenge, as Wang Ning pointed out, is that China consumes one-third of the world's semiconductor output but only 7% of that is domestically produced. The cost of semiconductor imports has now reached twice the cost of crude oil imports. It is the biggest user of foreign exchange reserves. Bottom line is that China needs to improve its global competitiveness.

[He didn't point it out, but I will. Much of those imported semiconductors are re-exported inside finished goods, which has to be the biggest contributor to foreign exchange reserves. You can't say that about much of the oil, although it is a feedstock for plastic.]

When Ajit Manoja, the head of SEMI, presented, he pointed out that in 2016 China started construction of 26 volume production fabs and by 2019 they will be ready to produce. By 2020, China will be #1 in global investment in the semiconductor equipment market (I'm actually surprised they are not already). However, China cannot be successful on its own since there are interdependencies everywhere. Ajit pointed out that we need to grow this industry beyond its current $400M. Of course in 2017, he said:

> *Some of the growth is due to memory prices. But we have always been relying on something to give us growth, and now there are ten or fifteen things. I'm optimistic we can have a CAGR of 6% taking semiconductor to $1T by 2030.*

The mantra of SEMICON China is "connect, collaborate, innovate". But, as Ajit said, we need to add "prosper and grow."

The head of the Chinese Semiconductor Industry Association came on. I'm not sure of his name since he wasn't on the agenda and I didn't seem to write it down. He was there to take away the punch bowl:

It takes two to three years to build a new fab, and another two to three years to ramp it to full volume. So to build a profitable plant takes a long time. I expressed my viewpoint last year and I'll say it again. For China, the semiconductor industry is rising, but has not quite risen yet. It will take time.

Huawei

The wrapup for the opening day was Chu Qing, the Chief Strategy Officer for Huawei. I'll try and give a flavor of his presentation, and pull out one or two points that I thought were interesting. He spoke in Chinese, very fast, and the simultaneous translators couldn't really keep up. Plus he leaped around from electronics, to biology, to philosophy, which made it harder still. Yes, he mentioned Plato and the shadows on the wall of the cave. He mentioned emperor penguins and how you need to get into the group to survive. He was also the only person whose slides were only in Chinese. Most other people had both Chinese and English on their slides, a few had only English. Paul Dempsey was sitting next to me, and he recorded the whole thing in Chinese. His wife is Chinese and he's going to get her to tell him what it all truly meant.

The title slide, which gives you an idea of how surreal his presentation was. It was titled 终结？还是元年 which, between Google and my own Chinese, means "The End? It's still the beginning". I'm not sure if there is any significance attached to the title being written in two font sizes (the small two characters mean "or").

He said he had three stories. The first story is about artificial intelligence (AI). He sits in a lot of meetings at customers on AI and they are all about bus bandwidth, and getting the IC power down, and stuff like that. But that's engineering, not philosophy. He prefers to look at the mythical future.

Then we had a diversion into René Descartes (that's him in the above picture), bionics, Plato's cave and the shadows, and how the human brain is constructed for intelligence and divided into different functional zones. He's not a big believer in big data, since that's basically just recording the past and trying to use it to predict the future. Like Moore's Law:

> *Many people regard Moore's Law as an objective rule, but it is just a commercial strategy.*

Next diversion was into Hilbert Spaces, which extends vector algebra into infinite dimensions. He wrapped up this section talking about what the translator (who was Australian, to make the whole thing more surreal) called "mechanism theory", although I suspect it is something else. Future innovations should be truly new thoughts and challenge the basic philosophy. AI is about more than just engineering and memory bandwidth.

The second story was about internet of things (IoT). There is a huge gap between IoT expectations and reality, especially in the networking. There is a network for the car, the home, all with different business models and technologies. It's a pseudo-concept. IoT should be the largest network ever. That's where the value is. The first generation was Bell and Ericsson who invented the phone. Now the high valuations are no longer in banking, they are in our industry. First AT&T, then IBM/Microsoft, more recently Google and Apple. So the big question is who will be the big guy in the IoT era. Maybe someone we don't suspect.

The third story was the IC industry. The life or death of the industry.

The above picture needs a little decoder ring. The fish at the front is Nokia. The next one (Q) is Qualcomm. Then Broadcom (although that bit isn't happenening, it seems). Who comes next?

We are in an era of M&A that will shape the whole industry. This is where the penguins come in. An emperor penguin can survive easily in the center of the flock. The ones at the outside not so much. Isolated they will die. So the IC industry has to huddle together to survive.

Will it consolidate completely into a single...no, not a penguin, a tree. The Chinese on the above picture means (at least according to me) "The last IC company?". The industry is going to sort of stop at 3/5nm and then it all becomes software. Anyone can build anything but almost no companies can afford the cost of a full die

development, which might measure in billions. But schoolkids will be able to program 3nm chips.

There has been lots of consolidation. The slide above shows some of the names that have disappeared (although if I was Hock Tan I'd take exception to Avago being on the list, since the only reason it disappeared is that it swallowed Broadcom and took on its name).

The big question for each semiconductor company is whether we'll be inside or outside the wall, in the middle of the penguins or isolated. Summer is approaching. In a clear dig at the recent blocking of the Qualcomm acquisition by CFIUS and the Trump administration, he ended with:

> *If you try and eliminate M&A, your local companies may just get eliminated faster.*

The Great Firewall of China

China is not a democracy, of course, so the elite who control the country cannot be voted out of power.Even more so after this week when term limits were removed. I'm not sure how China could be where it is today, almost overnight, as a democracy, since one-man-one-vote means that the rural poor would have

overwhelmed everything. We have a mild version of that in the US Senate, where we don't have one-man-one-vote, we have one-state-two-votes meaning our own rural inhabitants have an outsized influence on policy despite agriculture being a small part of the economy and an even smaller part of the employment picture. Ethanol subsidies, anyone? But that's a topic for another day (and another blog).

The one thing the powers that be fear in China is another revolution. They are in a pretty good shape so long as they continue to deliver prosperity, but there are still dissidents. Those in rural areas who do not have the economic prosperity of the big cities, and people with political grievances of one form or another. So in China, everything is monitored. To make that easier, a lot of the internet is blocked. This is known as the Great Firewall of China.

It is unclear if the original purpose was to create Chinese companies that are more easily controlled or just to keep places where free speech is too free inaccessible. A lot of Chinese policy requires exporters to transfer technology as part of the deal. Make China Great Again.

So whether it was driven by people being able to find pictures of that guy in front of the tank too easliy, or to make their own ecosystem, the result has been Chinese versions of US companies. Google is blocked in China, the big search engine is Baidu. YouTube is blocked. Facebook is blocked. People use WeChat for that (also known as Weixin). WhatsApp doesn't seem to be blocked, despite being owned by Facebook, but everyone in China uses WeChat for messaging already, so nobody cares. Blogs on blogger are blocked, maybe just because Google owns it, since blogs on Wordpress don't seem to be.

Some of the Chinese companies are bigger than their US equivalents, although there is reasonable doubt about any numbers coming from China, whether from the government or from big companies, which operate with the government's license, official or otherwise.

Mobile payments are clearly much more advanced than in the US. Even roadside vegetable sellers take electronic payments, and perhaps don't want to bother with cash that can be stolen. I noticed the same thing a couple of years ago when I was in Tanzania, one of the homes of M-pesa. Mobile payments in China are at least 10 times as big as in US. Governments like that, since cash can't be tracked. Harvard economist Ken Rogoff has proposed abolishing the $100 bill.

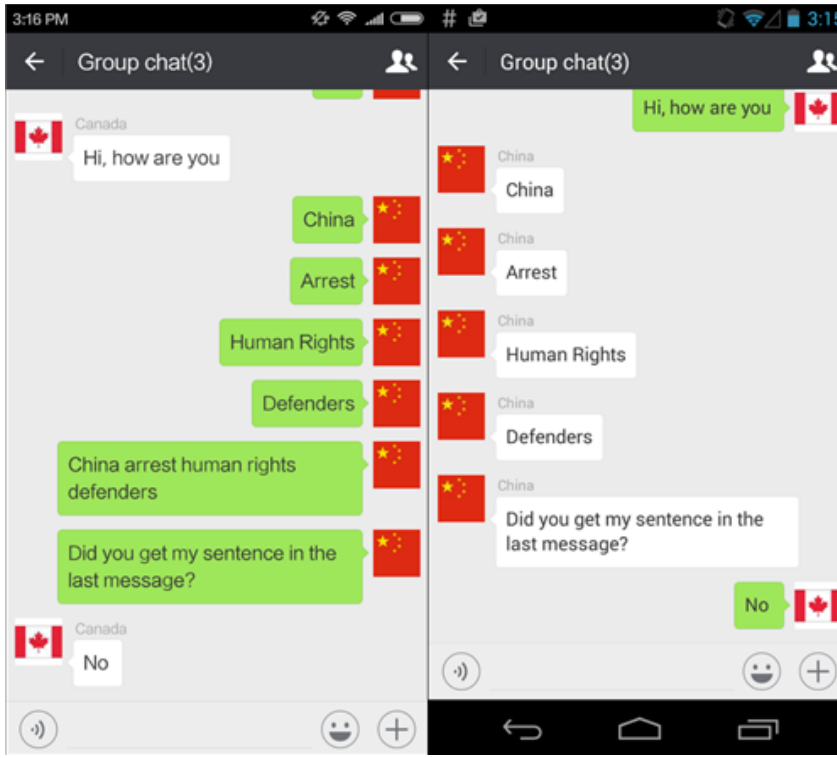
Alibaba is more than twice the size of Amazon, more than three times the size of eBay, more than the two of them combined. On Singles Day in November, it says it had sales of $25B. As the always amusing but dubiously reliable Zero Hedge says:

> *To put this figure in perspective, this year, "Singles Day" GMV came in at just a few billion more than the annual revenue of Sears/K-Mart (140,000 employees and 1,500 locations world-wide)..... again, I'll repeat that..... Alibaba sold, shipped and delivered the annual, global, sales volume of Sears/K-Mart in just one day! ....800 Million orders to deliver! Incredible! Bravo!.....all those guys on the tuk-tuks, scooters and bicycles must be exhausted...*

However, outside of China, these companies are nobodies. I see some parallels with the Japanese markets for mobile phones twenty years ago, They dominated in Japan, were very advanced, but gave up competing globally, and were eventually largely driven out of the rest of the world and, increasingly, even Japan. Japan didn't directly block competitors like the Great Firewall does, but it had its own Japanese standards that nobody else could be bothered with, that came to the same thing.

One reason that Chinese companies aren't getting anywhere outside of China is that inward focus, of course. But it's also because nobody trusts them not to be controlled in some way, maybe minor, maybe major, by the Chinese government. WhatsApp has end-to-end encryption, meaning that even WhatsApp/Fecebook can't see what you are saying (which US law enforcement hates, because...terrorism...child porn...drugs). WeChat is encrypted from phone to WeChat, and from WeChat

to phone. They claim "third parties are unable to view the content of your messages." But then you see a screenshot like this (from the Fish out of Water article):



Of course, even my photoshop skills are up to making a graphic like that, but I assume it's a genuine log.

So when it comes to China, the net is certainly not interpreting censorship as damage and routing around it. For now, China is more like AOL's walled garden in the early days of the internet, when a few geeks who were determined could find a wormhole out into the raw, dangerous internet. Today, that wormhole is the VPN (virtual private network).

VPNs

One way to tunnel through the great firewall is with a VPN (virtual private network). Cadence, like most large companies, uses a VPN for security. However, it also has the side effect of letting Cadence employees in China see Facebook and YouTube.

There have also been a lot of VPNs accessible to consumers on app stores. The great firewall reportedly sometimes slows these down since it can tell it's a VPN even though it can't read the content (like I can tell people are speaking Japanese without being able to understand what they are saying).

US Firewall?

But it's not like the US is blameless. It is clear that the government has been reading pretty much everything on the net, and you would be naive to believe any assurances that they've stopped beating their wife. Especially if the assurance comes from a politician who doesn't understand much about technology but is supposedly part of the oversight committee (*The Internet is a Series of Tubes* even has its own Wikipedia page). And doubly so if they are only allowed to perform their oversight in a locked room where they are not even allowed to take notes on what they read.

When the NSA gets caught intercepting and bugging Cisco routers in transit, why would anyone trust a Cisco router, or any American router?

> *Here's how it works: shipments of computer network devices (servers, routers, etc,) being delivered to our targets throughout the world are intercepted. Next, they are redirected to a secret location where Tailored Access Operations/Access Operations (AO-S326) employees, with the support of the Remote Operations Center (S321), enable the installation of beacon implants directly into our targets' electronic devices. These devices are then re-packaged and placed back into transit to the original destination. All of this happens with the support of Intelligence Community partners and the technical wizards in TAO.*

Breaking News: While I was on the plane to China, EFF showed the results of its FOI inquiry that if you get your computer fixed at Geek Squad (Best Buy) they may show data on it to the FBI (even though they would need a warrant to look if it wasn't "shown" to them). So far the only public case is child pornography, so it's hard to raise a lot of enthusiasm for the rights

of child pornographers. But remember Niemöller's poem, "1st they came for the socialists..." Think I'm being overdramatic? Apparently, Geek Squad has been working with the FBI for *ten years*, so who knows how deep the rathole goes already.

The New Walled Gardens

However, we seem to be starting to have new walled gardens in the rest of the world outside China.

Google used to be a search engine. If you wanted to know when your American Airlines flight would land, Google would point you to the right page on the AA website. Now, it just answers the question for you (and serves you some ads). Increasingly, it tries to answer the question it thinks you want answered. That's fine for you as a user, and in this case, you are just a cost if you go to the AA website and do anything other than buying a ticket.

Facebook and Twitter give you a feed of stuff you might be interested in, but mostly it keeps you on their site to see it. They have their own browsers built into their apps, for example.

For better or worse, we currently largely have an ad-supported web. I don't think it is supportable long-term for most sites. I went to some local newspaper yesterday, I forget which, but it wouldn't show me the article because I had an ad blocker turned on. But a model that works by charging third-parties to show me stuff I don't want to see (and I'm pretty good at not even looking at) doesn't seem supportable long term. However, arguably just that model worked for 50 years until ad blockers (aka DVRs) meant that only old people watched broadcast TV live (except sports). I gave up having a landline phone, not because of the cost, but because it got to the point that the only calls I ever got were people trying to sell me stuff, ads in effect. Anyone who knew me called my cellphone. My cellphone is almost at that point now though—anyone that knows me texts me so I don't usually answer my phone if it is a number I don't know.

It is all changing, of course. But if your business model is to be a news aggregator, that requires that there are at least some independently successful news sites to aggregate. There are only

so many pictures of what your friends had for dinner that you can stomach.

So now we've gone from the net interpreting censorship as damage and routing around it, to it interpreting it as an attempt to avoid seeing ads and blocking access to the content.

The Long Castle

One question people have about the Great Wall of China (the stone one) is how Mongolians couldn't just cross it. In Chinese, it is called chang cheng (长城), which means long castle (or long city, or some other things, too). The purpose was to be able to move an army fast to where it was required, not to keep people out due to its being a wall. Plus control customs and immigration (insert your Trump joke here, or point out how old walls are).

The Economist on Silicon Supremacy

A couple of weeks ago, the cover story of The Economist was *Chip Wars: China, America and silicon supremacy*. For the last few years it has been the biggest story in the semiconductor industry. You may already know the incredible fact that the value of semiconductors imported by China is greater than the value of all the oil imported by China.

PCAST covered a similar topic. Although their report was officially titled *Ensuring Long-Term US Leadership in*

Semiconductors, it was not the challenge of Japan or Europe that was the focus of the report. It was China.

Obviously, in semiconductors, it is a hot topic. In fact, it has been hot for a couple of years. At SEMICON West in San Francisco in 2017 I titled my overview blog post of the conference *SEMICON: China, China, China*.

DARPA's Electronic Resurgence Initiative meeting took place in San Francisco this July. There Bill Chappel gave the introduction to the ERI, which he runs. There I said:

> *The US government has a particular problem, with no access to leading-edge processes for semiconductor fabrication and state-of-the-art assembly, due to "our own regulations." I'm assuming this means things like fanout 3D packaging. However, I assumed that they had access to 14nm at GlobalFoundries' Fab8 in New York, but apparently that is too entangled with Samsung since it is their process.*

Of course, to make it worse, 14nm is no longer the true leading-edge process, that would be 7nm. But Global Foundries recently refocused their effort and decided not to pursue 7nm (and EUV lithography that goes with it). Intel has 7nm (they call it 10nm) although they have had widely-publicized problems getting it to yield.

Trade

It is difficult to understand trade in any context, let alone in a market as globalized as semiconductors. Let me start with what seems to be the economists' view (with a small "e", meaning most academic economists, although the magazine basically espouses the same position, as it has since its founding in 1843 to advance the repeal of the British corn laws, the big trade dispute of the day).

The economists' view starts with exchange. If you have a car to sell, and I want a car, then I can give you some money and drive off. You wanted the money more than the car. I wanted the car more than the money. So we both benefit from the exchange. If

not—because I won't pay your price, or I don't like your car—then the exchange doesn't happen. It doesn't really matter who the two parties are. Safeway wants $5 more than a gallon of milk, and I want the milk more than the money, again we both benefit. From a trade point of view, it really doesn't matter if the two parties are in the same city, the same state, or even the same country. I'm unlikely to buy a used car or a gallon of milk from China, but Société in France wants $10 more than it wants a pound of Roquefort, and I'm in the mood for blue cheese more than $10. Again, we both benefit from the exchange. It doesn't really change anything that there are other companies involved along the path of the exchange, and they are in France and I'm in California.

This applies to semiconductors too. Apple wants a wafer of A12 chips more than it wants a few thousand dollars. TSMC wants a few thousand dollars more than a wafer. Both benefit from the exchange.

So the economists' view is that all exchange like this benefits both parties and thus anything inhibiting it causes a loss of value. Tariffs, to the economists, are "throwing rocks in your own harbor."

The really big gains to exchange come from the fact that it enables specialization. Cadence doesn't run its own cafeteria, it gets Guckenheimer to do it for us. Guckenheimer doesn't need to make their own cellphones since we play a part it getting that done for them. This works fine across country boundaries too. Fabless semiconductor companies in the US don't need to make their own wafers since foundries in Asia do that for them. Correspondingly, foundries can specialize in manufacture and don't need to design their own chips, since fabless semiconductor companies do that "for" them.

Another good rule in economics is to take the perspective of the consumer. By this standard, the traditional economists' view is correct, and voluntary exchange is always a good thing for both parties. Indeed, in that model, there isn't even such a thing as a "country", they are just lots of individual producers and consumers.

When, above, I quoted Bill Chappel saying that the US government has no access to leading-edge processes "due to our own regulations" he wasn't really being critical of the regulations. For obvious reasons, the US is not going to buy chips from China and then put them in its latest jet fighters or ships. They have the ultimate in tariffs, they are infinite. There is no price, even if each chip came with a twenty-dollar bill, at which the DoD is going to buy their chips from China (nor vice-versa). It is probably true that this causes a loss of value, in the simple financial sense that the DoD could buy chips from overseas more cheaply, This limits the amount of specialization that can occur, which increases costs. Apple can have phones "designed in Cupertino and assembled in China" but the DoD cannot. They care about more than plain economic gains from exchange.

Scaling up to Countries

I thought about this quite a bit after I went to hear Clayton Christensen talk about his forthcoming new book. He pointed out that individual companies doing what is best for them doesn't automatically end up with doing what is best for the country. So every electronics company making a rational decision to buy chips from a handful of foundries outside the US, also leads to the decision for the US to abandon leading-edge semiconductor manufacturing. This happened without anyone making a decision that this would be a good idea, or even realizing it was happening until it was too late to do anything about it. You can argue, if you like, that this doesn't matter, since we can buy wafers more cheaply than we can make them, this is all good. Or you can, especially if you are the US DoD, argue that this is a big problem since they need leading-edge chips, and now they can't get them.

But, for better or worse, the result is that the US is largely out of advanced semiconductor manufacturing.

It is not just semiconductor, but a lot of other areas. Furniture is one I read about recently. The US used to have a big furniture industry but between 2001 and 2012, over 60,000 furniture factories closed in the face of Asian imports. The US steel industry declined since the big users of steel, such as the automotive industry, can buy it cheaper from China and India. A similar thing happened with rare earths, where from the 1960s to

1990s most of the world's rare earths were supplied by the Mountain Pass mine in California, but it became uneconomic and was closed due to environmental restrictions and competition from China. Now the market is almost entirely supplied by China. But rare earths are very important in magnets and so are important in cellphones, windpower generation, electric traction in cars, and more.

In all of these cases, the economists consider this a good deal for the US, since we can buy semiconductors, steel, furniture, and rare earths more cheaply than we can make them (or mine them). It is only when you look outside the simple economic arguments that there is any reason you might worry. For example, around 2010 China was rumored to have decided to restrict exports of rare earths to some countries, or to give priority to Chinese companies. It's not just a one-way thing, in 2015 the US decided not to let Intel sell high-end chips to the builders of Chinese supercomputers. Of course, in both these cases, the economists' argument is that these are mutually beneficial exchanges that are not taking place, and so an unambiguously bad thing.

That Economist Article

I'm perhaps making it sound like everything is bad for US manufacturing and great for China. But, as The Economist (capital "E") said in their leader about the semiconductor industry:

> *Firms from outside America and its allies, such as South Korea and Taiwan, dominate the most advanced areas of the industry. China, by contrast, remains reliant on the outside world for supplies of high-end chips.*

Actually, I don't know what The Economist means by "by contrast". It seems to me that both the US and China are mostly in a similar position, "reliant on the outside world" (meaning suppliers outside their country) for "supplies of high-end chips." Intel builds its own microprocessors of course. Micron builds some memories. In mobile, Apple mostly it buys from Taiwan. But Huawei, ViVo, Oppo, Lenovo, and the rest are the same. They either buy chips from the same foundries, or they use chipsets from Qualcomm or Mediatek that come from the same

foundries in Taiwan and Korea (Samsung does have a big fab in Austin TX that runs foundry, so maybe some come from there).

Later in the piece, The Economist indulges in another bit of wishful thinking:

> *Today America has the edge over China in designing and making high-end chips.*

I am sure the US probably has a greater number of high-end design groups, but there are islands of excellence in China, even if fewer in number. For example, Huawei/HiSilicon designs their own mobile applications processors. When the US stopped Intel selling high-end microprocessors to China, they designed their own chips. But China graduates a lot more engineers and computer scientists that the US, so that will probably change over time. It is perhaps technically true that America has the edge over China in manufacturing chips, since the US has Intel and China has nothing equivalent. Intel has fabs in other countries such as Ireland and Israel too. But it's a bit like deciding whether Malaysia or Thailand has the edge in automotive manufacturing. All the leaders in manufacturing are elsewhere—for leading-edge semiconductors, in Taiwan and Korea.

A third bit of The Economist leader that I'm critical of is in its conclusions. The US needs:

> *...to prepare for a world in which Chinese chips are more powerful and pervasive. That means, among other things, developing proper testing procedures to ensure the security of Chinese-made products. And tightening up on data-handling standards so that information is not being sprayed about so carelessly.*

In other words, to get all the advantages of exchange that go with the economists' (small "e" again) model requires proving that chips not only do what they are meant to, but don't do anything evil too. I don't believe anyone has a clue how to do this, certainly not starting from the chips themselves, as opposed to design data like the RTL. As for the last sentence on spraying data around carelessly, I don't even know what it means. If it just means we need better security, then nobody is going to argue.

Trust

All transactions depend somewhat on trust. If I buy that gallon of milk from the supermarket, I assume that they are selling me real milk, not just some white fluid. If you buy a semiconductor chip, you are assuming it does what you expect. Despite what The Economist suggested, you can't actually check it completely. If you are buying from some future hypothetical large leading-edge Chinese semiconductor company, then you have to trust them. Otherwise, buy from someone else.

The US DoD doesn't trust just anyone. They only trust you if you follow their rules, manufacture in the US, perhaps using only US citizens, perhaps with security clearances. Automotive companies won't buy semiconductors from just anyone, they audit the procedures at the manufacturing plants and qualify the reliability of the semiconductor process.

If you want to read something scary about trust, then you should take to heart the lesson of Turing Award winner Ken Thompson, who shocked the entire computer science community by showing that you can't even trust a program even if you have the source code.

If you use Cadence tools for designing your chips (and you should!) then you are trusting Cadence. Of course, in reality, we have a huge team of engineers doing what they can to make your design successful. But you are implicitly taking that fact on trust. An evil Cadence could be adding gates to your chip. An evil Intel could sell you chips that leak your information to the management processor. An evil AWS...you get the idea. You can check some things, but checking has a cost too.

You don't run lab tests on your milk before putting it in your coffee. It might seem like talking about milk is just a joke. But back in 2008, China had a scare about contaminated baby formula. As I said above, if you can't trust your supplier, get a new supplier, in this case Hong Kong (HK). People from mainland China stripped the shelves in HK to such an extent that HK mothers couldn't find formula at all, and so HK enacted a rule that people could only export a limited amount out of the territory. Of course, this meant there was beneficial exchange that

was not taking place. The economists would say that the correct solution was that the price of formula in HK should have risen to the market-clearing rate so that there was enough on the shelves, which would also have had the side effect of people from other places (US?, Australia?) stepping up production to supply the newly attractive high-priced market. But at some point, if HK can't even feed its own babies, the pure financial nature of transactions is not the only thing to consider.

The big question behind all these reports and articles is whether semiconductors are like baby formula, where we need to feed our own babies whatever happens geopolitically, or whether it is like furniture, where all Americans can benefit from cheap furniture (paradoxically often sold by a Swedish company) at the cost of a relatively small number of non-strategic jobs.

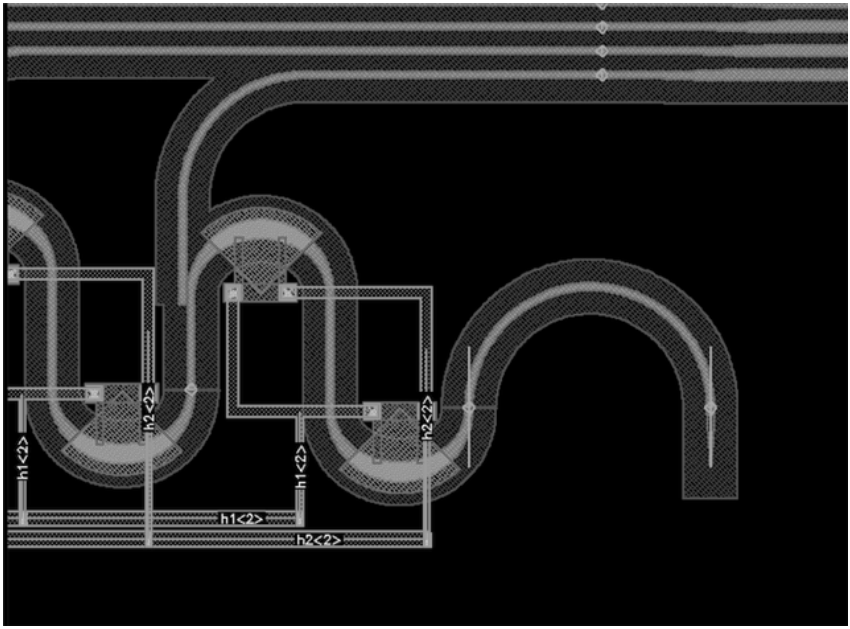
Chapter 6: Silicon Photonics

Yoga Is Passé, the Future Is CurvyCore

Despite CurvyCore sounding like something that you might take classes in at your local gym, it is actually new technology that allows computing and representing non-Manhattan shapes in Virtuoso.

The CurvyCore technology is targeted at a wide range of applications such as microfluidics, MEMS, and conformal routing. But the initial application driving the technology is photonics. Photonics involves many strange shapes, far from Manhattan geometry—let's say Lombard Street geometry, after the crookedest street in the world (actually, Lombard Street isn't even the crookedest street in San Francisco, that would be Vermont Street on the back of Potrero Hill, but that's not a tourist area).

I sat down with Gilles Lamant, the Cadence Distinguished Engineer who is leading the CurvyCore development.



The picture above gives you an idea of just how "non-Manhattan" we are talking about. This is a very different direction in terms of layout from where advanced processes have been heading. With technologies like self-aligned double patterning (SADP), advanced processes are even more regular than just Manhattan layout, sometimes with only one dimension allowed on a layer, sometimes with only certain widths and spacings permitted. However, having both of these in Virtuoso means that both the ultra-regular FinFET geometry and the curvy geometries are supported in the same tool at the same time, as you can see at the bottom of the picture above where the rectilinear electrical connections are mixed with the curvy optical ones.

Unlike regular IC layout, you can't really draw layout like this by hand with adequate accuracy. Instead, at the core, is a mathematical model so that we can compute paths, offsets, ribbons, boundaries, and do mathematical operations equivalent to Boolean operations. The table above shows the data model, which consists of three main layers. The pink layer at the top consists of the actual polygons of the layout, OpenAccess shapes. Since OpenAccess doesn't support curves, there can be a large number of polygons to represent one of these layouts with adequate precision. The middle blue layer is an IEEE double floating point model, with discretized shapes. The bottom green layer is a purely mathematical model, interfacing through symbolic equations. In effect, the green at the bottom is an accurate mathematical representation of the shapes, the top pink layer is "polygon level layout", and the intermediate layer serves as a sort of translation layer between mathematics and polygons, preserving as much precision as possible without going all the way to arbitrary precision arithmetic.

Photonics

For the rest of this section, I'm going to focus on photonics.

For the last few years, photonics has largely been driven by datacenter and HPC requirements. Since datacenter networking is mostly optical, they have moved heavily into photonics. But that has opened the door to other applications, too, such as Lidar, biomedical, and military. 5G is starting to drive RF directly over fiber (without first digitizing it). Some people see photonics as

the replacement for the current SerDes approach, which is starting to require too much energy. From a business point of view, foundries have had to bring up photonics capability to satisfy the HPC market, but having developed the technology these other markets are opportunities.

The reason that CurvyCore plays nicely with photonics is that light is not like electric current. You've probably heard that light likes to travel in a straight line, and that is true. But light really doesn't like to go around sharp corners since they reflect and cause signal losses—the same effects happen with electrical signals once you get up to RF frequencies. To make light go around a corner you have to do it gradually around a curve, hence CurvyCore. Light is unlike electrical signals, even RF, since it has different modes (different colors aka frequencies, different polarizations, electric and magnetic field components).

An attractive application area is in planes and satellites since optics is radiation-robust. In satellites, there are particles coming from the sun, but photonics is well-behaved under those conditions. And it is way lighter. It is unclear where its role will be in automotive since that is extremely cost sensitive (the other end of the scale from satellites), although anything that reduces weight is attractive.

Another attractive area is the microfluidic medical market. This is a mixture of photonics (for optical sensing) and microfluidics, involving channels for moving almost infinitesimal amounts of samples and reagents, and taking measurements. They are very complex, involving a lot of computation so requiring advanced processes on the same die. It has the potential to be a big market since these devices are use-once-and-throw-away. Like an inkjet printer cartridge, each one contains a chip, but if you are monitoring your blood in a home lab, you will get through many more units than you do printer cartridges.

Diwali, the Hindu Festival of Lights...and Photonics, the Silicon Festival of Light

November 7 is the big day in Diwali, the Hindu Festival of Lights, as it symbolizes the victory of light over darkness, and of knowledge over ignorance. All of which made it a perfect day to have Cadence's Silicon Photonics Summit, and celebrate light over electrons, and knowledge of photonics over ignorance of the subject.

I'm going to pick two presentations from the day to highlight:

- Vladimir Stojanović of Berkeley Wireless Research Center: *More than Moore with Electronic-Photonic Integration*
- Rick Stevens of Lockheed Martin: *RF Analog Photonic Applications*

Vladimir Stojanović

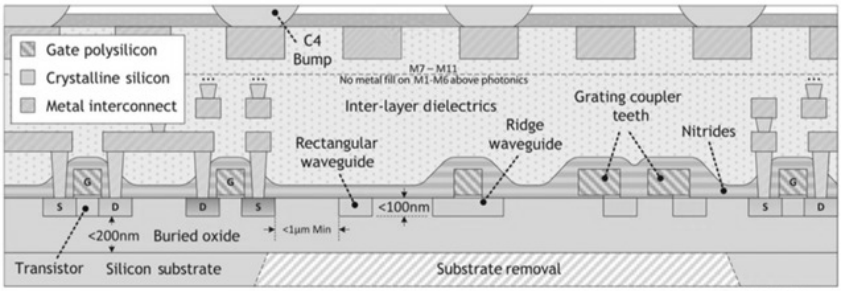
The day's keynote was delivered by Vladimir of the Berkeley Wireless Research Center at UC Berkeley. He went into a lot more technical detail than is appropriate here, but to me, the key message was that they came up with three different approaches to integrating CMOS and photonics, and then built fairly significant test vehicles to understand the advantages and disadvantages.

Back in 2010 or so, he and his team had wondered whether they could get a photonics device in a standard CMOS process with no changes. Even if the device wasn't great, the upside of doing this would be incredible, and they could continuously improve it. They built the first device in 2012 on a 45nm SOI process, and since then in 32nm. This is what they call "zero change".

Next, they considered a More than Moore approach, with any standard CMOS wafer with a photonics wafer (fabricated by CNSE, the Center for Nanoscale Science and Engineering) flipped over and bonded onto the top.

The third approach was "deposited photonics" using bulk CMOS (as opposed to SOI). They manufactured this successfully with Micron at 180nm and with CNSE at 65nm.

They decided to build some realistically complex designs with each approach to see the tradeoffs.



Bulk CMOS peaked at 65nm/40nm and then performance decreased at more advanced nodes due to the higher gate resistance. 32nm/45nm had both the fastest transistors and thick enough silicon bodies to guide the light. Lower dimensions don't have enough thickness for the photonics. They picked the IBM/GLOBALFOUNDRIES 12SOI (45nm) CMOS. This allowed them to build zero-change optics in 45nm, with no process modifications thus getting the closest proximity of electronics and photonics. There is a single substrate removal post-processing step. This gives a monolithic photonics platform with the fastest transistors.

As a demonstrator, they build a chip with zero-change 45nm SOI with photonics integrated into the chip. The chip had a memory band, and a RISC-V processor, and photonics transceivers. The chip could be configured either in "memory mode" or "processor mode" and then a pair of them could run together. It was the world's first processor to communicate with light. The processor/memory interface was completely optical.

Autonomous driving has made lidar important. Most lidar is pulsed: a laser pulse is sent out, and the time difference is measured to when it returns. But Vladimir thinks that it is potentially more attractive to build FMCW (frequency modulated continuous wave). It is less sensitive to shot noise and also less sensitive to background noise. Instead of putting out a chirp, the

frequency is ramped in a sawtooth wave, so that the beat frequency can be measured between the outgoing "teeth" and the return.

They built the two chips in IBM 65nm for the CMOS, and CNSE for the photonics. This allowed them to build a complete monolithic lidar system including a beam steering optical phase array and an FMCW receiver. The photonics chip was flipped and bonded to the CMOS chip, as in the above diagram.

Finally, the most recent development is deposited polysilicon photonics platform, deposited onto deep trench oxide. It is the only way to integrate photonics with advanced nodes, workable with any of planar bulk CMOS, FinFET, and ultra-thin-body SOI CMOS.

To wrap up, he made three points:

Silicon photonics is an enabler of new capabilities, and a good way to think about it is as analogous to a new on-chip inductor or a new on-chip transmission-line.

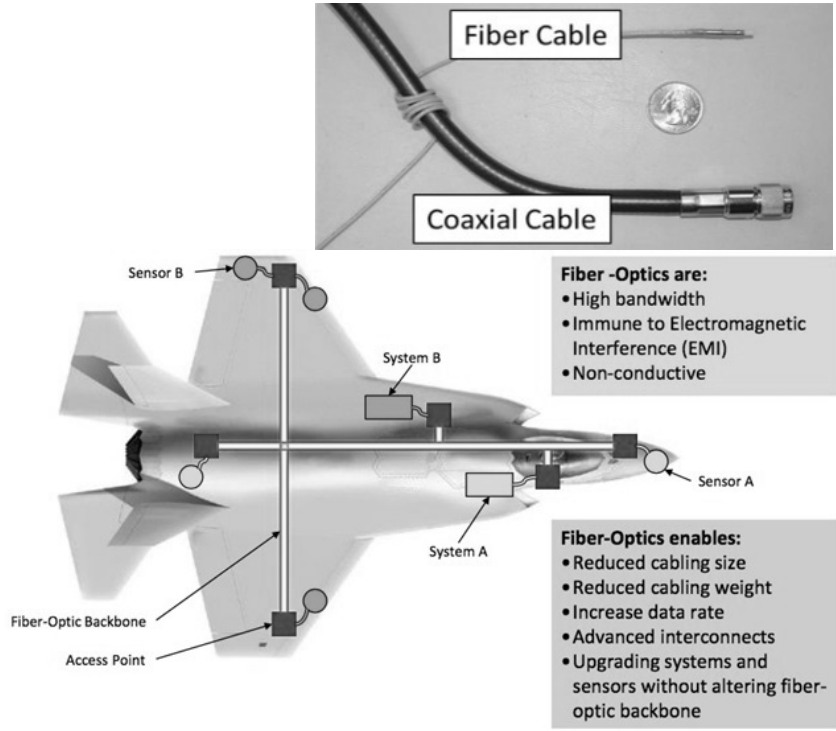
It has the potential to revolutionize many applications despite the slowdown in CMOS scaling.

Deposited polysilicon-photonics is the key to monolithic integration with advanced transistors.

Rick Stevens

Rick turned out to be the second presentation of the day. He works for Lockheed Martin, so is focused on defense applications. From Rick's point of view, the importance is that:

> *Photonics enables capabilities that otherwise could not be achieved within the same size, weight, and power.*



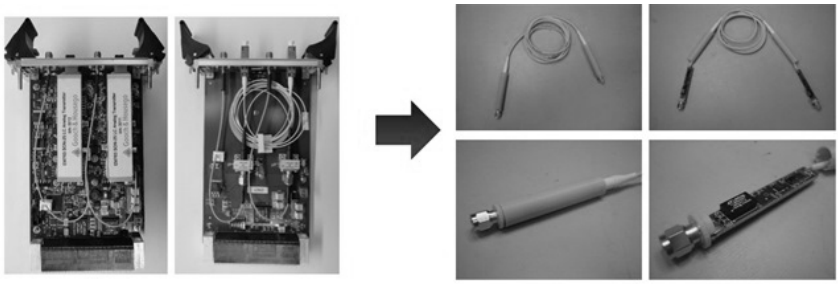
A military aircraft (a "platform" in military speak) has a central core processor and a photonics backbone. If you don't centralize the main processing, then you end up with a lot of duplication of processing resources. The picture above shows the possibilities. Using fiber like this has the advantage of less loss per distance, wider bandwidth, EMI immunity, and non-conductive (lightning strikes, etc). Today, coax is used. It has limited distance due to losses, limited bandwidth due to losses, and needs equalizers to adapt to frequency-dependent losses.

Another practical problem with coax is that it is easily damaged if it is bent on too tight of a radius. Replacing it (which also can't bend it much) is a 10-day exercise that involves taking all the "stealth" surfaces off the platform, replacing the cable, replacing everything, then putting it in an anechoic chamber to make sure it is still stealthy. Rick loves the photo to the right that shows the difference more impressively than any number of words. The

black coax is the old way of doing things, with cables that cannot be bent too much. The yellow fiber can be coiled so tightly that it can be wrapped around the old solution. The fiber is also higher bandwidth. The quarter gives you the scale.

> *This is what gets people so excited about putting this on the platform.*

Yet another advantage is that fiber is fiber. There is no need to re-cable the platform to upgrade all the electronics. New electronics can run over the "old" fibers without having to strip down the platform. In the military, platforms can last a long time, with the electronics going through several generations. The most extreme example I know is the B-52 bomber, operating since the 1950s, with the last delivery in the early '60s, and is expected to last until the 2050s. Since its maiden flight was in 1952 (I think that is a coincidence, and not where the name came from) that would make it 100 years old.



The really big gains come with an integrated solution. On the left is a discrete solution (still with fiber, not coax though). On the right is the integrated solution. Rick had brought the device in the picture and passed it around for us all to handle. Okay, sorry, you had to be there. Some of these devices are located on the leading edge of the aircraft and there is not a lot of room out there. For stealth reasons, it also needs to be thermally isolated and so that also requires pumping coolant out there. The integrated version contains a thermo-electric cooler (or TEC) that he'd still like to get rid off for obvious power/reliability reasons.

Obviously, these solutions need to work in harsh environments and extreme conditions (war is messy) and that needs to be tested

as they move toward deployment. The proof of concept is this year. But:

> *It won't be on a platform any sooner than two years, since we have to convince the platforms to take the risk. But a couple of years is pretty quick for a military application.*

Summary

I think these two presentations make a nice matched pair. Vladimir showed a lot of technical detail about how to integrate CMOS and photonics on the same chip. Rick gave a dramatic demonstration of what the upside from doing this could be, at least in the context of stealth aircraft. You can use your imagination to take all this and extend to, say, automotive applications.

Chapter 7: 2019 Predictions

Let’s finish with my predictions for 2019. These were made at the beginning of the year, so depending on when you read this some might have come true or false already.

These are the topics that I expect I will spend a lot of time writing about this year. Let's start with the big picture, and work down to the small (do we call it a 30Å process yet?). You’ll notice many of the topics are the same as the chapters in this book. It takes several years to get a process into production, and one or two to design a major SoC, so the themes don’t change all that fast.

Memory Pricing

In 2018, the overall semiconductor market was very strong, but much of that was more demand than capacity in the memory market, especially DRAM. With additional capacity coming online, people who follow the memory market full-time all seem to predict softening of prices. This could mean that the overall headline numbers for the semiconductor market might shrink.

From an EDA point of view, it won't matter if the memory market shrinks, since memory companies only do a few designs and then manufacture them in massive volume. I'm always reminded of VLSI's corporate counsel at one point, who had come from Micron, saying in one meeting "At Micron we used to put a couple of designs into production per year, and in the ASIC market we put a couple into production most days." EDA, IP, and generally anything associated with design, is much more driven by the other end markets.

China

With a couple of dozen fabs under construction, this should be the year that some of that capacity starts to come online. But China is the big geopolitical story in many other ways, a big one being that they are committing $150B to semiconductor in a drive to be more self-sufficient. As Chinese fabs come online, even if they are not globally competitive (at least at first) there will be a lot of pressure for Chinese manufacturers to use the chips. Since about half the semiconductors in the world are imported into

China, this could have a big effect. It will affect memory first, since China has no competitive leading-edge fabs, but it is worth remembering that a huge percentage of designs are done in non-leading-edge processes. In fact, the #2 foundry GLOBALFOUNDRIES, decided this year not to pursue the leading edge anymore, and China is not pursuing the leading edge anyway (at least for the time being, that could change eventually).

5G

This is the year that 5G rollouts will begin. But 2019 will mostly be the year of marketing hype, with pilot projects and expensive handsets that you can't really use anywhere. I expect that this year's Mobile World Congress in Barcelona in March will be an all-out 5G show, with little else being talked about.

5G is not like previous standards in that there are multiple frequency bands and there is much more of a tradeoff between coverage and performance. I fully expect lots of deliberately planted confusion where the performance of the so-called mmWave band offering speeds of up to 10Gbps gets blurred with the performance of the mid-band and low-band spectrum, which is more like 100+Mbps. The reality is that mmWave won't go through walls, so you won't get that performance from the big basestations, only from so-called small cells.

EDA in the Cloud

Last year the first announcements of EDA in the cloud were made. At ESD Alliance meetings during the year, EDA in the cloud was the big thing that dominated the discussions. So far, it has been a technology using the scalability of the cloud to bring more compute power to bear on the problems. The business models haven't changed, but it wouldn't surprise me to see some tentative changes during the year.

Deep Learning, Artificial Intelligence, Neural Networks

I use these terms interchangeably. The advances in AI have been truly astounding over the last five or so years. I think that there

are two different aspects in which deep learning will affect the semiconductor industry.

First, there are all sorts of applications of the technology for end-user applications. I think that training will continue to be largely done in the cloud using GPUs, but that increasingly inference will be done on the edge devices since that is where, in aggregate, most of the compute power is.

Second, deep learning will be increasingly used in EDA tools, both under-the-hood in driving the algorithms inside tools, and also as part of the flow, automating a lot of the iteration that goes on, especially during phases like design closure, synthesis, and physical design. The buzzword here is "no human in the loop." That will remain a dream in 2019, but is certainly the general direction.

Automotive

Automotive will continue to be a fast-growing segment of the semiconductor industry, driven by the need for foundries to find markets that are growing as mobile slows or perhaps (in terms of dollars, not transistors) shrinks. It is also driven by the end-markets switching away from human-driven internal combustion engine vehicles owned by individuals to...whatever vision of the world turns out to be true. This is another area where China is leading since, for pollution reasons, they are basically making it increasingly hard to purchase anything other than electric vehicles.

Processes: EUV, 5nm, 3nm

In 2019, 7nm (and Intel's 10nm which is similar) will be mainstream. EUV will be in true volume production. The bleeding edge of process development will move to 5nm. Just because EUV works at 7nm doesn't mean that there are not major issues for 5nm. As was said at SEMICON West last year, "we're gonna need more photons." Next-generation interconnect might well be based on ruthenium (Ru). At 3nm, it looks like transistors will be nanosheet gate-all-around (GAA).

2019

So that's my predictions for 2019. Look for a new book next year and we can see how the year panned out.