Thursday, September 15, Justin Rattner Presentation

Tigerfish®
Transcribing·Editing

203 Columbus Avenue · San Francisco 94133
toll-free 877-TIGERFISH

www.tigerfish.com

Thursday, September 15, Justin Rattner Presentation

[Start of Recorded Material]

[Music]

Male Voice: Today's presentations contain forward-looking statements. All statements made that are not historical facts are subject to a number of risks and uncertainties, and actual results may differ materially. Please refer to our most recent earnings release and our most recent Form 10Q or 10K filing available on our website for more information on the risk factors that could cause actual results to differ.

[Music]

Female Voice: Ladies and gentlemen, please take your seats. Our program is about to begin. As a courtesy to our presenters as well as those around you, please take a moment to silence all electronic devices, especially cell phones. Thank you for your cooperation. Please take your seats.

[Music]

Female Voice: Ladies and gentlemen, please welcome Johan Jervøe.

Johan Jervøe: I'm a little quiet this morning. I've heard that we had ninjas on stage yesterday, and I feel that that's like swimming with sharks. So, okay, it's too early for jokes. I get that. Or maybe it's just my Danish humor, so I apologize for that. But what a great day we had at IDF yesterday. Mooly laid out the roadmap for Ultrabooks, describing their compelling value proposition, starting today, all the way through its

continued transformation, both through Ivy Bridge and Haswell, with all their additional capabilities. I really like the Windows 8 demo. Very smooth. The drag-and-drop feature that lets me personalize the screen both with touch obviously as well as with keyboard and mouse. And, as will.i.am would say, that's fresh.

And now, today, third day, it's actually one of my favorite moments of all our keynotes is when Justin Rattner comes onstage. He doesn't really need any introduction. Personally, this is the favorite part of IDF. Justin shows us what's just around the corner, what the future may hold, and sort of what's behind the curtain. And on that note, what's behind the curtain, actually, this is a picture from this morning of what is behind the curtain. And I have a little favor to ask. You know, it takes quite a lot of work, a lot of people, and a lot of logistics to put IDF together -- 81 semi trucks, 11 hotels booked with over 7,792 rooms, 5,000 plus attendees. So let's jointly say thank you to the IDF team. Enjoy the rest of the day. Enjoy Justin. And have a nice trip back. Bye.

[Video]

Female Voice: Ladies and gentlemen, please welcome Justin Rattner.

Justin Rattner: Would you like to see my Mooly impersonation?

Male Voice: Yeah.

Justin Rattner: Really?

Male Voice: Yeah.

Justin Rattner: Okay. Have to get ready for this. You all set? That's it. Come on. You don't know how many hours I practiced that. Okay. I'm going to lose the chapeau here. I don't know. We'll toss it to the camera crew. Thanks, everybody. Anyway, thank you. Welcome to day three. Most of all, thanks for sticking around. I know for some of you that's not easy to spend the entire three days or for some people even four days at IDF, so we're just delighted that you're here. And as Johan said, it's always -- it's our tradition on day three to talk about the future and what we see coming. It could be in the next few years. It could be as much as the next 10 years.

And, in fact, today, we'll actually look at developments both here and now and ones that will take the better part of the decade for us to achieve. This day actually has a certain significance, I think, to me, at least, and I think to the industry as well. It was five years ago today that I stood on this stage to give the opening keynote. I should note that's probably the first and last time I will give the opening keynote, unless something really untoward happens at Intel. It's usually the CEO thing, and I'm not vying for the CEO position, so have no fear.

But it was my privilege five years ago today to introduce the Intel core microarchitecture and to explain the reasons for Intel's move to multi-core architecture. And some of you -- maybe more than a few of you were here for that keynote and remember this slide. I think this was sort of the pivotal slide in the presentation, where we talked about

slightly slowing down a couple of cores to get a higher overall level of importance. So here we are five years later, and I thought this would be a particularly good time to assess the progress and the impact of multi-core and, more recently, many-core architecture. So that's our mission today, to take a look at where we've been and where we are and where we will be with multi-core and many-core technologies. We've come a very long way in those five years. I don't think any of us could've imagined in 2006 that we'd literally be talking about processors with certainly tens and maybe even hundreds of cores. It seems like cores just took off. Two cores begat four cores and four cores begat eight cores, and now we're already just beginning the age of many-core processors with many, many more cores to come.

Some of these cores are different from one another, right? We're talking about heterogeneous multi-core systems. Some of them use specialized processors, like graphics processors, GPUs, to do certain parts of the workload. But soon Intel will introduce its first general purpose many-core product, and that will literally have many tens of cores. This will be called Knights Corner, it's part of the Knights architecture family, and it implements our many integrated core architecture, what we call the MIC architecture. We're already out in the field with our many-core software development vehicle, that's Knights Ferry. I'm sure some of you actually have Knights Ferrys in your lab and are getting the opportunity to take the architecture for a bit of a spin.

So far, people are reporting great results, and, most importantly, for us, an easy transition from multi-core to many-core software. And that's --

fundamental to the Knights family and the MIC architecture is to deliver a very consistent programming model. If you're familiar with programming, let's say, Xeon cores, it's a very easy step to programming the MIC cores. It's the familiar memory model, the familiar instruction set, albeit it enhanced with some extraordinary amount of floating-point power. I'll actually show you an example of that in just a moment. We've got a couple of Knights Ferry systems behind me, and we'll talk about that in some detail.

We also launched, the middle of the last decade, our Tera-Scale research program, and that's been a very active effort, I'm sure, as a number of you know. And it's really there to help us chart the course to higher and higher levels of parallelism. In Intel Labs, we've actually built a few experimental many-core processors. You may remember the 80-core processor, our teraFLOPS processor, we showed, oh, I think in about 2007, and more recently we talked about our 48-core single chip cloud computer, which was intended to test out some ideas for what a future high core count server processor might look like targeted at the large-scale IP datacenters.

We'd even built a global research community around the SCC, and that community's called MARC, and there are literally hundreds and hundreds of researchers that are part of that community, and they're all working with the SCC. And we've been busy creating better tools and better methods for programming many-core architecture, and, in fact, we'll show you probably the newest one of those today, in just a little bit. What we're seeing in the lab, and this is research that we've done in our parallel research laboratory within Intel Labs. We've looked,

really, at a very large number of problems that cross entirely different domains of computing.

Most importantly, we haven't limited applications of the MIC architecture to just technical applications, just HPC applications. You'll see all sorts of things on this list, and this list is by no means exhaustive. The reason I chose this graph to show you this morning is to show you the kind of scalability we're seeing from many cores. This has always been one of the big dreads that, you know, we'll program up a few apps, and they'll get a speedup of, you know, three or four or maybe six or seven, and then the lines will flatten out, and adding more cores won't increase performance at all. In fact, in some cases, it may decrease performance. You can actually get speed down if you throw too many cores at a problem.

But across this very large range of applications we're seeing excellent scalability. In fact, I don't think there's anything on this chart that's seeing less than 30X scalability, and you can see a number of those lines are close to the linear speedup line as well. So this has been very encouraging and has really given us a lot of confidence that people are going to be able to put this architecture to work and really take performance to newer and higher levels. One of the most challenging and profound applications of this kind of technology is the search for subatomic particles. And, to look at that, I've invited Andrezj Nowak from the CERN open lab to talk to us about his work in particle physics and in particular with the MIC architecture. Morning, Andrezj!

Andrezj Nowak: Morning Justin. Thanks for having us. It's a pleasure.

Justin Rattner: Great to have you here. Great to have you here. I know you came a long way, and we really, really appreciate that. Why don't you start out by telling the audience a little bit about openlab, because I'm sure a primarily U.S. audience may not have heard of it.

Andrezj Nowak: So we look in a collaboration with industrial partners -- CERN works in a collaboration with industrial partners. It's called CERN openlab. And one of those prime partners is Intel, and we have been working together since 2001 and advancing computing solutions for the Large Hadron Collider. So it's an advanced collaboration in research of advanced computing.

Justin Rattner: And what is your work, in particular?

Andrezj Nowak: So I personally focus mostly on parallelism and optimizing not only for multi-core and many-core, but also in core efficiency, so we do a lot of work on low-level performance. And optimizing performance for Large Hadron Collider software is a really difficult ordeal because it's so complex. It has millions of lines of code and it's not so easy just to port it to many-core, but we have achieved that.

Justin Rattner: Well, I've had the privilege of visiting the Large Hadron Collider. I mean, it's truly impressive. Is it possible to share a little bit about that overwhelming experience?

Andrezj Nowak: So it's a huge machine that produces a really large amount of data, and we only select a small amount of it and we keep it. And all the rest is

thrown away because we already know it. It's physics that we already know. It operates at a temperature that is cooler than outer space, at 1.9 Kelvin, and I think the best way to introduce it is to show you a video of what we do at CERN.

Justin Rattner: Let's take a look.

[Video]

Justin Rattner: Well, I have to tell you, as good as that as good as that video is, it doesn't compare to actually being there. The physical scale is just overwhelming.

Andrezj Nowak: Exactly. So the machine is really huge. You could say that this is the largest machine ever built by man, a really huge [ring] underground. It's superconducting. The detectors that we use to capture the data are probably half of the size of this room or maybe a little bit bigger.

Justin Rattner: Yeah.

Andrezj Nowak: So in order to process all that data coming in, and in order to sift through it, to select what is really interesting to us, we have underground farms. But the true challenge comes with processing it later. The real value of the data comes with what comes later. To that extent, for that, we have built the Large Hadron Collider Computing Grid. It runs on 250,000 Intel Architecture cores.

Justin Rattner: Whoa, whoa, whoa, whoa. Say that again.

Andrezj Nowak: 250,000 Intel Architecture cores on the Large Hadron Collider Computing Grid.

Justin Rattner: So not only is it the biggest machine in the world; I think it's the biggest computer in world.

Andrezj Nowak: This is a distributed structure, so it's spread across hundreds of data centers across the world, which work in unison in order to be able to process the whole physics complexity that comes with the data.

Justin Rattner: Okay. So talk to us a little bit about the process.

Andrezj Nowak: The processing takes place in four major domains. And out of those four major domains -- CERN open lab is working actively on all of them. And on the video on the screen, you see an example of a simulation framework where we simulate the physics that take place in the detectors, and then we compare it against the real response of the detectors to understand where the differences are and where could we make a discovery.

So one of these workloads is done for us in Relation Framework, which we have managed to parallelize. It has over two million lines of code. And we have worked with Northeastern University to parallelize and to optimize it. And we have achieved almost a 40X speed-up on a 40-core Xeon.

Justin Rattner: Wow. So almost perfect speed-up.

Andrezj Nowak: And we really like linear speed-ups.

Justin Rattner: Don't we all, don't we all. [laughs] So I also understand that you've had some early experience with the MIC architecture. Can you tell us about that?

Andrezj Nowak: Yes, that is correct. We were very excited to find out that Intel is working on the MIC, because it not only brings a large amount of cores and many threads but also wide vectors. That means that we can put very advanced and dense workloads on that architecture and develop them very easily.

Now, the reason why we were so excited is because Intel has decided to provide the same toolset for the MIC as it provides for the Xeon. It means that we could directly apply our experience from the Xeon to the MIC, and that way developing for MIC becomes a breeze. It's really easy and we could port our applications in a matter of days instead of expected months, as you could expect with another architecture.

Justin Rattner: Amazing. Can you actually show us an example of some high-energy physics code?

Andrezj Nowak: Yes. Let's take a look at the high-energy physics code that Intel helped us visualize.

Justin Rattner: All right.

Andrezj Nowak: On this machine, we have a high-energy physics code called the track fitter. So we basically take signals that come in from the detector, and we try to fit the tracks of particles into that response of the detector, and to see where it goes, and what is the energy that we deposit in the detector. So we have a single-core MIC system here that runs only on a single core of a MIC. So if we press "Run," we will see the framework.

Justin Rattner: I think he means to say he's using one core out of all of the cores on the processor.

Andrezj Nowak: Yeah, of course.

Justin Rattner: We don't have a single core --

Andrezj Nowak: This is a 32-core MIC, but we only are using one core, so it's a sequential application, right? This is what we have if we don't have any speed-up, if we don't try to parallelize our application. So you see that the tracks on the screen are appearing very slowly. It's not really a fast application that can give us a quick response and really the performance that we are looking for.

Justin Rattner: Well, the power of the MIC comes obviously from using all of its cores. Can you show us the full array and what it can do?

Andrezj Nowak: Exactly. On this machine, we have engaged all 32 cores of the MIC to process that application. And let's just take a look at that. And you see

that the workload, the whole workload is processed in a matter of mere seconds, as opposed to minutes as it would take on a single core. So we achieve a very good speed-up, which allows us to get much more out of the hardware, and we are not leaving any performance on a table. And we really do not like to leave the performance on a table.

Justin Rattner: For sure. Not with the kind of workload you've got. So this has shown good scalability. What are your thoughts about scalability in general?

Andrezj Nowak: So even before Intel started helping us with this particular example, we have implemented it on our own without talking too much to you guys. And on this screen, you see the scalability graph for that application. So it's a heavily vectorized and heavily threaded workload which scales almost perfectly with all the cores in the MIC. So it means that not only our expertise on Xeon has paid off and we were able to get a really good speed-up on the MIC, but it also means that Intel has delivered a machine that really works very well with very dense code that might not be easy to run in an efficient way.

Justin Rattner: So where do you plan to go from here?

Andrezj Nowak: So we think that it is very important for us to focus on many-core scalability. And it is very important for us to make sure that we move forward during this period of time. And that means that we look very much forward to the continued development of the MIC technology. We hope to have more cores. And we will take any amount of cores you can throw at us. And we definitely look forward to good scaling on the Xeon.

Justin Rattner: Hey, if you can use them, we can build them. Thanks, Andrezj.

Andrezj Nowak: Thanks. It's been a pleasure.

Justin Rattner: It's a pleasure to have you here. [applause] Hey, I have one request, one request. I know this impacts everyone here. In fact, it probably impacts everyone on the planet. Please don't make any little black holes that are going to suck all of us into extinction. Can you promise me that?

Andrezj Nowak: If you can promise me more cores, I can promise you that.

Justin Rattner: All right, it's a deal.

Andrezj Nowak: Thanks a lot.

Justin Rattner: Thanks, Andrezj. [applause] I couldn't sleep last night. I was worried about those little black holes forming in the Large Hadron Collider. [laughs] Well, there's been obviously a lot of angst about multi-core and many-core programming.

I really hope that you're beginning to see, with the kind of experience that Andrezj was sharing with you this morning, that those fears, those concerns, I think, are beginning to subside as more and more people take this kind of programming challenge to heart and engage their minds and their energies at programming, because the potential here is so great. And I'm talking about real programmers that are

programming multi-core and many-core machines every day -- not somebody locked away in some ivory tower trying to figure it out, but people who are solving real problems.

The way the media has portrayed it -- and these are just a few of the quotes that we picked out over the last five years, since the introduction of the Core Duo processor -- and the way it's been portrayed, you'd think you'd have to be some kind of freak to actually program these machines. Or, as we like to say in the labs, you'd have to be a ninja programmer. [gong rings]

Don't worry. I'm not going to bring some guy dressed in a black outfit onstage again. We've already done that once in this IDF. In fact, I think they sent that outfit back to the cleaners. So no ninjas today. But if you look at the people doing multi-core and many-core programming, they look just like you and me. And they don't carry throwing stars of various forms, or knives, or whatever. They are just the kinds of programmers that tackle challenging problems every day. So our goal at Intel is really to banish ninja programmers forever. Now I hope a ninja doesn't come running out right now. [laughs]

But what I want to do is show you a set of four examples -- that's four, not that -- four examples of real live multi-core and many-core applications. They're all running live here onstage. No magic, no tricks. And nothing is getting piped in from a national lab someplace. Some are running on PCs, and some are running on servers. And both are equally appropriate for multi-core and many-core applications. And you'll meet the developers and you'll hear their stories.

I'm going to start with Web applications, just like the ones we use every day when we launch our browsers and we start to move around the Internet. To support Web applications, it takes many, many databases, if you want to think about it that way -- content repositories of various sorts -- in order to build those dynamic Web applications. And if that complexity wasn't enough to scare you, you have to do that over millions of concurrent users.

So the legacy databases are really in trouble here, because they were never designed for these kinds of massive workloads, the kind of traffic volumes they see, and the complexity and rapidity of the queries. As a result, there's been a substantial amount of innovation that's gone into making those legacy systems perform as if they were built yesterday, and a lot of that work involves multi-core architecture. So, come on with me, come back here. Billy's here. Billy, you brought your data center, I see.

Billy Cox: I did. Good morning, Justin. We've got it rolling on wheels right here.

Justin Rattner: I know. I don't think Google has anything to worry about here.

Billy Cox: Not at all, no.

Justin Rattner: So, how do people go about solving a problem that I just described?

Billy Cox: So, after a lot of experimentation, what they finally ended up doing is moving the database into memory, just move the whole thing into RAM.

Justin Rattner: So, an in-memory database kind of solution?

Billy Cox: Yeah, and actually what we're showing here is a use of the open-source project [Memcache] where they move the entire database into RAM. Now, RAM is fast. It's 100 times the bandwidth that we got off of disk, and one one-thousandth the latency. So, the performance boost is huge.

Justin Rattner: Okay, so if you do that, how does multi-core get in the mix? I mean, just putting it all in memory isn't going to give you the horsepower you need.

Billy Cox: Well, it helps certainly, but we take advantage of multi-core as we're showing here, and what we can do now is, the best reported results to date have been 560,000 transactions per second. But with multi-core, we can go over 800,000 transactions per second, and at about one fourth the latency, a huge boost in performance.

Justin Rattner: With just this much computing power?

Billy Cox: Just 48 cores.

Justin Rattner: 48 cores in that little rack. Wow, that's, that's really, really impressive. What's the next step here? Where do we go from here?

Billy Cox: Well, we continue to optimize the code and work with the open-source community to make it go even faster.

Justin Rattner: Okay, so a million queries per second, just around the corner?

Billy Cox: It's on the horizon.

Justin Rattner: All right, thanks Billy.

Billy Cox: You bet.

Justin Rattner: Sure, give it up for Billy Cox. Excellent. Well, we've seen the benefits that multi-core brings to web applications on the cloud side of the connection. What about web apps that have seen all that growth that I talked about but actually are running on the client end of the connection? And I think a lot of you know that the development of those web applications was really empowered and enabled by the development of scripting languages such as JavaScript. So, to help us answer this question about what can multi-core and many-core do on the client end of the connection, I've asked Brendan Eich, CTO of Mozilla and the inventor of JavaScript, to join us this morning. Hey Brendan.

Brendan Eich: Hey.

Justin Rattner: How are things at Mozilla?

Brendan Eich: Very good, happy to be here.

Justin Rattner: Super. Is this your first IDF?

Brendan Eich: Yes, it is.

Justin Rattner: Okay, well they're mostly harmless. I don't think you have anything to fear. As the creator of JavaScript while you were at I guess Netscape at the time, you're probably the best person to give us a little bit of the history and the evolution of the language.

Brendan Eich: Sure, I had 10 days in May 1995 to put it together. I was inspired by Self and Scheme, so it has good parents -- first class functions from Scheme and [prototypes] from Self. I was told to make it look like Java. And it was used at first for form validation and image rollovers and annoyances, but it's grown up to be the full language of the web.

Justin Rattner: Wow, and it's really incredibly popular. Can you give us some statistics?

Brendan Eich: Sure, I mean it's been growing in popularity thanks to libraries for it like jQuery. It's the most popular programming language on GitHub, the social hacking site. And a lot of that is due to [node JS], which is a server-side use of JavaScript. So, it's grown outside the browser, too.

Justin Rattner: Wow. So, we're here today. We're talking about multi-core and many-core computing. Can you say how multi-core has impacted JavaScript?

Brendan Eich: JavaScript so far in the client is predominantly sequential, but we really would like to utilize the full capabilities of multi-core hardware.

Justin Rattner: Okay, well come over here, we may have a solution just at hand. Here's [Tatiana]. Come on up, Tatiana. All right, so we've just been talking about JavaScript, and I know you've been working on some of the technology that might address this problem of no multi-core for JavaScript, so tell us about it.

Tatiana Shpeisman: Sure, this is the amount of [River Trail] out of parallel extensions to JavaScript. What you are seeing on the screen is a physics simulation of n bodies. So, what is here are specks of dust floating through the space and interacting with each other.

Justin Rattner: Okay, how interesting does this sound? Okay, so obviously it's not doing very well just running sequentially -- three frames per second, not too impressive. What else can you show us?

Tatiana Shpeisman: Okay, well, this wasn't River Trail, this was standard JavaScript. Now what you are seeing is River Trail. This is parallel, this is 10 times sped up, 45 frames per second.

Justin Rattner: So, this is running now across all the cores at easily 10X speed up.

Brendan Eich: Wow, 15X. I mean, this is amazing. We can do image processing, audio/video, computer vision, voice response in JavaScript.

Justin Rattner: So, you know, talk about how difficult it would be for a JavaScript programmer to learn how to use this parallel extension.

Tatiana Shpeisman: Actually, this should be quite easy, because this is not a new programming language, this is just JavaScript. We simply gently extended it to add parallelism in a very simple way. And because this is JavaScript, it also preserves the security and safety properties of JavaScript. We don't add any security concerns with this, and we're also as portable as JavaScript.

Justin Rattner: So, what do you think, Brendan?

Brendan Eich: I'd like to try it myself. How do you plan to make it available to the community?

Tatiana Shpeisman: Well, actually, it's already available. It is on GitHub.com. The project name is River Trail, and we'd like web developers to come and start using it and help us to make us better.

Brendan Eich: That's wonderful, and do you have any plans for standardization?

Tatiana Shpeisman: Absolutely, we'd like to see this as part of standard JavaScript, we'd like to take it to [ECMAscript] standard.

Brendan Eich: I work with the ECMA JavaScript Standards Body, I'm going to promote it there.

Tatiana Shpeisman: Thank you.

Justin Rattner: Well, I think you guys should get together backstage. All right, thank you Tatiana, and thank you Brendan for joining us for this first public demonstration of the parallel extensions for JavaScript. Thanks very much.

By the way, and I just heard earlier this morning, the competition's already downloaded it. So, those of you who are interested, you know, get to GitHub and check it out. I think you'll find it a pretty interesting tool. So, we continue to move out the boundaries of what's possible with multi-core and many-core programming.

But, let me move quickly to the third of our examples, and that's the notion of building wireless base stations out of standard parts, in fact out of PCs. We all know how quickly the infrastructure is growing. If you've seen any of these base stations -- and we've brought one here today. As far as I know, it's not operation, but if it was, you'd probably have incredible signal strength if you had an LTE phone in your pocket. Most of this is built out -- or, all of this is built out of highly custom components: A6, digital signal processing, almost pure hardware architecture. And we had the idea, along with our friends at China Mobile, that it might just be possible to build a complete LTE base station on Intel architecture. And we'd love to be able to tell the base station community that there's an app for that, an app that runs on standard PC hardware and turns a standard PC into a base station, there's an app for that.

To prove that we could do this, we actually entered into an agreement with China Mobile two years ago. It's part of a larger project called the Cloud Radio Access Network that's sort of organized under the auspices of China Mobile. The architecture's interesting, and time doesn't really permit me to do it justice, but the key idea is that at the cell tower, there's just the RF front end. The radio signals are digitized and then moved over fiber optic networks to a data center, and that's where the actual base stations are located. So, it's kind of base stations in the cloud. Let me show you. Hey Sunny.

Sunny Zhang: Hi, Justin.

Justin Rattner: How you doing, good to see you. Okay, doesn't look like this. Are you telling me this is an LTE base station?

Sunny Zhang: Yeah.

Justin Rattner: Wow, that's really amazing. Kind of give us a walkthrough, tell us how it works.

Sunny Zhang: Okay, so this is a time [unintelligible] station, so actually it's a Sandy Bridge, [unintelligible] Sandy Bridge desktop.

Justin Rattner: It's got a core i7 in it.

Sunny Zhang: Yeah, we implemented TDRT base station processing in software.

Justin Rattner: Super. So how do we apply multi-core to the base station problem?

Sunny Zhang: Yeah. That's the most challenging part. As you know, [wireless] processing is a very high workload for any type of process.

Justin Rattner: Oh, absolutely.

Sunny Zhang: Yeah. So we optimize [abogen], and we vitally use the SCC and [aviax] [vectoring instruction]. And we also redesign the [abogen] so suitable for Intel architecture.

Justin Rattner: Super.

Sunny Zhang: Yeah.

Justin Rattner: So, in some sense, you're using the vector engines in the Core i7 like digital signal processors. Is that right?

Sunny Zhang: Yes. So you can see the -- here's the benchmark, it shows here. It has the computing cycle of each [wireless mojo], so with all this optimization, the result was we achieved very good performance and can meet real-time and throughput IRT requirement.

Justin Rattner: So is this maxing out the Core i7?

Sunny Zhang: Oh, not yet. So you can see here, so this is a [quad core] processor. We've divided workload across multi-cores, and there are still significant remaining computing cycles. And you can use it to do other non-real-time tasks.

Justin Rattner: Wow. Boy, we put an Ivy Bridge in here, you could probably run the whole network on one PC. No, just kidding. I didn't mean that seriously, but it obviously shows how much computational capability there is in a box like this versus a box like this. So tell me about how you've dealt with the real-time issues. I mean, this is communications. You know, this isn't just, you know, putting up a UI or something like that. You have to meet deadlines. How did you go about doing that?

Sunny Zhang: Yeah, so this is -- software is running in real-time mode on Linux with a real-time patch, and with multi-core and the multithreading technique, the software have enough flexibility to meet real-time requirements. And you can see the round trip here is about 3 millisecond, it's able to meet real-time requirement, and you can see it here. So this is a real-time video.

Justin Rattner: Oh, this is streaming live.

Sunny Zhang: Streaming video from the base station to the plant over the emulated RT link.

Justin Rattner: Wow. That's great. So what's the next step here? Where do we go now?

Sunny Zhang: Well, we completed live trial this year, and next year we will go to field trial with China Mobile and other ecosystem partners.

Justin Rattner: Wow. That sounds fantastic. Good luck with the actual field trials.

Sunny Zhang: Thank you very much, Justin.

Justin Rattner: Thanks, Sunny. Appreciate you being here. As I said, there's an app for that. And I might add that, you know, we're not just doing this experiment with wireless base stations. We're looking at building network routing equipment and switches basically out of standard high-volume commodity parts for PCs and for servers. The final example of the diversity of multi-core and many-core programming that I'd like to show you this morning is using the technology to improve security on the personal computer. You heard Paul talk about some innovations in that area on Tuesday.

We want to go a little bit further or quite a bit further this morning and show you how we can bring the power of multi-core and many-core to desktop security. So I think Dave's over here. Dave's waiting for me. I'll catch up, Dave. Jeez. I can never go fast enough. Welcome, welcome, welcome.

David Durham: Thank you, Justin.

Justin Rattner: Okay. So this is not a very interesting screen, Dave. Why is that?

David Durham: Well, first of all, let me tell you, it's wonderful working in the labs, doing security research because security's such an interesting topic today, and we've had so much impact with, you know, the AES crypto and the processors that's already out there. You saw some of the most

recent stuff in Paul's demo, and now, I'm going to show you what's next for the exciting world of computer security.

Justin Rattner: Okay, let's have at it.

David Durham: What you can see here is with the rise of social media, an unprecedented amount of personal content is just getting dumped on the web and getting distributed everywhere, copied everywhere, and so forth. What we want to show is how we can protect people's confidential, personal information, their pictures, their videos, whatever, their objects, online, in the cloud, wherever it goes.

Justin Rattner: Okay.

David Durham: And, you know, I love this demo because you can actually see the cryptography in this one, right? All these are pictures on my website.

Justin Rattner: Oh, so these are all encrypted images, is that it? That's why they sort of have that scramble of pixels.

David Durham: Yeah. And if that wasn't enough we added facial recognition as a biometric authentication for all of these individual pictures.

Justin Rattner: Okay, okay.

David Durham: So if I step over here to my website, it says, oh, it recognizes me, this is my website, all these pictures decrypt, and you can actually see this

happen in real time, you know, for all my photographs. And, you know, why don't you give it a try, too, Justin?

Justin Rattner: Okay. You're sure it just didn't turn them all on? All right, I'll try it. Let's see what happens. Okay, here I am. Whoa, wow, oh, and I can see not all of the pictures decrypted. So how are you doing this? I mean, this is a really computationally intense problem.

David Durham: Well, every single image here is wrapped in its own policy, with its own soft biometrics, which is why we can program them to recognize some pictures go for you -- wherever you put these pictures on the web, copy them on a USB stick, put them back, move them around, it'll always recognize based on the policy that's associated with it. It's just an egregious amount of cryptography that's going on.

Justin Rattner: So how are you using the multi-core and the many-core features to do this?

David Durham: Oh, we're not just using the cores. We're using the entire processing capability of the die. You can see on the graph over here that it's not just the CPU cores we're using in this demonstration, but processor graphics and its compute resources, as well. So you see those spikes every time it recognizes one of our faces and we step in front of it, you know, you get one of these spikes up because there's all this cryptography that's going on [all simultaneously].

Justin Rattner: Oh, there it is. It's just coming on the frame.

David Durham: And, you know, this is using the entire platform, as it is, and that's why it's so fast.

Justin Rattner: Okay, so tell us exactly how it's done.

David Durham: Well, if we move to the slide, I can sort of show you the stack that we're actually running on this thing.

Justin Rattner: I'm trying to get it to move, Dave, but it's not. Oh, there we go.

David Durham: So, basically, we've got the CPU cores running the camera application, they're doing the actual recognition when we get a face, the AES new crypto instructions are doing all the actual, you know, individual data decrypt in line with the display, and we have our social media applications and all the programming that you normally do on the IA cores. And then the workloads that we can parallelize very well, we move those over the processor graphics. So this continuous facial detection isn't eating up your CPU resources. We're just running it over there on the side. You don't even know it's going on --

Justin Rattner: And we still have enough graphics capability to put up the screen images.

David Durham: Absolutely.

Justin Rattner: So all of that's running through those pipelines at the same time.

David Durham: Yep.

Justin Rattner: Wow, that is really amazing. True heterogeneous multi-core, many-core computing, all in one box, all in one application.

David Durham: And keeping your data safe.

Justin Rattner: And keeping the data safe. Thank you, Dave. Excellent. Give it up. Okay. So four, I think you'll agree, extremely diverse applications, all bringing the power of multi-core and many-core technology to bear and even, as we just saw, with the personal security application, harnessing the power of both the CPU cores and the GPU cores on the same die, in Sandy Bridge and of course Ivy Bridge, and Haswell, beyond that. So I hope, at this point, there's no question in your mind that the time is now, if you haven't already, to start building multi-core and many-core applications, and you don't need to be a ninja programmer to do it.

You know, Dave's office is just down from mine, and I've never seen him in a black outfit at any time during the day or night. At this point, I think it's fair to ask the question, well, what lies ahead, what lies beyond multi-core and many-core computing? And our answer is something you may not have heard of. It's called extreme-scale computing. This is multi-core and many-core computing at the extremes. Our 10-year goal here is to achieve nothing modest, a 300X improvement in energy efficiency for computing at the extremes.

That means the equivalent, if you're keeping score at home, of 20 picojoules per floating point operation per second -- 20 picojoules,

right? Trust me, we're orders of magnitude away from that goal today. The example of computing at the extremes that I'll share with you today is this one -- the extreme of exascale computing, computing at the exaFLOPS level. The challenge here is generating that much computing power within a modest 20-megawatt power budget. And if you think that doesn't seem too hard, let me tell you, today, a petaFLOPS computer is burning somewhere somewhere between five and seven megawatts. So if we just scaled it up by 1000x, it would be in the gigawatt range, and I'd have to buy everybody a nuclear reactor to run that machine, maybe a couple of nuclear reactors.

Well, to help us get a better understanding of extreme-scale computing, I'm going to bring out Shekhar Borkar now. He's an Intel fellow, and he is the principal investigator on our DARPA-funded Ubiquitous High-Performance Computing project. Come on out, Shekhar.

Shekhar Borkar: Hi, Justin. Good to see you.

Justin Rattner: Wow, do you look [snazzy] today.

Shekhar Borkar: Thank you.

Justin Rattner: You never look this good at work. And you've got your Intel Labs shirt on. Oh my goodness. I should have worn mine today. You just didn't tell me. If you had called me, I would've packed it. All right. Well, I've just introduced this notion of extreme-scale computing. Take a minute

and give us a real kind of hard-nosed example of what extreme-scale computing will mean as we move towards the end of the decade.

Shekhar Borkar: Sure. So let's take an example. Let's take an example of 100-gigaFLOP system today. If you want that performance, it will require about 200 watts of power.

Justin Rattner: Mm-hmm.

Shekhar Borkar: With the extreme-scale technology, we would like the same level of compute performance requiring two watts or less.

Justin Rattner: Wow.

Shekhar Borkar: And think about in a handheld, 100 gigaFLOPS in your hand.

Justin Rattner: So that's two orders of magnitude.

Shekhar Borkar: Yes. It's only possible if you look at the entire stack and innovate across all aspects of the system.

Justin Rattner: I can believe that. Well, can we see an example of the kind of technology we're creating to deliver those kinds of gains?

Shekhar Borkar: Sure. So let me give an example of one technology, what we call near-threshold voltage logic. Let's start with the transistor. Today, we operate a transistor at several times its threshold voltage.

Justin Rattner: The threshold voltage is the voltage at which the transistor turns on.

Shekhar Borkar: That's correct. The transistor turns on at the threshold, and we operate at several times the threshold, because we like performance. One thing we can do is reduce the supply voltage and bring it closer and closer to the threshold of the transistor.

Justin Rattner: I see.

Shekhar Borkar: The theory tells us that it will increase the energy efficiency by about 8X or so. I can show you an example if you are interested.

Justin Rattner: Okay. Let's go take a look. Sriram's back. Where's your lab coat?

Sriram Vangal: [laughs] Not today.

Justin Rattner: That really looked good. I think we should have a rule -- all Intel Labs people have to come onstage in white coats.

Sriram Vangal: Certainly, yeah.

Justin Rattner: Okay. All right. Sriram, we can say a lot more now. I think Paul wanted to just preview it. So tell us what we're looking at here.

Sriram Vangal: Certainly, yeah. We're going to take a closer look at the experimental microprocessor that Paul showed two days ago. What we have here, and we're disclosing for the first time, is the code name for this, called Claremont. So Claremont is actually a Pentium-class experimental

microprocessor that's capable of running very close to the threshold voltage of the transistors, just like how Shekhar was explaining. You saw it on day one running Windows. And here it is running Linux. And it's capable of booting multiple operating systems. And here again, you see the same funny cat animation.

Justin Rattner: You're still torturing that cat.

Sriram Vangal: Well, we're trying not to, but yes. I mean, you know? So this is basically the video of the Linux boot.

Justin Rattner: Okay. Do you want to tell us more about the electrical performance we're looking at?

Sriram Vangal: Certainly. I mean, yeah. Basically, we're operating within a couple of hundred millivolts of the threshold voltage of the transistors. And here's the thing. We get about a 5X improvement in energy efficiency, like Shekhar was saying, when operating at near the threshold. And the power consumption is just a few milliwatts of power, allowing it to be powered off here again from the small solar cell, powering the entire processor core.

Justin Rattner: You know, people are already calling this the "postage stamp processor" in the press.

Shekhar Borkar: One thing I want to clarify. You want me wondering why only 5X, because I said 8X earlier.

Justin Rattner: I know, but you're prone to exaggeration. [laughter]

Shekhar Borkar: Well, it's not exaggeration. There is a reason behind it. The reason is we use an old code here. If we had started this design from scratch, we could've gotten 8 to 10X improvement. But this is a proof of concept.

Justin Rattner: I see. Okay. Yeah. So this is actually a Pentium core. And I understand it's so old --

Shekhar Borkar: It's so old.

Justin Rattner: It makes me feel old when you say that. It's so old that you had a hard time finding Pentium motherboards.

Sriram Vangal: That's right. Absolutely.

Justin Rattner: And you actually went out on eBay looking for them?

Shekhar Borkar: Yes, we did. I mean, we were literally dumpster-diving, trying to find motherboards that would work. I mean, yeah.

Justin Rattner: So here we are at the frontiers of computing, and we're looking in the dumpsters for motherboards. Fantastic. [laughter] Okay. Well, so we've seen how this is working very close to threshold voltage, delivering that kind of actually dramatic increase, 500 percent increase in energy efficiency, but obviously running very slowly. I mean, the cat's not moving too quickly this morning. What do we do when we want that performance? How do we turn this into a speed-demon?

Sriram Vangal: For that, let's walk over to the second setup. Basically, one of the capabilities of this technology is its ability to ramp up in frequency, ramp up at higher voltages to deliver the performance that the applications might be asking for. Here we have a second setup. In this case, we're running Windows. And in this case, the processor's actually running 10 times faster than the case in the first setup where it was running at near threshold. Here, you may have increased energy consumption, but you do get the performance on demand, emphasizing wide dynamic range capability of the microprocessor.

Justin Rattner: So, in fact, using that dynamic range -- I see you're running an old version of Quake here.

Sriram Vangal: Quake. Exactly, yeah. Here's a Quake demo. And you can see that it could really benefit from the 10X improvement in performance that it demands, and --

Shekhar Borkar: So it's very important -- one other thing I want to add is in our research, we look for a wide dynamic range, so you can get performance as well as energy efficiency. You can tune it in dynamically.

Justin Rattner: Okay. Well, we're going to talk more about that in just a second. I think we should thank Sriram for an excellent demo.

Sriram Vangal: Certainly. Thank you.

Justin Rattner: Thanks for being here with us.

Sriram Vangal: Thanks a lot. Yeah. Thank you, Shekhar.

Shekhar Borkar: Thank you, Sriram. [applause]

Justin Rattner: So I don't get the question later, the reason why we didn't just slew the voltage infrequency is you can't do that on one of these old motherboards, and we didn't want to take the time to design a new one. So Shekhar, okay, we've seen really, I think, an outstanding example of the work towards increasing the energy efficiency on the processing side. I think you and I both know that memory consumes a huge amount of power really in any class of system. What are we doing for memory?

Shekhar Borkar: So for memory, I want to show you another example over here.

Justin Rattner: Okay.

Shekhar Borkar: Here's a prototype that we built, what we call HMC, or Hybrid Memory Cube, in collaboration with Micron. What this is is a DRAM prototype with a stack memory concept, so your memory and logic stack together, with an advanced --

Justin Rattner: The memory chips, I think, are sitting under these heat sinks.

Shekhar Borkar: Exactly. And with an advanced memory interface, highly efficient energy [crosstalk.]

Justin Rattner: And how high a stack of DRAMs?

Shekhar Borkar: I think it's about four to eight that can be stacked.

Justin Rattner: Okay. And I think these are four high, if I recall.

Shekhar Borkar: These are four high, yes.

Justin Rattner: Right, okay.

Shekhar Borkar: But the technology can go up [from there]. And with the stack memory, it's a very high energy efficiency. And this is showing right now [the order of] a terabit per second transfer with energy efficiency [7X].

Justin Rattner: So this is 120, 122 gigabytes per second.

Shekhar Borkar: Gigabytes per second.

Justin Rattner: That's nearly a terabit per second.

Shekhar Borkar: At 7X energy efficiency.

Justin Rattner: So it's not only the fastest DRAM ever demonstrated --

Shekhar Borkar: The most energy --

Justin Rattner: -- it's probably the most energy-efficient DRAM. Wow. It sounds like we're getting a good jump on extreme scale. Come on down here for a minute and talk to me about the rest of the extreme-scale computing program. I think it's pretty interesting.

Shekhar Borkar: Yes. Today, what we did was we looked at just a couple of examples. But the research agenda for our extreme-scale computing is really quite broad. The only way we believe that you can get there by the end of the decade, with this kind of a challenge, is to optimize up and down the stack and across both hardware and software.

Justin Rattner: So it's really a co-design problem.

Shekhar Borkar: It's a co-design concept.

Justin Rattner: Right.

Shekhar Borkar: It's the only way we'll get there.

Justin Rattner: And are we going to need new ideas and operating systems?

Shekhar Borkar: We need new ideas and operating systems, the execution model, the programming systems, the hardware, the [all-core] design concepts.

Justin Rattner: So really across the board.

Shekhar Borkar: Across the board.

Justin Rattner: Okay. Well, it sounds like an incredibly challenging problem. I want to wish you luck with that effort --

Shekhar Borkar: Thank you.

Justin Rattner: -- leading that effort for Intel. And I'm sure you'll be back here in a few years --

Shekhar Borkar: I'll be back in a few years.

Justin Rattner: -- sharing more of the progress with us.

Shekhar Borkar: Thank you.

Justin Rattner: Shekhar Borkar, everyone. [applause] These guys have all the fun. Well, as we always do, we come to the keynote on Thursday to talk about the future. And I think as you've heard through the week, the future is very important to us at Intel. To get a better idea of the future we want, we've actually embarked on something we call the Tomorrow Project. And a few of you may have heard of that, particularly if you were down in the Technology Showcase and visiting the Theater of Tomorrow.

We've been doing all kinds of interesting projects. We've been interviewing influential people, you know, talking to thought leaders, working with science fiction writers who do a wonderful job imagining the future. But I think it would be best summarized for you

if you heard from Brian David Johnson, our resident futurist at Intel about the Tomorrow Project. So, let's watch this video.

[Video]

Justin Rattner: And we do want to invite you to join the conversation, to be part of the Tomorrow Project. You can do that by visiting the Theater of Tomorrow in the technology showcase. That'll be open the rest of the day. And if you don't make it, you can always go on the web to TechResearch.Intel.com/TomorrowProject and get involved that way.

Well, it's been quite a week at IDF, I think you'll have to agree. Paul got things kicked off by describing where we are in the evolution of personal computing, and Mooly came yesterday to show us some of the truly elegant technology that's just around the corner for personal computing. I know I'm going to be lining up to get one of the Ultrabooks that you people will create and make that part of my computing stable. And I've just described for you the start of a decade-long effort to take computing to new extremes of energy efficiency and performance.

I think what that means to all of us, and I know when I step back and think about it, I realize we're at a very significant point in time. It's a time where technology is no longer the limiting factor. What limits us today is really our own imagination. And given that, I want to challenge each and every one of you here to create your own vision of the future, the future you'd like to invent. Why? Because if you can dream it, we can invent it together. Thank you, and see you next year.

[Music]

[End of Recorded Material]