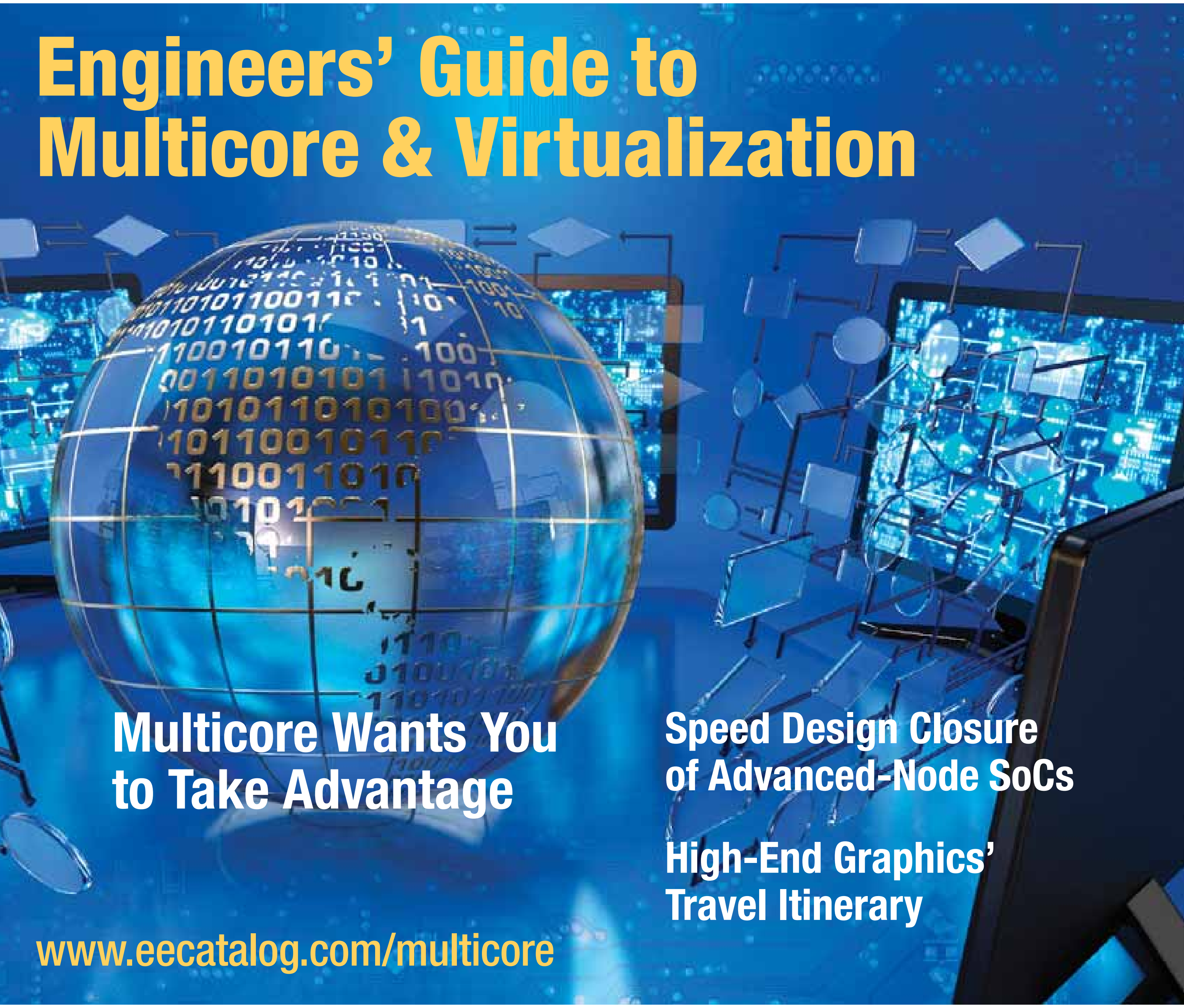
EMBEDDED SYSTEMS ENGINEERING

powered by EECatalog

Guiding Embedded Designers on Systems and Technologies

Engineers' Guide to Multicore & Virtualization



Multicore Wants You to Take Advantage

Speed Design Closure of Advanced-Node SoCs

High-End Graphics' Travel Itinerary

www.eecatalog.com/multicore

Platinum Sponsors

freescale

WIND

Imagination

Silver Sponsors

SILEXICA
multicore meets simplicity

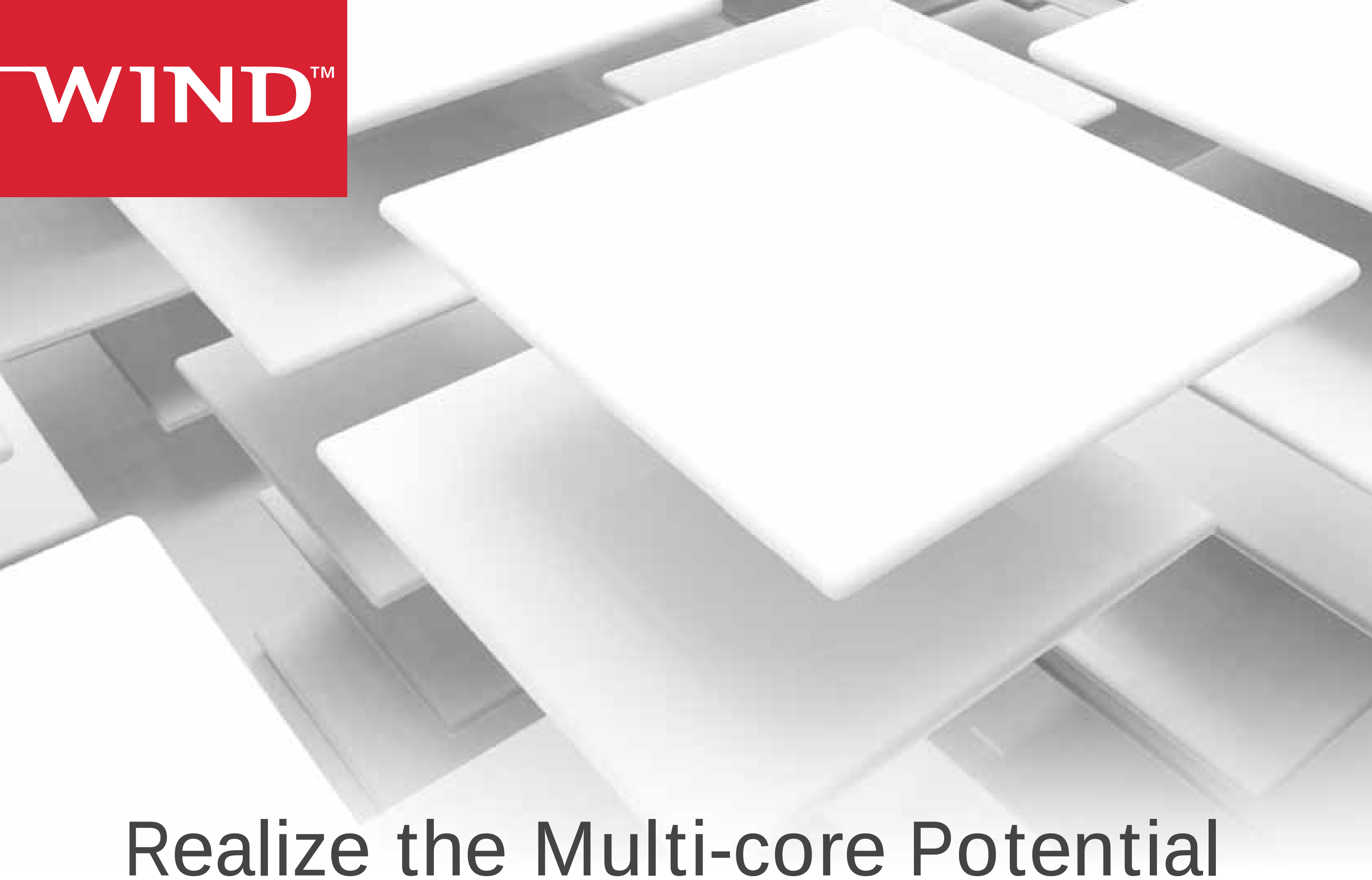
SYNOPSYS®

SONICS

Texas Instruments

Bronze Sponsor

tmt
Texas Multicore Technologies, Inc.



Realize the Multi-core Potential

There's no question that multi-core technology brings higher performance and lower power consumption to a broad range of systems and devices. The question is how best to take advantage. Wind River® is the one vendor that brings a comprehensive software solutions approach to multi-core enablement. We bring together the right software, tools, technologies, training, and support; and we provide all of the key building blocks organizations will need today, tomorrow, and into the future.

Performance

Consolidation

Migration

Safe and Secure Partitioning

Contact us to find out how Wind River can help you create the next generation of systems and devices.
www.windriver.com, 800-545-WIND

VxWorks 7

THE SAFE AND SECURE RTOS FOR THE INTERNET OF THINGS

VxWorks® delivers unrivaled deterministic performance on multi-core and sets the standard for a scalable, future-proof, safe, and secure operating environment for connected devices in the Internet of Things. The VxWorks portfolio of add-on technology profiles and industry-specific profiles enables customers to differentiate their platforms with best-of-breed capabilities, reduce development costs, and accelerate time-to-market.

Industry-Specific Profiles

- Aerospace
- Consumer
- Industrial
- Medical
- Networking

Technology Profiles

- Microkernel
- Safety
- Security
- Virtualization

Platforms for Certified Environments

- VxWorks 653 3.0 Multi-core Edition
- VxWorks MILS, Multi-core Edition
- VxWorks Cert

www.windriver.com/products/vxworks/

Wind River Linux

OPEN SOURCE INNOVATION READY-MADE FOR THE INTERNET OF THINGS

Wind River Linux is the market-leading industry standard for embedded Linux software and offers the high performance needed for scaling to large multi-core systems. Wind River Linux is also a highly cost-effective alternative to roll-your-own Linux, offering up to 98% in savings over the lifecycle of your device. With 24/7 award-winning long-term global support and professional services, our maintenance, compliance, and security monitoring teams are with you every step of the way.

Technology Profiles

- Open Virtualization
- Carrier Grade
- Security
- Infotainment

www.windriver.com/products/linux/

CONTENTS

EMBEDDED SYSTEMS ENGINEERING

Features

Engineers' Guide to Multicore & Virtualization
www.eecatalog.com/multicore

Vice President & Publisher
Clair Bright

Editorial
Editor-in-Chief
Chris Ciufo
cciufo@extensionmedia.com
Managing Editor
Anne Fisher
afisher@extensionmedia.com
Contributing Editor
Caroline Hayes

Creative/Production
Production Manager
Spryte Heithecker
Graphic Designers
Nicky Jacobson
Caldin Seides
Media Coordinator
Stephanie Bradbury
Senior Web Developers
Slava Dotsenko
Mariam Moattari

Advertising / Reprint Sales
Vice President, Sales
Embedded Electronics Media Group
Clair Bright
cbright@extensionmedia.com
(415) 255-0390 ext. 15
Sales Manager
Michael Cloward
mcloward@extensionmedia.com
(415) 255-0390 ext. 17
Marketing/Circulation
Jenna Johnson

To Subscribe
www.eecatalog.com

Extension MEDIA

Extension Media, LLC Corporate Office
President and Publisher
Vince Ridley
vridley@extensionmedia.com
(415) 255-0390 ext. 18
Vice President & Publisher
Clair Bright
cbright@extensionmedia.com
Vice President, Business Development
Melissa Sterling
msterling@extensionmedia.com
Human Resources / Administration
Darla Rovetti

Special Thanks to Our Sponsors

freescale · Imagination · WIND · SILEXICA · SONICS · synopsys · Texas Instruments · tmd

Multicore Hopes You'll Take Advantage

Unlocking the performance multicore processors offer has implications for everything from machine vision to network function virtualization to security.

By Anne Fisher, Managing Editor

Raghu Rao
AMD

Dr. Arnon Friedmann
Texas Instruments

Dr. Paul Anderson
GrammaTech

Weihua Sheng
Silexia

Mark Throndson
Imagination Technologies

Drew Wingard
Sonics

Mark Nadeski
Texas Instruments

Our panel comments on multicore and the roles software, tools, and standards play in its success. [Editor's note: An expanded version of this article is available online]

EECatalog: It's hard to find a 32-bit single core (contemporary) processor. Yet there is some evidence that software—both written/executable, and the tools which help create that code—still doesn't truly take advantage of the performance multicore processors offer. Why is that?

Raghu Rao, AMD: Parallel programming on multicore processors is inherently harder than single threaded programming. Splitting an application across multiple cores is inherently a hard thing to do. You have to think at an entirely different level when dealing with multiple threads of execution that need to communicate with each other, share data and avoid race conditions and resource deadlocks, etc. This is likely the main reason most programmers shy away from multi-threading. This is exacerbated by the fact that languages do not make it easy to write code for multicore. As a result, most programmers and developers don't even try to tackle multi-threading—even for portions of applications that could benefit and accelerate.

In the High Performance Computing (HPC) space, this is available in programming models like OpenMP, and mainstream languages like C++, Java and Python are evolving to enable easier use of multicore. Additionally, recent systems architecture innovations are enabling these multicore paradigms to work for heterogeneous compute as well. Of particular interest is the Heterogeneous Systems Architecture (HSA) standard (recently finalized) that includes features like shared virtual memory, coherency, platform atomics and device enqueue. The standard will help to simplify the programming model, making it much easier to develop code that takes advantage of multicore.

Dr. Arnon Friedmann, Texas Instruments: Most multicore code today is migrated from single core code or developed with a single core in mind, and it therefore doesn't algorithmically take into account how to best partition the problem so that it works optimally across multiple cores. When developing code to run efficiently on multiple cores, a more holistic view of the original algorithm must be taken so that a program can expose the concurrency in that algorithm. Most engineers either don't have the experience or don't take the time to re-architect their code for multicore.

One way to tackle this problem is through parallel programming languages like OpenMP, which allow programmers to incrementally add parallelism to a 'serial' program. As multicore tools continue to improve, it will become easier to take advantage of the processing power of modern processors. To this end, standards are important so that vendors can focus their efforts on improving a 'limited' number of languages/models and supporting tools.

Dr. Paul Anderson, GrammaTech: With multicore systems, the key challenge is to design an architecture that can keep all of the cores doing useful work. Programmers accustomed to writing for single cores have to learn entirely new techniques to exploit the full potential of the hardware. Worse, multi-threaded programs are vulnerable to entirely new classes of bugs, such as deadlocks, livelocks, data races, starvation, and lock mismanagement. These can be very difficult to diagnose because they are usually non-deterministic, they may occur rarely, and they can have mysterious symptoms. Finding and eliminating these can be extraordinarily time-consuming and requires new tools and debugging strategies.

Weihua Sheng, Silexia: The revolutionary switch from single core processors to multicores is really evolutionary from the technology point of view—the problem of power consumption and thus heat dissipation forced SoC providers to look into placing multiple processors on a chip. Just like a one-way street, it is highly unlikely that vendors will make the U-turn. However, the software development process for multicores was only considered as an afterthought in the transition, and now it lags much behind what is demanded. The only way to solve this is to bring in expertise from both hardware and software communities and advance compilation and optimization technologies for multicores. We, Silexica, are glad to join and contribute to this effort.

Mark Throndson, Imagination Technologies: The long-standing methods of achieving higher performance through increasing clock frequency, process migration and maximizing single-thread CPU performance have yielded diminishing returns for some time. The move to multicore was a natural progression to enable performance scaling.

Today, there is enough parallelism in many software workloads such as video codecs and image processing software to readily make use of multiple cores or threads in a CPU. With their non-interdependent, packetized code, many networking workloads are also easily parallelized.

Drew Wingard, Sonics: Actually, if you look at the embedded space, you find the migration from 8-, 16-bit processor architectures to 32-bit architectures has gained momentum. See Sonics customer Microchip as a prime example of this trend. The number of 32-bit processor implementations that are available to a broader range of embedded systems designers has exploded. Most of those are single core.

This is true for both the MCU and the MPU market spaces. The market is segmented into the traditional microcontroller space, which typically doesn't need external DRAM, and the MPU segment, which typically does require external DRAM. Much of the MPU market is still single-core, 32-bit processors.

EECatalog: What are some of the best tools—or coding practices—to create multi-threaded, multicore programs?

Mark Nadeski, Texas Instruments: TI recommends using OpenMP and OpenCL to create multi-threaded, multicore programs. These are both widely adopted standards and are supported by many open-source tools for debugging and performance analysis. These can be used in a staged tooling flow from high-level, where the tools do all the work, to low-level where the programmer works on detailed multicore code development.

For higher-level (or simpler) development, runtime libraries in conjunction with OpenMP can be used. The processor vendor supplies multi-threaded libraries that are optimized for their architecture, i.e., an optimized FFT function on a DSP. The programmer uses OpenMP directives to launch the processing across the multiple cores, and the mapping is handled automatically by the compiler or runtime code.

An example of lower-level development would be where the programmer rewrites their code to expose parallelism and uses OpenCL or Pthreads to map code explicitly onto the different cores. Note that this model can also take advantage of runtime libraries, but here the parallelism is being implemented by the programmer rather than the tools.

Sheng, Silexia: As the computing spectrum is so wide and multicore processing is virtually everywhere, there will be no one-size-fit-all solutions. The best tools are subject to domain specific requirements, e.g. application processing, product features (e.g. timing constraints) and target architecture characteristics (homogeneous or heterogeneous). Those tools that understand and interpret those requirements well enough will win in the end.

Rao, AMD: The aim is to create threads that are managed by the run-time rather than by the programmer. Threading techniques that require the developer to use threading libraries to create and manage threads are inherently difficult to use. If you can get the run-time to manage parallel execution without exposing the complexity to the programmer, that's clearly a win. Libraries can do this too by providing a simple API to programmers while using data or task parallelism under the covers. This is a very promising way to utilize multicore and heterogeneous resources on the platform without requiring programmers to come up the parallel programming learning curve. OpenCL is an example of a programming model, and OpenCV 3.0 is an example of a library with a transparent API that allows programmers to implement multi-threaded, multicore programs.

Throndson, Imagination Technologies: The way we've designed our multi-threaded CPUs—such as the MIPS Warrior I6400—makes them appear as multicore CPUs to Symmetric Multiprocessing (SMP) operating systems. So when it comes to multicore versus multi-threaded, they are supported by the same software programming model.

In addition, thanks to advancements in heterogeneous compute, developers today can also use the GPU for the heavy-lifting part of an algorithm. Because the GPU is inherently a massively multi-threaded machine, it can handle these types of tasks much more efficiently.

Anderson, GrammaTech: A radical-sounding approach is this: don't use threads! Instead, write single-threaded programs that use process-level parallelism. There are two good reasons for this: confidence and resiliency.

Confidence: The biggest hazard of multi-threaded programming is that it is hard to avoid concurrency bugs. If you use multi-threading, then you must be very sure that all the code that may execute in parallel is thread safe and free of concurrency errors. Even if you are convinced that your own code is good, there may be bugs lurking in the third-party libraries you use. A design that prohibits the use of threads entirely is one that rules out entire classes of concurrency bugs.

Resiliency: A memory access error in a multi-threaded program will cause the entire program to fail, regardless of which thread it occurs in. In contrast, the same error in a process will kill only that process. A well-architected system will be tolerant of such a failure and will be prepared to work around it.

There is a cost, of course, but modern operating systems are good at managing efficient communication between separate processes, and there is an increasing number of real programs that successfully use process-level parallelism.

Admittedly, there are cases when this isn't possible, such as when coding on the bare metal. In such cases it is important to use every technique available to find and eliminate threading bugs. Static analysis is recommended because it can reason about executions that are not exercised by dynamic testing.

EECatalog: The HSA Foundation recently introduced specifications and guidance on creating heterogeneous (inherently multicore) systems. What is the significance of this announcement? Are heterogeneous systems in our future?

Friedmann, TI: Heterogeneous systems with a mix of processing solutions are prevalent today and will likely remain so in the future. Each of these architectures has different and often complementary strengths and often times the optimal system solution will utilize two or more of these different architectures. We have seen this for many years in high-end systems across a wide variety of areas including medical imaging, mission critical and high performance computing.

TI is a member of the HSA Foundation and our architectures support the HSA programming model. The HSA computing model provides an open, industry-standard approach to developing low-level API programming models for heterogeneous multicore. The significance of HSA will largely depend on its adoption across multiple architectures, which would allow developers to easily port code to the heterogeneous community and could be used in conjunction with more mature standards like devices from different vendors. This would clearly be of great benefit to the OpenMP and OpenCL.

Anderson, GrammaTech: Heterogeneous systems are already with us. The HSA specification brings some order to the software developers by providing a standard set of abstractions for programming these systems, and we can expect to see useful libraries of reusable code emerge that build on the HSA interface. However, the kinds of processor that HSA is designed for are quite different from standard homogeneous multi-core systems, and the hazards that homogeneous multi-core systems are vulnerable to are very different from those that are most likely to occur in heterogeneous systems. It is unlikely that the HSA specification will be hugely significant for conventional multi-threaded programming for homogeneous multicore systems.

Peter McGuinness, Imagination Technologies: With heterogeneous processors, and specifically The HSA Foundation with its promise of support for high-level programming languages, developers will be able to access the compute performance they need without having to pay the penalty of the high power consumption that accompanies arrays of CPUs. We can expect to see a reversal of the trend towards multicore and a new short-term trend towards GPU hardware support. I'd expect to see a trend toward more general heterogeneous computing in the longer term that will possibly incorporate general-purpose DSPs or specialized programmable engines in the vision pipeline, for instance. This is an important shift away from programming for a CPU architecture and towards programming for a system architecture, reinforcing trends already underway to reduce or eliminate the importance of CPU instruction sets.

Wingard, Sonics: HSA adds a software layer that allows a scheduler to pick a single program and split it across processors of different types. This is an interesting and aggressive idea. It is expensive from a hardware implementation perspective because it requires capabilities like coherence and virtual memory support, etc.

Large semiconductor companies are doing more than experimenting with these techniques. The situations in which these HSA-compliant systems are most interesting to them are the ones that are designed with the least knowledge of the end application. To the extent that they are going to build general-purpose SoCs that mix and match these types of resources and can provide more optimum results for programs that are known when the chip is designed, these techniques are interesting.

However, if you look at a domain like application processors for phones, where we have a much better sense for the operating modes of the device, we can do a better job both from a cost perspective and especially from an energy/battery life perspective by optimizing the SoC architecture to solve the required use case.

Rao, AMD: The announcement is huge, because it introduces real standards for building hardware that enable all of the above languages to use the GPU just like a multicore CPU. With no changes to languages that are already going multicore, someone can get heterogeneous computing off the ground.

There are many examples of usage models that cannot be accomplished in reasonable timeframes or power envelopes without parallelism: video and image processing, machine vision, machine learning, big-data analytics, climate simulation, and so on. Providing a standard architectural approach to exposing heterogeneous parallelism provides a multiplier effect for the industry since software tools and library vendors can write to a single target and run across any platform that supports the standard.

Using Physically Aware Synthesis Techniques to Speed Design Closure of Advanced-Node SoCs

At smaller process nodes, chip designers are struggling to meet their aggressive schedules and power, performance, and area (PPA) demands in the ever-so-competitive system-on-chip (SoC) market, from machine vision to network function virtualization to security.

By Gopi Kudva, Cadence

At 28nm and below, SoCs are much more complex, making it more challenging than ever to meet PPA targets. Wires dominate the timing at these advanced nodes, so there's a greater chance of encountering issues such as routing congestion and timing delays. You must cram more transistors into the die, and have to reduce dynamic and leakage power.

Physically aware synthesis – the ability to bring in physical considerations much earlier in the logic synthesis process – is something that can dramatically improve the design process and significantly shorten the time spent fixing problems. Let's discuss some key physically aware synthesis techniques that can help you speed up the physical design closure process for your next high-performance, power-sensitive SoC.

PHYSICALLY AWARE SYNTHESIS

Today's physically aware synthesis technologies bring physical interconnect modeling earlier into the synthesis process to help you create a better netlist structure, one that's more suitable for today's P&R tools.

You can start with no floorplan, and allow the synthesis to come up with one. You can give it a very basic floorplan. But the better the floorplan you have, the better you can take advantage of global synthesis optimization with the more detailed physical interconnect. Essentially, you are getting rid of the old logical-physical barrier. You'll no longer need to, with fingers crossed, wait for your "backend" engineer to say "yay" or "nay."

There are four physically aware synthesis innovations: physical layout estimation (PLE); physically aware mapping (PAM) physically aware structuring (PAS); physically aware multi-bit cell inferencing (PA-MBCI).

Of course, before you can come up with a good floorplan, you need to have a good initial netlist. To create that initial netlist, you can still use physical information and use physical layout estimation (PLE). For this, you just need some basic physical information, such as LEF and cap tables/QRC tech files. The floorplan DEF is optional here.

PLE is a physical modeling technique for capturing timing closure P&R tool behavior for RTL synthesis optimization. It allows you to create a good initial netlist for floorplanning. And the result? Better timing-power-area balance. PLE uses actual design and physical library info, dynamically adapts to changing logic structures in the design, and has the same runtime as synthesizing with wireload model.

Once you have a good initial netlist, you can create a good initial floorplan. Previously, this floorplan was used for P&R stages, and not in synthesis. But now, you can use this floorplan to allow the synthesis engine to "see" long wires before actually building the logic gates for the improved, physically aware netlist.

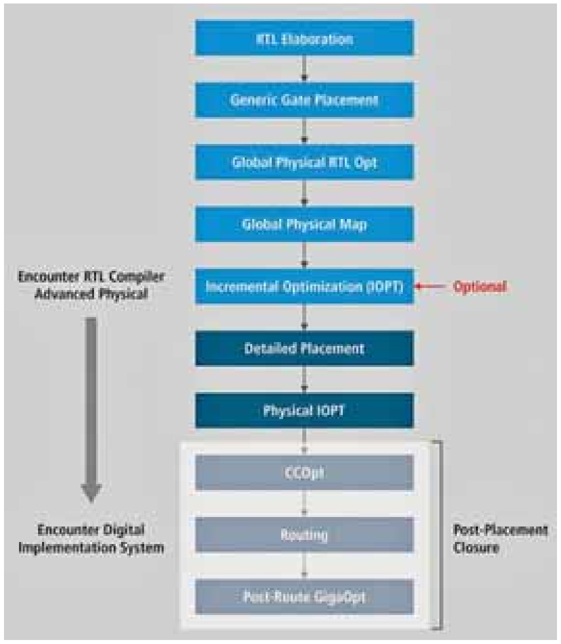


Figure 1. Physically aware RTL synthesis flow

Figure 1 shows the main steps in the latest physically aware RTL synthesis flow: generic gate placement; global physical RTL optimization; and global physical mapping.

PHYSICALLY AWARE MAPPING (PAM)

All about improving timing with increased correlation, PAM initially places the optimized generic gates and macros, and it optimizes the placed generic gates and macros (RTL level optimization, including datapath optimization).

- Estimates routes and congestion for the placed generic gates and macros, taking into account physical constraints such as placement and routing blockages
- Performs parasitic (Resistance and Capacitance) extraction using a unique extraction method on the estimated routes

After generic gate placement, every wire in the design has a physical delay. The synthesis engine can now accurately "see" which paths are critical. Global synthesis now does timing-driven cell mapping based on physical wire delays, translating the generic gates into standard gates based on the provided technology library and creating an optimized netlist.

By considering real wire delays, PAM has demonstrated the ability to deliver up to 15% improved timing. After all, if you know in advance that a certain wire will be long and you know where that extra delay is because of the long wire, you can structure the netlist more accurately to account for these delays. With this knowledge, synthesis is also in a better position to "squeeze" critical paths and "relax" non-critical paths based on wire delays.

PHYSICALLY AWARE STRUCTURING (PAS)

What PAS does:

- Provides optimized binary/one-hot multiplexer (mux) selection
- Targets high-congestion structures, such as cross bars, barrel shifters, and memory-connected mux chains
- Decomposes a large mux into a set of smaller muxes, each of which can potentially share the decode logic. Decoding logic, in turn, is intelligently partitioned using physical input pin knowledge.
- Generates congestion-aware decode islands via smarter select line sharing and duplication

The result of PAS: better placement that decreases routing congestion.

To illustrate the benefits provided by PAS and PAM, we considered a Flash memory design whose floorplan had a small channel of digital logic surrounded by Flash memory. This design suffered from congestion and timing issues due to a poor logical synthesis wire model. Timing closure was impossible. Once the engineering team utilized Cadence® Encounter® RTL Compiler Advanced Physical Option, which features the physically aware synthesis capabilities we have been discussing, TNS improved from ~12,400ns to ~750ns. The technology helped improve timing correlation by identifying long paths during physical synthesis and it also helped identify and alleviate congestion. In the end, the engineering team was pleased to experience significantly reduced design turnaround time and synthesis to place-and-route iterations.

As another example, we have a networking SoC with a one million instance block and with a large volume of muxes. Initially, with traditional synthesis, the engineers working on this design had initial significant horizontal and vertical congestion, hence the design was not routable. Using the physically aware capabilities of Encounter RTL Compiler Advanced Physical Option, the engineering team met their timing and area goals with a routable design with little congestion.

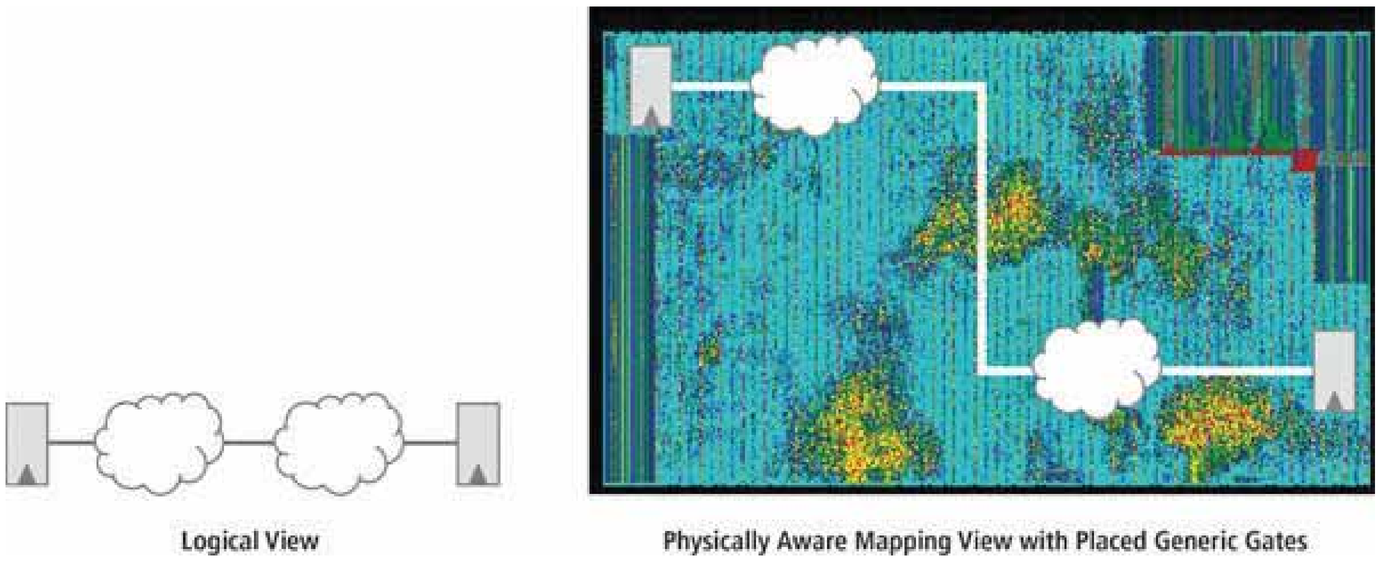


Figure 2. Physically aware mapping accounts for long wire delays in RTL synthesis

Typically, just to get the design to route, engineers have to "pad" the layout so much in order to account for the bad structure of the netlist! The wires are also longer due to extra spacing the padding creates, and leads to extra buffering and increased power. With physically aware synthesis, you can easily remove the extra padding and margins, thereby reducing area significantly and shrinking the die, lowering the wire length and power.

PHYSICALLY AWARE MULTI-BIT CELL INFERENCING (PA-MBCI)

Multi-bit cell inferencing (MBCI) merges single-bit flops into a multi-bit version of flops. Using a physically aware MBCI (PA-MBCI) synthesis strategy can help reduce total chip power—10% or better dynamic power savings in many cases!

MULTI-BIT FLOPS – ADVANTAGES AND BEST PRACTICES

As an example of the benefits of using a MBCI flow, let's take a look at the impact of this flow on development of a design based on an advanced-node embedded processor. Compared to using traditional synthesis techniques, applying physically aware synthesis to this processor yielded:

- 15% clock tree area
- 60% TNS (improved hold timing)
- 6.4% dynamic
- 4% leakage
- 4.7% routing

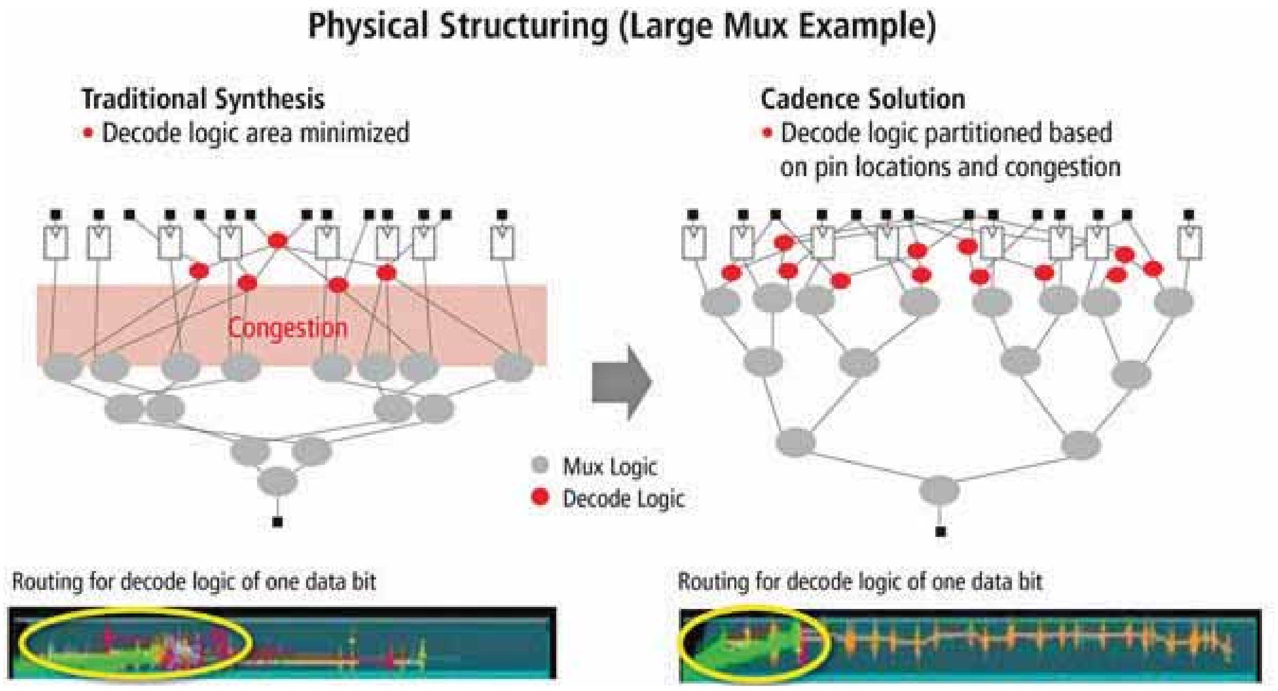

Physical Structuring (Large Mux Example)
Traditional Synthesis
• Decode logic area minimized
Congestion
Cadence Solution
• Decode logic partitioned based on pin locations and congestion
Mux Logic
Decode Logic
Routing for decode logic of one data bit
Routing for decode logic of one data bit

Figure 3. Physically aware structuring RTL synthesis flow

In synthesis, you can merge single-bit flops into a multi-bit flops using either a logical or physical method. A logical method is where synthesis considers only the netlist and converts as many flops into multi-bit flops without considering the flop locations and proximity. The disadvantage of this method is that you could end up with flops at two opposite ends of the floorplan merged, creating a placement problem and unnecessarily long wires, which in turn can create timing and routing problems.

Encounter RTL Compiler Advanced Physical Option features physically aware multi-bit merging. Physically aware multi-bit merging merges the sequential cells while considering the compatibility and physical neighborhood from the natural placement. This is a "correct by construction" process which makes sure flops are merged after placement, only when there is benefit in a specific cost factor (typically timing, area, leakage, and dynamic power), while not degrading other cost factors.

The result: the PA-MBCI process avoids timing degradation, reduces wirelength, minimizes congestion, and reduces power.

TIPS AND TRICKS

To get optimal results from physically aware synthesis, consider these techniques:

- For generating an initial netlist, use PLE
- Use this PLE netlist to create a starting floorplan
- With this floorplan, perform synthesis staring from RTL
- Enable PAM
- If your design has high-congestion structures, such as cross bars, barrel shifters, and memory-connected mux chains, enable PAS
- If you have multi-bit flops in your technology libraries, enable PA-MBCI
- Now you have a physically aware netlist: use Encounter RTL Compiler Advanced Physical Option to perform standard cell placement and optimization
- Note that in this article, we only discussed generic gate placement and not standard cell placement.

High-End Graphics have a Travel Itinerary: Conversation with Peter McGuinness, Imagination Technologies

High-end graphics are refusing to stay in high-end devices, instead hitting the road and making themselves at home in everything from doorbells to driver assistance systems to washing machines to printers and beyond.

By Anne Fisher, Managing Editor

Peter McGuinness, Imagination Technologies

One of the trends Peter McGuinness, director of multimedia technology marketing, Imagination Technologies, remarked on during his Q&A with EECatalog is that the mobile space is hankering for "what at least looks like and behaves like a smartphone, but does not actually have a real smartphone inside it—so it's high-end graphics." McGuinness has been keeping an eye on the trend of high-end graphics migrating to the low end and on its manifestation as a high-end user interface look. He notes the trend is a factor behind the company's announcements at MWC last month, which saw Imagination announcing a highly optimized, tiny 4-core GPU that brings Imagination's high-end PowerVR Rogue graphics to cost and area-constrained devices such as wearables and IoT.

Following are edited excerpts from our EECatalog Q&A with McGuinness:

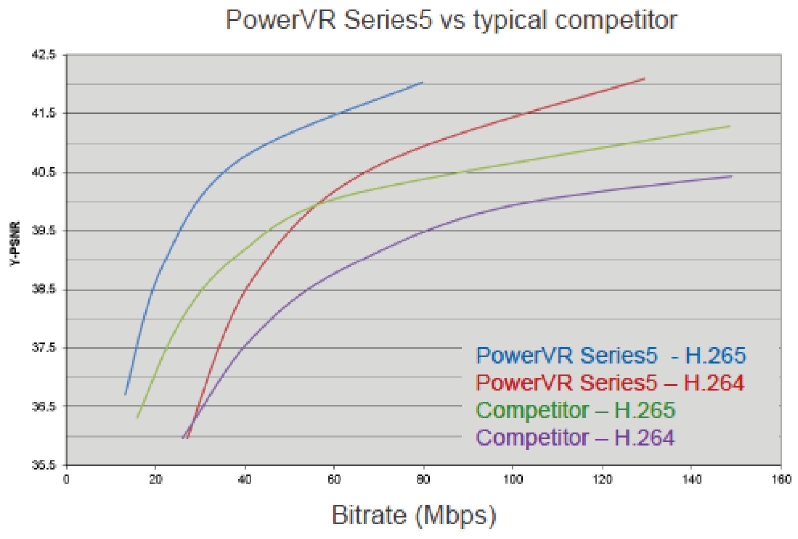


Figure 1. McGuinness comments, "…if you compare our selected H.265 bit rate at that selected point, you see that it is achieving just over 20 Mbps for that quality, whereas if you look at the green line (the competition), that same quality results in a bit rate of over 40Mbps." [Image courtesy Imagination Technologies.]

EECatalog: What trends are you seeing?

Peter McGuinness, Imagination Technologies: Camera and vision apps are huge, and wearables are starting to come into their own.

The technology that is being driven by these trends [means] we are seeing a lot of always on/always aware devices, and that has an impact on the technology that we have to provide—in particular, things like deep sleep modes so that the awake device can be minimally aware of being woken up—and, when appropriate, progressively woken up so that we can preserve battery life and have a device that lasts for a useful length of time.

We are going to continue to see large multifunction devices put into cell phones, but we are also going to see diverse development of remote sensors, which will feed into that, and I think that is quite established as a development track.

The GPU compute story is very strong. People are using the GPU, using OpenCL and other programming languages on the GPU to support vision applications and computational photography in handsets and mobile—it's a very strong trend.

EECatalog: What does this trend of high-end graphics making their way to low-end devices mean for the direction Imagination is taking?

McGuinness: We saw a slot in the market for an area optimized GPU [the G6020] that supports OpenGL ES3.0, which is more than is needed today for the minimum OS requirements, but we believe that is the direction the market is moving in anyway. It's beneficial because it gives programmers the same programming model from low- to high-end devices.

The OpenGL ES3.0 API interface is significantly different than the API that came before it, but it's very similar to iterations of API that have come after it, and that is very beneficial when you are developing applications. We made the G6020 GPU very small to address the needs of devices with limited silicon area and bandwidth. And we have optimized it specifically for the use cases that we see as being relevant to this market.

The application space that we are targeting with the G6020 GPU does not require lots and lots of visual effects that demand complex shaders. [For applications that are] user interface oriented, you do have a lot of layers, and you do have blending going on between the pixels. However, you don't need to run the high-precision shaders, and you don't need a lot of geometry transforms.

Some of these applications are entry-level phones, where we are expecting those phones and other equipment to have screen sizes up to 720p. To address this, we size the performance of the device to give it very good, very responsive User Interface (UI) performance at 720p, with a 60Hz refresh rate [so as to achieve] quite a sophisticated user interface. So Android will work well. Chrome OS would also be a very good candidate for this. For some of the higher-end wearables, which have a screen, we can clock it down to meet the necessary power profile.

Tablets and devices with a GUI in the commercial enterprise world are additional examples. Multi-use printers are starting to incorporate quite sophisticated screens. We also seeing applicability for home automation stuff, washing machines and door entry control consoles—to name a few.

And in the automotive world, at the low end, where you have a basic UI, it would be appropriate for that as well.

The characteristics of using the GPU at this level differ (somewhat) from the characteristics needed at the more general level. We [modified] the Unified Shading Cluster (USC) to make it more suitable for pushing pixels, so it has a higher fill rate and is [therefore] more suited to driving a smooth, liquid user interface rather than high-performance 3D graphics.

EECatalog: Imagination announced a High Efficiency Video Coding (HEVC) IP family at MWC—what are the key take aways there?

McGuinness: The promise of [the H.264 and H.265] encode standards are very high compression ratios. Ultra High Definition TV is driving H.265, similarly to how HD drove H.264.

With video compression being the driving factor, we wanted to produce IP that delivers on the promise of H.265. That is, you get 50 percent better bit rate, which is 50 percent better compression in the same format by comparison with H.264.

We saw that earlier introductions by other firms of H.265 encoders don't implement the encoding toolkit in such a way that you get the real promise of H.265. Typically we get a 30 percent lower bitrate at equivalent quality in comparison with the competition. That is a key feature because it is that extra compression which is really going to drive the viability of 4k in consumer and prosumer equipment.

EECatalog: Compare different approaches to implementing the encoding toolkit for H.265.

McGuinness: The competition is implementing less of the encoding toolkit [than we do]. [This approach] means that then they can build a cheaper device, which occupies less silicon area. And of course, it takes less time to design, so they are able to get into the market faster. Those are the driving factors behind their design decisions, but the effect of [those decisions] is to have a mediocre bitrate outcome. In order for the competition to close the gap of bitrate versus quality, they would need to redesign their architecture. [Should they do so] they would lose their time to market advantage and they would lose their area cost advantage.

EECatalog: Where are you finding opportunities to make your whole solution more than just the sum of its parts?

McGuinness: We own the major blocks of the multimedia subsystem, and we have been working over the past few years to make sure that they interoperate in the most efficient fashion. For instance, we have the camera ISP, and we have created a private streaming path between that IP and the video encoder IP so that we have direct IO.

You can capture the image directly from the camera, stream it into the encoder and out through the transmission channel in bitstream form, without the images having to touch DRAM. You give up something in compression, because you are not going to be using B-frames, but you get an enormous power advantage for the application.

The common thrust of all of our IP families is that we have gone for a combination of the highest quality and the lowest possible bitrate because we think that is the real value proposition of H.265 (Figure 1).

EECatalog: What comments do you have from a high-altitude perspective about the IoT?

McGuinness: We work with a lot of people who are creating products for the IoT, and one of their difficulties, and one of the things that we have seen, is that the with first iteration of IoT—and Google Glass is a good example of this—they have taken devices, SoCs, from merchant semiconductor companies, and they have tried to repurpose them to fit the requirements of the IoT.

And so for instance Google Glass, the one that has just been discontinued, has TI OMAP 4 in there, which is a cell phone chip. That repurposing really has not worked because the mobile phone SoCs are a bit of a Swiss army knife, they do a bit of everything, whereas the IoT SoCs need to be (I would not say single-function, exactly) but they need to have a more restricted set of functionality, and they certainly need to be more fine tuned in order to meet the power and cost and actual physical size constraints of the IoT.

So we are seeing a number of companies, a lot of companies in fact, becoming "vertical" in the sense that they want custom SoCs designed for them but they don't see that coming from the merchant semiconductor companies, who have different imperatives.

In response to the pressures of IoT and the absence of specific chips coming from the semico's, companies are coming to us, they are having us work closely with them to design optimized chips, and then they are taking it to a provisioning company, using a foundry. Then they source their own chips directly from the foundry in order to get the fine-tuned, exact chip needed for their market.

Examples include a company called Ineda Systems that is designing a very clever niche space wearables-oriented chip. And a company called Toumaz Group in the UK, which is operating in the medical appliance sphere.

EECatalog: What are the problems that Imagination and others in your market space are wrestling with?

McGuinness: Certainly in the mobile world the big problem that everyone has now with very large devices with many, many millions of transistors, and with Moore's Law driving [us] toward finer and finer geometries, is the problem of dark silicon.

We can put so many transistors on a chip now, and those transistors can be clocked so fast, and hence burn so much energy, that we cannot afford to turn them all on at the same time, so some of the silicon has to be dark at any one time.

And the consensus approach to some of these problems is to look at the high-cost items with regard to power—bandwidth utilization, throughput and the energy per operation.

One of the strengths that Imagination has is that we have all of the major elements needed for our customers to actually produce an SoC. We control all of those elements. So we are able to do things like create a standard compression format that all of those elements can share, so you can compress everything that goes in and out of memory, DRAM, and that reduces power dissipation by an enormous amount.

We are focused on the energy-per-operation of all of the things that we do, [even] focused to the extent that sometimes we irritate our industry collaborators on standards bodies, because we understand well that down to the very low level of implementation detail, it is really important, to use, for instance, the right data quantities, even in your software, in order to get the best energy efficiency.

For example, one of the optimizations that we made for the G6020 computing cluster, the shading cluster, was that instead of having 32-bit floating point throughout, which you can do, we changed the balance, so we've got a bit of 32-bit floating point capability, but the majority is in16-bit, which for that application space where users want a feature phone with a smartphone interface, where you have high-end graphics migrating to the low end in a certain form, is perfectly adequate. For a given operation, by using a 16-bit you reduce the energy required for that operation.

Another example: The OpenCL Embedded Profile from Khronos allows you to have different ways of doing rounding and much lower energy per operation in floating point functions, and that is the profile that we implemented in our GPUs.

The other OpenCL profile from Khronos, the Full Profile, has much higher precision than floating point. It has 64-bit floating point as standard, and is aimed at high-performance computing. It is not aimed at embedded.

We know of one GPU vendor that opted for the Full Profile OpenCL, which made its OpenCL solution unusable because it was too power-hungry. For example, attempts to run an OpenCL app cause the power management software to come in and shut the GPU down because the GPU is gobbling too much power and threatening to destroy the chip.

Plenty of Room for Innovation: Interview with AMD's Raghu Rao

Machine vision is just one of the applications spanning industrial, consumer, medical and more that are poised to benefit from innovations that make it possible to unleash the performance of a platform's various cores.

By Anne Fisher, Managing Editor

Raghu Rao

Our thanks to Raghu Rao, senior director of engineering for Embedded Solutions, AMD, who recently offered his insights on a number of topics, including making "large amounts of compute available for platforms that are easy to design in and program and are easy to access…."

EECatalog: What are the primary challenges when working on machine vision applications to help end customers scan, inspect, track and solve?

Raghu Rao, AMD: Designers struggle in two main areas. First, system design is complex—all the way from the custom hardware including ASIC, FPGA, DSP, needed to deliver the required performance—through developing optimized software to target and extract these hardware capabilities. Balancing cost and performance is always a challenge. Custom design increases R&D costs and time to market and can also increase project risks.

Second, new algorithms and new methods pose a challenge to the feasibility of implementation. When fundamental design changes to software and hardware occur, these algorithms and methods are impossible to implement without multiple teams. Not only is this hard to plan and execute, the outcome is often unpredictable and has an impact on time to market and competitiveness.

EECatalog: If machine vision is the "killer app" for OpenCL, what is AMD's role in making some killer apps "more equal than others"? And how does the ecosystem need to further mature?

Rao, AMD: Programming models like OpenCL and OpenMP enable hardware capabilities such as heterogeneous and multicore computing. There isn't necessarily a single "killer app," however, as this provides a large advantage for many vertical or application areas.

AMD is contributing to innovative applications for each vertical by enabling scalable performance into domains such as signal processing, vision/image processing, natural user interfaces, analytics and more. This is largely achieved through parallel programming for special purpose accelerators and General Purpose Graphics Processing Units (GPGPUs.) Machine vision is one such area that can benefit significantly from what OpenCL affords developers: an ability to implement the latest algorithms easily and innovate further depending on the needs of the market and use cases.

AMD is expanding on this value for each vertical by making heterogeneous compute easier to program with fundamental hardware innovations such as Heterogeneous System Architecture (HSA). HSA allows anyone to use mainstream programming languages like C++, OpenMP, Python and Java to harness and benefit heterogeneous compute.

When the ecosystem of compilers and libraries from these mainstream communities adopts HSA, then value will be available to a very large audience of developers. Innovation will kick in, and we will see a proliferation of "killer apps" due to the innovation fueled by specification-compliant platforms from HSA Foundation partners, including AMD.

EECatalog: What myths, or misconceptions, if any, have you had to dispel about OpenCL support at AMD and about how to fully capitalize on OpenCL?

Rao, AMD: In the early days of enablement of OpenCL, there was a misconception that Compute Unified Device Architecture (CUDA) was fundamentally better at enabling performance and ease of use. CUDA started more than two years before OpenCL, but over time, OpenCL drivers and SDK have quickly caught up. CUDA, however, is proprietary. As with other open standards, customers showed a strong preference for OpenCL, which also helped this acceleration.

EECatalog: What is the short- and long-term significance of Khronos positioning OpenVX and OpenCV as complementary, i.e., developers can mix and match APIs and libraries?

Rao, AMD: OpenVX and OpenCV can definitely coexist. OpenVX is a Khronos standard that provides low-level primitives and allows developers to connect them together at runtime for more efficient execution on the hardware.

OpenCV is an open-source library that works across multiple platforms. OpenCV includes more than 2,500 algorithms that span both classic and state-of-the-art vision techniques and image processing functions. OpenCV will continue to evolve as a library of higher-level vision and image processing algorithms that use various APIs as available on the platform, including GPU acceleration via OpenCL. In the future, OpenCV could make OpenVX calls on platforms to make the best use of the platform's capability.

EECatalog: What are the top 3 things developers should know about the new implementation of OpenCV that includes the Transparent API (T-API)?

Rao, AMD: First, with OpenCV, the application developer can implement a single code base and release a single program binary while still targeting the full capabilities of the underlying platform. This is because under OpenCV 3.0, at runtime and based on the platform capabilities, either OpenCL code will be executed or if an OpenCL capable device is not present, CPU-only (C++) code will execute.

Memory transfers are handled automatically by T-API to accommodate both integrated and discrete GPUs depending on the system capabilities. With HSA-compliant platforms, this can be further improved, as there is no need for memory transfer or translation and the shared virtual memory pointer is merely passed through.

The OpenCV implementation is truly cross-platform, particularly due to OpenCL. Since the OpenCL dynamically linked library is discovered at runtime and enables compilation appropriate for the target platform, there is no burden of a bloated binary. The OpenCL executables are typically lightweight, and linking against them is efficient and convenient. This is the power of using an open standard.

EECatalog: What are the top three ways HSA can aid developers of machine vision applications?

Rao, AMD: HSA enables easy access to heterogeneous compute—meaning access to both CPU and GPU resources along with any purpose-built accelerators—via high-level programming languages, especially C++. This access makes it possible to move quickly from innovation at the algorithm level to actual implementation in a product. HSA also enables targeted programming models like OpenCL 2.0 for more advanced developers.

With HSA, you can extract the full performance of the platform, with work being partitioned across CPU and GPU, in the best possible fashion, with features like shared virtual memory (SVM) and device enqueue. HSA applications will obtain faster performance and lower power consumption.

HSA is a standard promoted by the HSA Foundation and supported by multiple companies, giving developers a larger install base of platforms on which these programs will run.

EECatalog: What should embedded designers ask about their own organization's roadmap before selecting the PC-based "smart cameras" and/or conventional DSP and FPGA-based processing platforms?

Rao, AMD:

- Is there any significant advantage to using custom hardware as opposed to using a GPU and multicore-based heterogeneous solution?
- Do we lose competitiveness, efficiency and confidence if we do not invest in software-based innovations moving forward?
- What are the specific advantages we gain if we accelerate innovation using software-based innovations?
- What advantage can we gain if we have a strong ecosystem of partners who can do software, hardware and firmware for us going forward wherever possible, letting us invest our own R&D resources on direct customer value, product differentiation and technology innovation?

EECatalog: What needs to happen across the ecosystem to benefit developers and shorten time-to-market cycles?

Rao, AMD: We need readily available tool kits that support heterogeneous compute that are available in mainstream programming languages like C++; good support in the ecosystem of libraries, services and IP partners for heterogeneous acceleration of workloads; and strong evidence of IP, differentiated products and proof points based on heterogeneous compute.

EECatalog: Are there any issues with regard to embedded developers working on machine vision technology that are being overlooked?

Rao, AMD: Given how high R&D costs are when amortized per unit, reducing the amount of R&D work being done needs more attention rather than just trying to bring down per-unit production costs.

How do we accelerate innovation? That is what makes the "thing" better from the customer experience standpoint. That, too, needs more attention.

EECatalog: As you consider what led to particular machine vision milestones or breakthroughs, are there principles and practices to be recognized that could apply to challenges the industry is experiencing today, and to anticipating future needs?

Rao, AMD: One practice would be to make large amounts of compute available for platforms that are easy to design in and program and are easy to access for innovative engineers—while keeping costs low and yielding performance. Applications such as high-speed inspection cameras, medical imaging/diagnostic solutions and autonomous cars would be accelerated with such access to large amounts of programmable performance.

EECatalog: How would you like to see the Military Internet of Things, the Medical Internet of Things and the Industrial Internet of Things mature? What is the role high-performance machine vision is playing and will play to help these three IoTs mature?

Rao, AMD: I would like to see intelligence move from the server to closer to the edge by enabling intelligence in the gateway. "Thing" does not have to go back to the server for every decision. Gateways can handle certain tasks locally, while the server should handle millions of "things" across the globe.

Much of the content for machine vision is video and pictures, which is an incredible amount of data. Two-thirds of Internet data is video/images, and it's set to be 80 percent in a few years. So intelligence in the gateway would help address this scale-up and make more and better experiences possible—and more reliable.

EECatalog: Will there be a point at which distributing the processing workload across available processor cores in parallel to help improve the real-time performance of the whole system can't be counted on to continue to improve performance—at that point, what are the options, and what are companies doing today to plan on differentiation among a field of other companies already achieving high levels of real-time performance via distributing the processing workload?

Rao, AMD: First of all, there's still plenty of performance left in SoCs, which distribute workloads across different processor cores, because we are only at the beginning of the heterogeneous computing era. This is difficult to harness today but will soon change with HSA. By enabling standard mainstream languages, HSA helps unlock the performance of different types of cores on a platform—especially accelerators and graphics cores—by making it easy to program and maximize the utilization of gigaflops for applications.

Second, even once you achieve this in the future, plenty of places still exist in the overall IoT setup—the "thing," the gateway, other intermediate edge devices, servers, network components—to improve performance. So it's not just about the number of cores but processing of overall data.

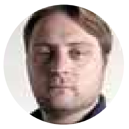
And there's still plenty of room for innovation—not just in compute but also in other IoT-related technologies such as memory subsystems, server technology, sensor technology and the network—all of which will contribute to a better user experience.

Anne Fisher is managing editor of EECatalog.com. Her experience has included opportunities to cover a wide range of embedded solutions in the PICMG ecosystem as well as other technologies. Anne enjoys bringing embedded designers and developers solutions to technology challenges as described by their peers as well as insight and analysis from industry leaders. She can be reached at afisher@extensionmedia.com

ARM Innovations and the Maker Movement: An interview with Dominic Pajak, Embedded Strategist

I sat down with Dominic Pajak, ARM Embedded Strategist, shortly before Christmas after having watched him "wow" an audience on ARM's support of the Maker Movement. A long-time ARM engineer and the original product manager for the company's Cortex®-M0—he's been in the thick of ARM's success in low-power microcontrollers, sensors and radios. In his new strategist role, he's involved with deployment of ARM's IP into all kinds of cool gadgets and use cases. Oftentimes, he concedes, the journey starts with a Maker who cobbles something together and then drives the prototype to success. Edited excerpts follow.
—Chris "C2" Ciufo, editor

By Chris A. Ciufo, Editor-in-Chief, Embedded Systems Engineering



Dominic Pajak, ARM

Chris Ciufo: Dominic, are you having fun?

Dominic Pajak: I've gone from engineering [ARM] products to working with semiconductor partners…all the way up to talking to people using the end devices. And there are some pretty cool projects being done. So I feel I have been on a journey that is pretty amazing.

I was the product manager for the Cortex-M0. I launched the product and then was very heavily involved in the Cortex-M processor family and later moved into our segment marketing group where we're focused in vertical markets where the devices are deployed. As the journey goes, in the first part of my marketing career I was launching these micro-controller type cores and [saw them] getting a great deal of traction in low power microcontrollers, and sensors and radios. Now I am focusing more on where they are getting deployed, which is really really interesting.

C2: I watched your recent YouTube video and wonder—how do you define a Maker?

Pajak: From my point of view (and I don't speak for everyone who considers themselves a Maker), this is really just a "catch all" label for anyone who has a passion or an interest to understand how things around them work and also to create things. They do this either for fun, or for business, or to solve problems in their homes or their communities. It is a really broad term. And it's not exclusively reserved for a hobbyist. Makers are people who are creating enterprises and products out of [bits and parts] and whatever works.

C2: If that's what a Maker is, how do you characterize the Maker Movement? Is it a fad?

Pajak: The way I would define it is this: "embedded" by its nature is a long tail type market, just like the emerging IoT market. [Editor's note: long tail refers to the shift from a few product or market hits, to a huge number of possibly smaller hits stretching out over time. The long tail successes exceed the shorter hits.] There are some very high volume areas within this, such as wearables, smart cities, or building automation in the home, but there is also a long tail of stuff.

The thing about Makers and part of their philosophy is if you rely on other people to make your stuff, then economics of mass production will shape what you get. They often go for the highest volume product, and that is not always going to cover every person's need. And so the way the Maker Movement would view this is that we now have tools to allow us to prototype and produce stuff in smaller batches to solve problems that are particular to me or to my community.

There are definitely businesses that are possible within this because they are addressing part of this long tail that hasn't previously been addressed. Examples are 3D printing and rapid protoyping tools. Commercial Drone. Or the Pebble Watch. Of course sometimes these products do become high volume.

But the Maker Movement also includes STEM education along with hobbyists that are just interested in doing stuff for the fun of learning and maybe solving things around their own home. They're not necessarily entrepreneurs but they would still be classified as Makers. The Maker Movement is very broad, but this is how I see it from my vantage point in ARM.

C2: Can you give me an example of a Maker product that was more than a hobbyist's project?

Pajak: We have a Kickstarter page on the ARM Connected Community that was launched last week, and I had an interview with Pebble's CEO and he talks about how in his dorm room he had this idea for a wearable, a connected watch. Then he prototyped it with an Arduino and he raised over $10 million through Kickstarter. This is an example of someone who has taken the accessible technology and platforms to prototype an idea and then has gotten the backing of a lot of people to go and make this thing a reality. It really gives a good picture of how the Maker Movement can be considered the sharp end of where [some] innovations are coming from.

The democratization of technology means these platforms are not just accessible to electronics engineers. People with other disciplines can also get access to them to prototype with, experiment with, and apply technology to new domains. This is where you are seeing crossovers into wearables—like Pebble—into fashion, or into medicine.

One of the notable Kickstarters you will see is a device called the qPCR DNA diagnosis machine. This project is funded to create low-cost DNA diagnosis machines. Some versions of the machine are based on an ARM Cortex-A8 on a BeagleBone Black. This is an accessible platform that people in biomedical sciences have taken to create something which is very disruptive but has really positive transformative potential to do good for society. This isn't just about pure commercial drivers; there is also a huge human factor.

C2: What is it about ARM and your ecosystem that allows somebody who may not be an engineer, but who may even be an artist, actually create something and achieve such success?

Pajak: We are working to abstract the technology and make it easier, through tools such as mbed™IoT Device Platform. There are several really good reasons for our successes and I'd like to explain. One is about choice and simply the diversity of the different parts the ARM partnership brings to bear. With the Internet of Things (IoT), for example, it isn't a one size fits all thing: we're spanning from swallowable medical devices, wearables and implantables, to pet trackers and connected cars, in all sorts of different shapes and sizes.

C2: How has ARM's low-power reputation aided the Maker Movement?

Pajak: If you take a look at the teardown of any of these leading-edge wearable devices, you will see an ARM-based device inside. People developing these products are pushing back the boundaries of where technology is being applied, and if it is a wearable or a remote IoT device it can be battery powered. That's an essential metric when they are creating these devices and [trying to] really squeeze down the form factor to something that fits seamlessly and unobtrusively into people's lives. The battery is really a key concern, and so ARM's energy-efficient background is well positioned to address pushing the boundaries where this technology has been or is going to be applied.

You see this low-power capability at its most powerful with people from cross disciplines, not just EEs, but also now industrial designers, fashion designers, or product designers. People across many disciplines apply this technology to their domains of expertise and to the problems they see. The diversity that ARM has and the fact that we can offer this low-power capability means they can squeeze [it] into smaller form factors and address the problems very efficiently.

C2: What's on your list of the top 5 benefits that ARM brings to the Maker community?

Pajak: We just talked about one: diversity and choice.

We also discussed energy efficiency through ARM's low-power reputation, and this is of huge importance. Crafting your design to the maximum functionality, longest battery life, and at an optimal price point. That comes back to diversity as well because you need the right choice of parts in order to craft your designs. Another one is accessibility. And so, making [all of this] accessible in terms of cost. There are extremely low-cost ARM development boards available today. One of our mbed boards—I believe from Freescale—is $12.00. But it doesn't end there.

The Raspberry Pi, which is based on ARM11™ Broadcom parts, I hear can be purchased for as low as $25.00 for the Model A+. Raspberry Pi is a fantastic vehicle, and has a very respectable mission in education by giving access to computing at a low cost. You have Arduino, which is extremely famous for bridging the world of electronics and design. And you'll have seen this year they launched the Arduino Zero, which is based on ARM Cortex-M0+. They have the Arduino Due, which is an ARM Cortex-M3. And then the Tre, which is going to be ARM Cortex-A8. So ARM offers accessibility to Makers through lots of different easy-to-use vehicles; people can get hold of ARM technology and innovate around it.

C2: You've mentioned three benefits. Is ARM's position in mobile a big factor?

Pajak: Absolutely. Another key benefit to Makers has to be our position in mobile. That's a springboard to so many things we offer to embedded Makers. ARM is the global leader in these low-power connected devices. We have an extremely well established position in mobile and embedded. Last year alone, the ARM partners shipped 10 billion ARM-based chips. Which is a lot of chips. 3 billion were Cortex-M actually, so we shipped more than that into the overall embedded markets because we span from low-powered Cortex-M and up to Cortex-A which is more suitable for these richer interactive kind of media nodes.

So this industry-proven technology invites some reuse, for certain. Chris Anderson, who is a former writer and Editor-in-Chief of Wired and now is the CEO of 3DR, makes a very interesting point about the economies of scale of mobile. The investment that ARM has made with its partners into efficient computing for mobile has a lot of reuse in the Internet of Things types of markets (such as wearables).

C2: So is there a 5th benefit? Would you maybe include your ecosystem of partners?

Pajak: That is where I was going to go. So, obviously software is vital to these systems. It's one thing to stitch the hardware together into your vision, but people still need an ecosystem of tools and OS providers to span all this stuff. Giving people choice in hardware is fantastic, but having software on a standard architecture makes it incredibly powerful.

If you were to look at the plans around mbed that were announced at TechCon, you will see there's already a very strong ecosystem around this—spanning from sensor providers, communications providers giving Bluetooth, Wi-Fi, cellular, all the way right up to account service providers. And so, it's important for this generation of IoT devices where we know that Makers want their end devices to integrate communications and sensors and they expect to compute very quickly. But we also know they want to connect [the bits] seamlessly, securely and efficiently for the Cloud. This is very much the focus of mbed, and you will see we've got some really fantastic announcements and products coming down the line.

By the way, security is front-and-center of the way they're approaching this and actually [the mbed] approach brings the same class of security you would use for your banking on the Internet down to these constrained devices.

C2: Earlier, you said you were excited about ARM and your job. Why?

Pajak: Certainly the most exciting part to me is that while the typical person on the street doesn't know what a microcontroller is, they may now know what Arduino is. And that is incredibly powerful because suddenly no matter what your discipline might be—artist or designer, for example—you can understand the power of the physical computing that might bring to your particular application area. The other exciting thing is helping educate people on the fact that computing is now so low cost, so small, so potentially "embeddable" in all these different products. This is an interesting point in history actually, where the technology is crossing over to all these disciplines. And ARM is right in the thick of it.

C2: Wearables is one of the first to have emerged from that.

There are other examples. Technology is bringing efficiencies to systems that were probably fixed in the past like garbage collection, making maintenance rounds or the trucks that are going around filling oil tanks. One example I heard was in Minnesota, where the oil delivery truck knows the customer's tank status in advance [remotely] so the route can be optimized to make sure the people who need the oil most get their tanks serviced and are kept warm. With this technology-enabled optimization, it also burns less fuel in the delivery fleet. But there are countless examples of this kind of efficiency and how technology is democratizing the status quo.

C2: What are some of the technologies still needed to keep the Movement going?

Pajak: There's a need for open standards for device communication, and interoperability is necessary to allow this trend to scale. Although not a technology per se, I think education is a big one. And the approaches that are going to be required of the future development of the Internet of Things and these kinds of interactive, interconnected devices is very different in some ways from traditional embedded engineering.

Now you need to be paying more attention to how [your design] is interacting with an Internet-connected system. You need to be thinking more about interaction design with the environment and with people, and it's really the human element that is critical here. We have to think about the value it is bringing businesses and the experiences it's bringing to people. I think more of a rounded perspective is needed, not just from the technical perspective.

It's also essential to be aware of what data is being generated and how can it best be formatted, especially targeting big data analytics. Some other questions that we need to educate system designers and Makers on is: How can we optimize a bigger system? How can we integrate with the existing IT systems that enterprises have? There is a need for a much more broader view when you're connecting these products.

C2: Any last thoughts?

Pajak: Two things. One that is fresh in my mind is last week we launched a curated page on Kickstarter.com, and to me this is amazing because there are 50 independent projects that are are basically start-ups, entrepreneurs. Some of them are small- to medium-size enterprises that have cool projects and they have just chosen to use ARM. They've just evaluated the technology, and the optimal solution is available from ARM partners, so they've based their products on it. Some of these products are absolutely amazing. One is FLUX, which is a 3D printer and scanner. So you put an object in there and it will scan it and then it will print it.

The other thing is how traditional product R&D is being supplemented and sometimes supplanted by a Maker approach. Just look at the amount of VC funding going into hardware startups, especially after Kickstarter campaigns. Crowdfunding has revolutionized the way the product design is happening. It can be far more nimble to have your idea and put it out to a consumer base and immediately get feedback, immediately get buy-in for that product to be funded. It strikes me that this methodology is not just for the startups and the hobbyist. This is something that major companies are looking at very seriously as a means to innovate more quickly.

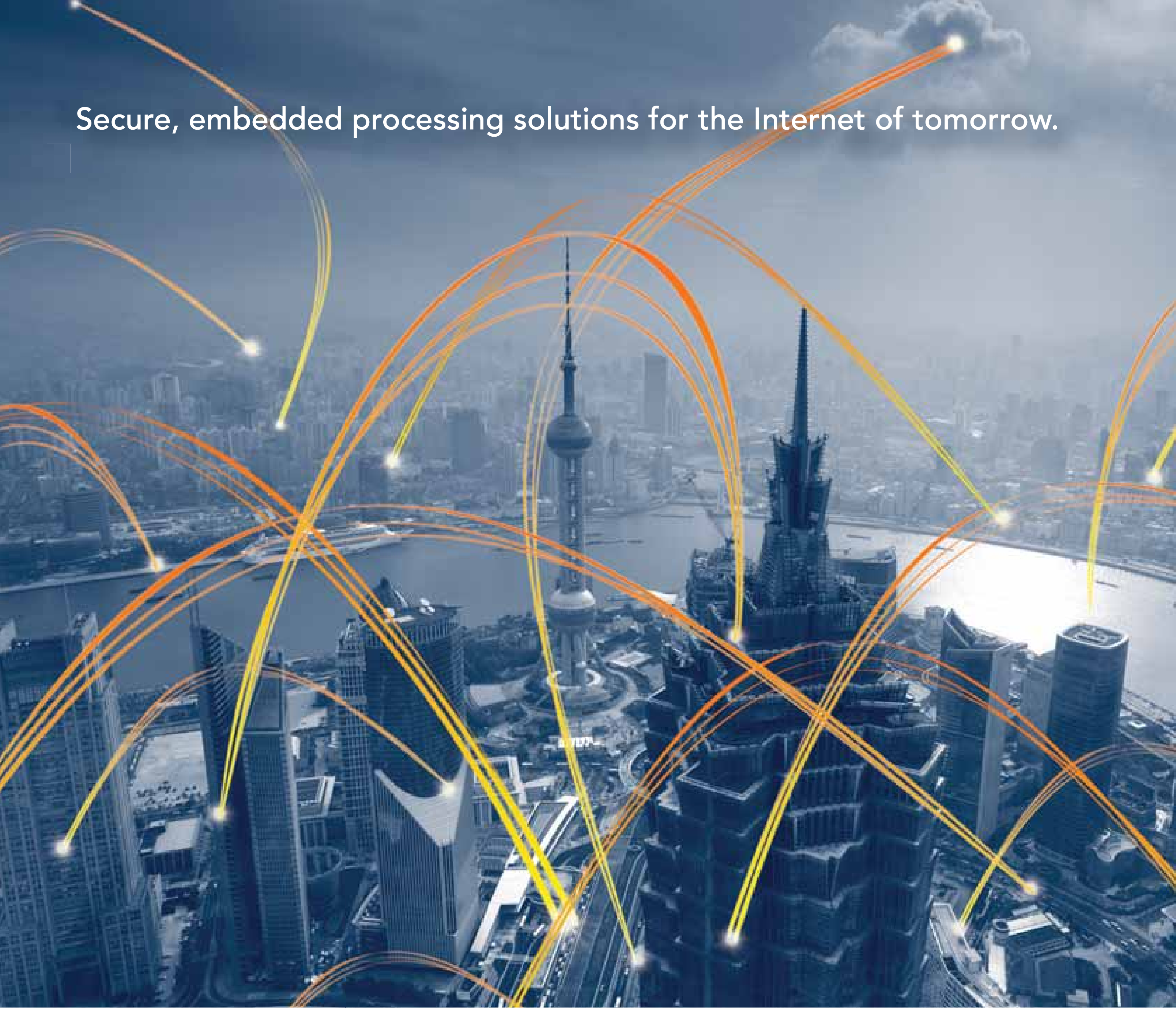
This article was sponsored by ARM.

Two Premier Conferences Showcasing the Embedded Systems Industry

INTERNET OF THINGS
DEVELOPERS CONFERENCE

11th ANNUAL
MULTICORE
DEVELOPERS CONFERENCE

Plan now to attend. **APRIL 27–28, 2016.** Santa Clara, CA USA


Secure, embedded processing solutions for the Internet of tomorrow.

Secure. Scalable. Energy Efficient.

Cars communicating with highways to avoid accidents and traffic congestion. Home hubs talking to the smart grid to reduce energy consumption. Medical monitoring from across town. The billions of intelligent connections that make our world smarter, greener and safer. Freescale's unique product portfolio makes us the only company in the world to provide secure, embedded processing solutions from the edge node through the gateway to the cloud. For more information, visit **freescale.com/IoT.**

freescale™