Always-on Vision Processing Unit for Mobile Applications

Myriad 2 is a multicore, always-on system on chip that supports computational imaging and visual awareness for mobile, wearable, and embedded applications. The vision processing unit incorporates parallelism, instruction set architecture, and microarchitectural features to provide highly sustainable performance efficiency across a range of computational imaging and computer vision applications, including those with low latency requirements on the order of milliseconds.

Brendan Barry
Cormac Brick
Fergal Connor
David Donohoe
David Moloney
Richard Richmond
Martin O'Riordan
Vasile Toma
Movidius

Computer vision is beginning to transition from the laboratory to the real world in a host of applications that require a new approach to supporting the associated power, weight, and space constraints. Current computational platforms and programming paradigms derive from PC-based implementations of computer vision algorithms using conventional CPUs, GPUs, and high-bandwidth DRAMs using programming environments like OpenCV and Matlab. Although it's convenient for prototyping new algorithms, this approach of building a reference implementation on a PC has no direct path to implementation in applications such as mobile phones, tablets, wearable devices, drones, and robots, where cost, power, and thermal dissipation are key concerns. Work has begun in earnest on transitioning such applications from the lab to embedded applications using conventional mobile phone application processors. However, such processors still present issues to the application developer; specifically, the hard real-time requirements imposed by managing low-latency computer vision are very difficult to satisfy on even a capable platform running an operating system such as Android which has not been expressly designed for low latency. Additionally, satisfying the computational requirements of computer vision applications requires almost all of the computational resources on a typical application processor, including a multicore processor, digital signal processor (DSP), hardware accelerators, and GPU, leaving very little over to run virtual reality workloads involving the GPU. Marshalling these diverse resources is possible, but it puts a major strain on both the programmer and the application processor fabric, meaning that corners must be cut in terms of quality while still straining to keep the power below 10 W. Dissipating 10 W in a drone might not be a thermal issue, but it leaves less power available for flying, whereas in a mobile phone 3 to 4 W can be dissipated on average before the phone becomes literally too hot to handle. Clearly, across all of the possible applications a better solution is

Published by the IEEE Computer Society

required to evolve to truly capable computer vision systems with intelligent local decision making for autonomous robots and drones as well as truly immersive virtual reality experiences. The vision processing unit (VPU) solution outlined in our Hot Chips presentation[1] and expanded upon here uses a combination of low-power very long instruction word (VLIW) processors with vector and SIMD operations capable of very high parallelism in a clock cycle, allied with hardware acceleration for key image processing and computer vision kernels, backed by a very high bandwidth on-chip multicore memory subsystem. The Myriad2 VPU aims to provide an order of magnitude higher performance efficiency, allowing high-performance computer vision systems with very low latency to be built while dissipating less than 1 W.

Always-on computer vision

The possibility of computationally fusing images has existed in the minds of scientists like Lippmann[2] since the early 1900s, yet it has had to wait for Moore's law to catch up with the enormous computational and power efficiency requirements to be realized, especially, in mobile devices. Computational imaging and visual awareness applications are shifting the imaging paradigm in mobile devices from taking pictures (and video) to making them, replacing complex optics with simplified lens assemblies and multiple apertures; combining images captured by heterogeneous sensors including RGB (red, green, blue), infrared (IR), and depth; and extracting scene metadata from still images and video streams. This revolution in image and video capture is enabling a broad range of new application areas and use cases that were previously limited to desktops and workstations rather than battery-powered mobile devices, including mobile handsets, tablets, wearables, and personal robotics. These devices support a broad range of computer vision use cases including multiaperture (array) cameras, high dynamic range photography and video, multispectral imaging (for example, IR plus visible spectrum and depth sensors working in concert), gesture or facial recognition and analytics, virtual reality gaming, visual search, simultaneous localization and mapping for robotics, and many others.

Always-on computer vision has the potential to greatly enhance user experiences; however, these applications are in a constant state of flux, precluding the use of completely fixed-function hardware because the algorithms are subject to change. The current crop of heterogeneous platforms, such as the mobile phone application processors shown in Figure 1, are often used to prototype algorithms and appear limited to VGA 30 frames-per-second (fps) resolution and 6 to 8 W power dissipation (http://elinux.org/Jetson/Computer_Vision_Performance). Despite the impressive computational resources available in current application processors, implementing even relatively simple pipelines involves a high level of complexity in marshalling heterogeneous resources. A typical application processor contains multiple ARM processors with NEON SIMD extensions and other DSPs and GPUs, as shown in Figure 1, all of which must be coordinated and must allow collaborative access to data structures in memory through shared access or copied data. Indeed, the latency of GPU access can rule out its usage for some applications, such as low-latency virtual reality gaming. In such architectures, the power overhead of explicitly copying blocks of data (memcopy) can be on the order of 1 W, and the latency in moving frames to and from the GPU subsystem can be on the order of 5 to 10 ms with target frame rates that can be in excess of 60 fps.

A typical example of this new breed of computational imaging devices and applications is Google's Project Tango (www.google.com/atap/projecttango), which uses two Movidius Myriad 1 computational vision processors.[3] Indeed, the next generation of computational vision requirements for array cameras, complex vision pipelines, and image signal processing pipelines for new sensor types such as RGB near-infrared, RGBC (red, green, blue, clear), and sensor fusion for heterogeneous sensors in devices such as Project Tango will demand exceptional levels of processing backed by high memory bandwidths for sustainable performance within a global 1-W power envelope to enable always-on user experiences (see Figure 2).

In this context, Moore's law (Dennard planar 2D scaling) is slowing.[4] Long before

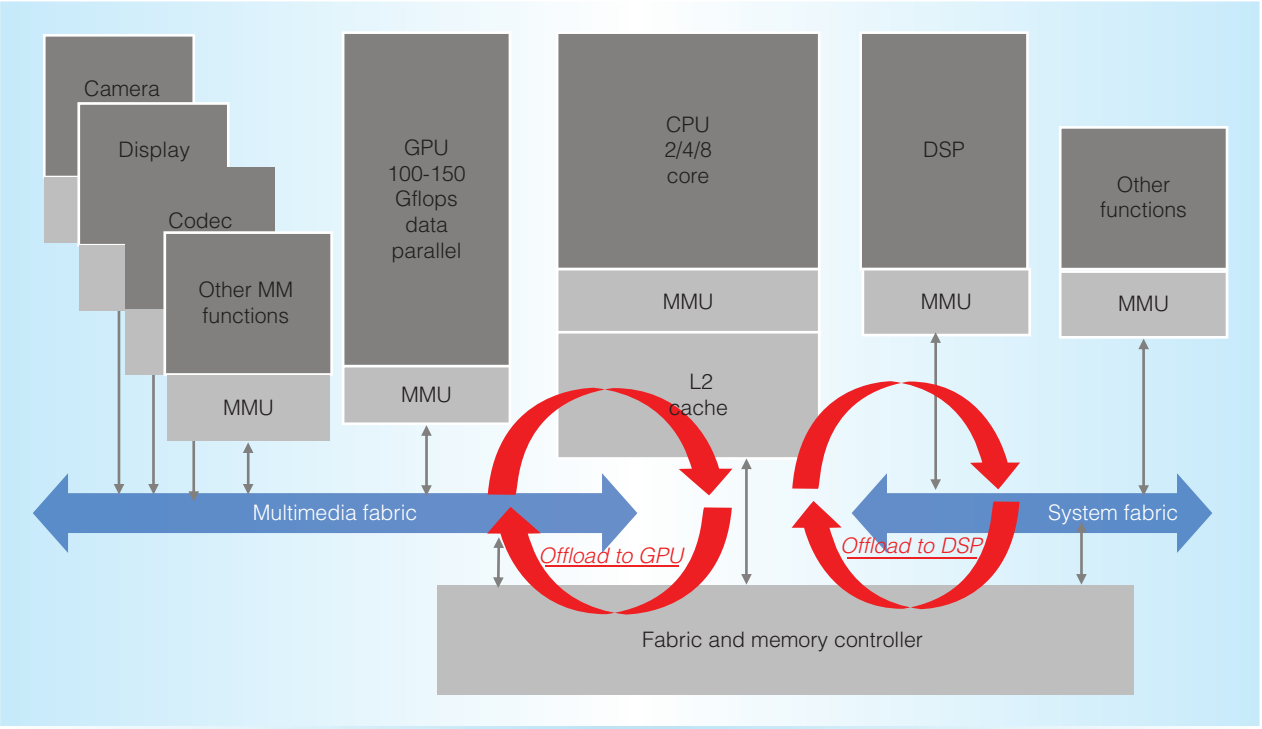

Figure 1. The architecture of a typical application processor contains multiple subsystems of dedicated ISP hardware, digital signal processors (DSPs), multicore processors, and a GPU subsystem that usually has its own private memory spaces between which data must be moved under program control or direct memory access hardware.

we reach the purported 5-nm limit around 2020, the already high wafer prices will rule out custom silicon solutions for all but the highest-volume applications; this means that solutions will have to be highly programmable to be economically viable. Horowitz identifies power efficiency as a major issue,[5] and wirelessly transmitting data for remote computation can cost up to a million times more energy per operation compared to processing locally in a device. Coupled with this, the power required for cloud-based video applications scales up to 1,000 times worse than the power consumed by local in-device computation,[6] meaning that large amounts of new infrastructure would have to be built if computation is not pushed out to the network edge and ideally right beside the image sensor. Another reason for pushing processing and even limited decision making to the network edge in applications such as virtual or augmented reality, autonomous vehicles, and robots is that datacenters have major latency issues. Privacy is clearly a major issue if video or still images are being uploaded to the cloud for processing, and, again, processing at the network edge and exchanging metadata can help mitigate privacy concerns.

The industry has an opportunity to distribute computation and build systems that can think locally and act globally by preprocessing data within personal devices using low-power processors. In this model, data is processed as close as possible to the sensor, in the same way as our human brains, eyes, and ears are located in close proximity. Only metadata rather than video is streamed to the cloud, resolving the power-efficiency, latency, and security issues and requiring radically less expensive infrastructure. The key challenge, of course, is programming such systems in a highly portable and performance- and power-efficient way.

Vision processor design challenges

While clearly interesting from a research and commercial point of view, computational imaging presents a series of fundamental

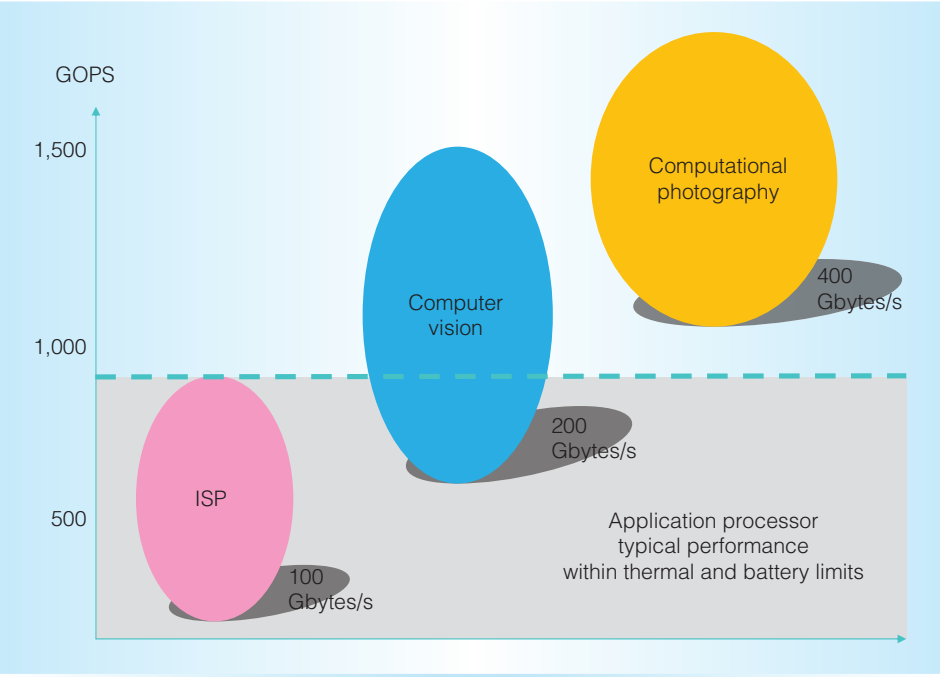

Figure 2. The requirements of always-on computer vision applications extend to all parts of the image and video-processing chain, from novel and heterogeneous sensors to array cameras to dedicated image and video processing acceleration to vector and SMID processors and high-level APIs that let programmers extract performance from the underlying hardware and firmware.

processing challenges: extreme computational complexity (hundreds to thousands of Gflops), a requirement for sustained performance, and high sensitivity to latency. Although these concerns are all valid, computational imaging has not been a reality in mobile devices until recently, primarily because of two power consumption issues: the heat dissipation potential in small form factor mobile phones, and the energy density of available battery technologies. Current mobile processing architectures (such as CPUs and GPUs) can't deal efficiently with computational imaging and computer vision workloads under these constraints, and the future does not bode well as Moore's law is slowing down, causing the power and performance benefits in the transition to the next node to decrease. In our opinion, the next decade will mark the era of special-purpose processors focused on decreasing the energy per operation in a new way.

The general design principles followed in the first-generation Myriad 1 design and extended in Myriad 2 are based on Agarwal's observation that beyond a certain frequency limit for any particular design and target process technology, the designer starts to pay quadratically in power for linear increases in operating frequency. Thus, increased parallelism is preferred in terms of maximizing throughput per milliwatt per megahertz. The objective in the Myriad 2 design in 28-nm process technology was to achieve 20 to 30× the processing per watt of that achieved in 65 nm by the previous generation Myriad 1.[3] A 5× performance increase compared to Myriad 1 was achieved by increasing the number of Streaming Hybrid Architecture Vector Engine (SHAVE) vector processors from 8 to 12 and increasing the clock rate from 180 to 600 MHz. The remaining 15 to 25× performance increase was achieved by including 20 hardware accelerators in Myriad, two of which can output one fully computed output pixel per cycle. The hardware accelerator rationale fits with the Moore's law trend below 28 nm, because memory access is expensive in terms of

power, and memory is now scaling at 20 to 30 percent (down from 50) versus 50 percent for logic below the 28-nm node.[7] Generally, this means that the chip architect must look at trading arithmetic operations per memory access for lower memory storage and bandwidth requirements, which also has the benefit of globally lower power.[8]

Myriad 2 system architecture

As Figure 2 shows, application processor systems on chip (SoCs) are typically based on one or more 32-bit reduced-instruction-set computing (RISC) processor surrounded by hardware accelerators that share a common multilevel cache and DRAM interface. This shared infrastructure is attractive from a cost perspective but creates major bottlenecks in terms of memory access, where multimedia applications demand real-time performance but must contend with user applications and a platform OS such as Android. Thus, the platform often underdelivers in terms of computational video performance or power efficiency. A more attractive model is to build a software programmable architecture as a coprocessor to an application processor, rather than using fixed-function or configurable hardware.[1] Such a coprocessor can then abstract away a lot of the hard real-time requirements in dealing with multiple camera sensors and accelerometers from the application processor and making the entire ensemble appear as a single super-intelligent MIPI camera below the Android camera's hardware abstraction layer. As a result, the architecture presented in this article focuses on power-efficient operation, as well as area efficiency, allowing product derivatives to be implemented entirely in software where previously hardware and mask costs would have been incurred. The software programmable model is especially interesting in areas where standards do not exist, such as body-pose estimation.[9]

As a VPU SoC, Myriad 2 has a software-controlled, multicore, multiported memory subsystem and caches that can be configured to allow handling of a large range of workloads and provide high, sustainable on-chip data and instruction bandwidth to support the 12 processors, two RISC processors, and high-performance video hardware accelerator filters. A multichannel (or multiagent) direct memory access engine offloads data movement between the processors, hardware accelerators, and memory, and a large range of peripherals including cameras, LCD panels, and mass storage, communicate with processors and hardware accelerators. Additional programmable hardware acceleration helps speed up hard-to-parallelize functions required by video codecs, such as H.264 CABAC, VP9, and SVC, as well as video processing kernels for always-on computer-vision applications. Up to 12 independent high-definition cameras can be connected to 12 programmable MIPI D-PHY lanes supporting CSI-2 organized in six pairs, each of which can be independently clocked, to provide an aggregate bandwidth of 18 Gbits per second. The device also contains a USB 3.0 interface and a gigabit Ethernet media access controller, as well as various interfaces such as Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I^2C), and Universal Asynchronous Receiver Transmitter (UART), connected to a reduced number of I/O pins using a tree of software-controlled multiplexers. In this way, these interfaces support a broad range of use cases in a low-cost plastic ball-grid array package with integrated 2 to 4 Gbit low-power DDR2/3 synchronous DRAM stacked in package using a combination of flip-chip bumping for the VPU die and wire bonding for the stacked DRAM.

Because power efficiency is paramount, the device employs a total of 17 power islands, including one for each of the 12 integrated SHAVE processors, allowing very fine-grained power control in software. The device supports 8-, 16-, 32-, and some 64-bit integer operations as well as fp16 (OpenEXR) and fp32 arithmetic, and it can aggregate 1,000 Gflops (fp16). The resulting architecture offers increased performance per watt across a broad spectrum of computer vision and computational imaging applications from augmented reality[10] to simultaneous localization and mapping.[11]

SHAVE processor microarchitecture

To guarantee sustained high performance and minimize power use, the Movidius

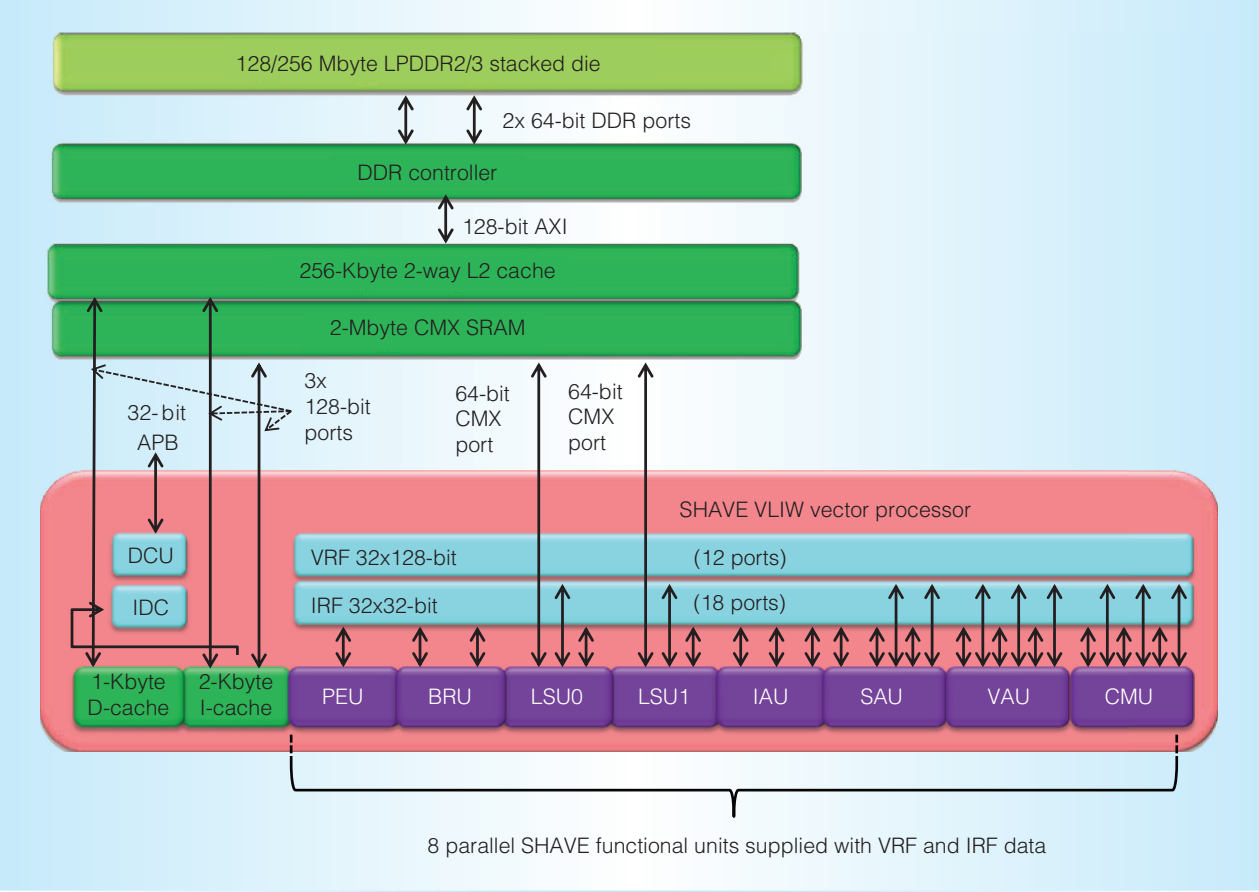

Figure 3. The Streaming Hybrid Architecture Vector Engine (SHAVE) processor microarchitecture is an eight-slot very long instruction word processor capable of performing multiple 128-bit vector operations in parallel with multiple load/store, scalar floating-point, integer, and control-flow operations in a single clock cycle.

SHAVE processor contains wide and deep register files coupled with a variable-length long instruction word (VLLIW) controlling multiple functional units including extensive SIMD capability for high parallelism and throughput at both a functional unit and processor level. The SHAVE processor is a hybrid stream processor architecture combining the best features of GPUs, DSPs, and RISC with both 8-, 16-, and 32-bit integer and 16- and 32-bit floating-point arithmetic as well as unique features such as hardware support for sparse data structures. The architecture maximizes performance per watt while maintaining ease of programmability, especially in terms of support for design and porting of multicore software applications.

As Figure 3 shows, VLIW packets control multiple functional units that have SIMD capability for high parallelism and throughput at a functional unit and processor level. The functional units are the predicated execution unit (PEU), the branch and repeat unit, two 64-bit load-store units (LSU0 and LSU1), the 128-bit vector arithmetic unit (VAU), the 32-bit scalar arithmetic unit, the 32-bit integer arithmetic unit, and the 128-bit compare move unit (CMU), each of which is enabled separately by a header in the variable-length instruction. The functional units are fed with operands from a 128-bit × 32-entry vector register file with 12 ports and a 32-bit × 32-entry general register file with 18 ports delivering an aggregate 1,900 Gbytes per second (GBps) of SHAVE register bandwidth across all 12 SHAVEs at 600 MHz. The additional blocks in the diagram are the instruction decode and the debug control unit. A

constant instruction fetch width of 128 bits for variable-length instructions—with an average instruction width of around 80 bits—packed contiguously in memory and the five-entry instruction prefetch buffer guarantee that at least one instruction is always ready while taking account of branches.

Computer-vision hardware acceleration

On Myriad 1, SHAVE implementations of software filters offered, on average, in the range of 1.5 to tens of cycles per pixel (the fastest kernel was downscaling Gaussian 5x1 at 0.675 cycles/pixel). With Myriad 2, designers made a concerted effort to profile the key performance-limiting kernels and define hardware accelerators that could be flexibly combined into hybrid hardware/software pipelines for computer vision and advanced imaging applications. These hardware accelerators support context switching in order to allow the processing of multiple parallel camera streams. For Myriad 2, the global performance target was to achieve 20 to 30× the performance of the previous generation Myriad 1 VPU.

The programmable hardware accelerators implemented in the Myriad 2 VPU include a poly-phase resizer, lens shading correction, Harris corner detector, Histogram of Oriented Gradients edge operator, convolution filter, sharpening filter, gamma correction, tone mapping, and luminance and chrominance denoising. Each accelerator has multiple memory ports to support the memory requirements and local decoupling buffers to minimize instantaneous bandwidth to and from the 2-Mbyte multicore memory subsystem. A local pipeline controller in each filter manages the read and writeback of results to the memory subsystem. The filters are connected to the multicore memory subsystem via a crossbar, and each filter can output one fully computed pixel per cycle for the input data, resulting in an aggregate throughput of 600 Mpixels per second at 600 MHz.

Multicore memory subsystem

The ability to combine image processing and computer vision processing into pipelines consisting of hardware and software elements was a key requirement because current application processors are limited in this respect. Thus, sharing data flexibly between SHAVE processors and hardware accelerators via the multiported memory subsystem was a key challenge in the design of the Myriad 2 VPU. In the 28-nm Myriad 2 VPU, 12 SHAVE processors, hardware accelerators, shared data, and SHAVE instructions reside in a shared 2-Mbyte memory block called Connection Matrix (CMX) memory, which can be configured to accommodate different instruction and data mixes depending on the workload. The CMX block comprises 16 blocks of 128 Kbytes, which in turn comprise four 32-Kbyte RAM instances organized as 4,096 words of 64 bits each, which are independently arbitrated, allowing each RAM block in the memory subsystem to be accessed independently. The 12 SHAVEs acting together can move (theoretical maximum) 12 × 128 bits of code and 24 × 64 bits of data, for an aggregate CMX memory bandwidth of 3,072 bits per cycle (1,536 bits of data). This software-controlled multicore memory subsystem and caches can be configured to allow many workloads to be handled, providing high sustainable on-chip bandwidth with a peak bandwidth of 307 GBps (sixty-four 64-bit ports operating at 600 MHz) to support data and instruction supply to the 12 SHAVE processors and hardware accelerators (see Figure 4). Furthermore, the CMX subsystem supports multiple traffic classes from latency-tolerant hardware accelerators to latency-intolerant SHAVE vector processors, allowing construction of arbitrary pipelines from a mix of software running on SHAVEs and hardware accelerators, which can operate at high, sustained rates on multiple streams simultaneously without performance loss and at ultra-low-power levels.

Myriad 2 implementation

Figure 5 shows a die plot of the 27 mm^2 device, highlighting the major functional blocks. Because power efficiency is paramount in mobile applications, Myriad 2 provides extensive clock and functional unit gating and support for dynamic clock and voltage scaling for dynamic power reduction. It also contains 17 power islands—one for each of the 12 SHAVE processors; one for

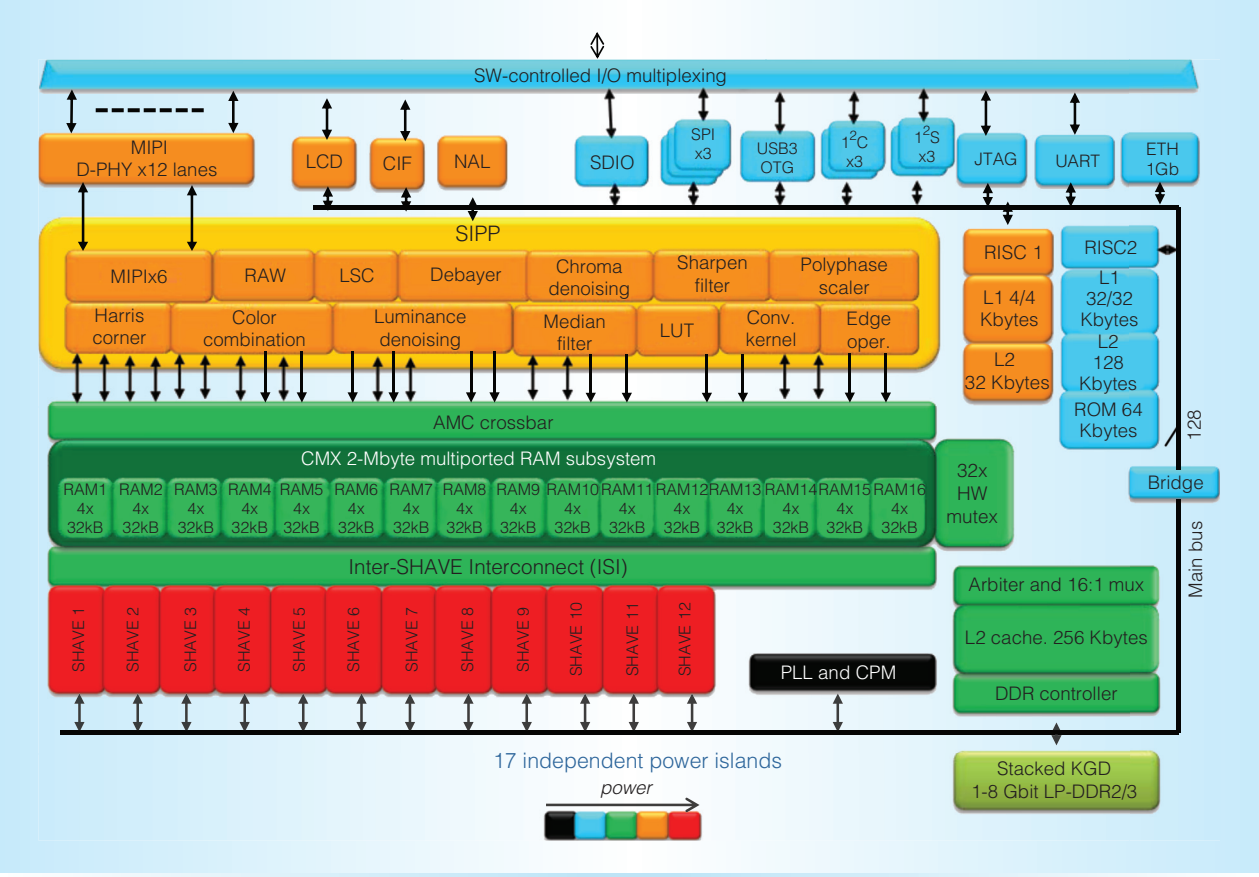

Figure 4. Myriad 2 vision processing unit (VPU) system on chip (SoC) block diagram shows the 12 SHAVE cores and associated Inter-SHAVE Interconnect, located below the multicore memory subsystem (connection matrix or CMX). Above the CMX are the hardware accelerators for computer vision and image processing controlled by a first reduced-instruction-set computing (RISC) processor, and above that again are the I/O peripherals, which are controlled by a second RISC processor. Finally, all of the processors in the system share access to the 64-bit DRAM interface.

the CMX memory subsystem; one for the media subsystem, including video hardware accelerators and RISC1; one for RISC2 and peripherals; one for the clock and power management; and one always-on domain. This allows fine-grained power control in software with minimal latency to return to normal operating mode, including maintenance of the static RAM (SRAM) state that eliminates the need to reboot from external storage. The device has been designed to operate at 0.9 V for 600 MHz.

Myriad 2 applications

The SHAVE DSP supports streaming workloads from the ground up, making decisions about pixels or groups of pixels as they are processed by the 128-bit VAU. Because 128-bit comparison using the CMU and predication using the PEU can be performed in parallel with the VAU, higher performance can be achieved compared to a GPU in which decisions must be made about streaming data because GPUs suffer from performance loss due to branch divergence. For example, the SHAVE processor excels when processing the FAST9 algorithm, where 25 pixels on a Bresenham circle around the center pixel must be evaluated for each pixel in, for instance, a 1080p frame. The FAST9 algorithm looks for nine contiguous pixels out of 25 on a Bresenham circle around the center that are above or below a center-pixel luminance value, meaning hundreds of operations must be computed for each pixel in a

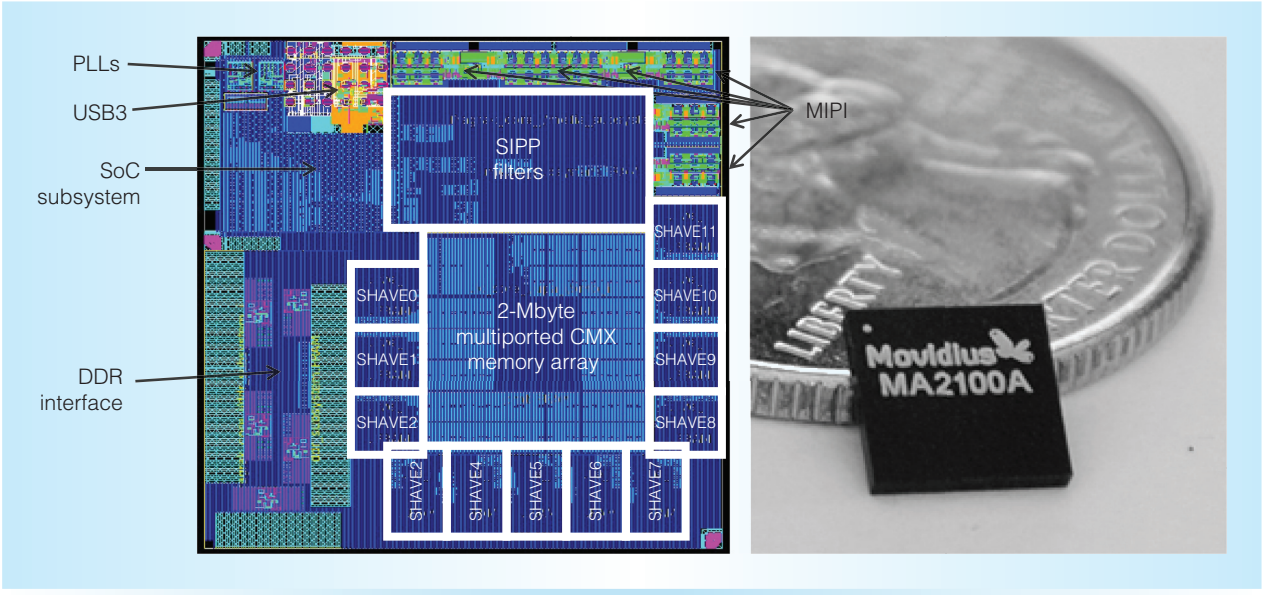

Figure 5. Myriad 2 VPU SoC die plot and 6 × 6 mm ball-grid array packaging. In the die plot, the SHAVE processors and hardware accelerators are clustered around the CMX memory system. On the outer rim of the die, the DRAM interface and MIPI, USB, and other peripherals as well as phase-locked loops dominate the periphery of the die, with the remainder of the die being occupied by the RISC processor and peripheral subsystems.

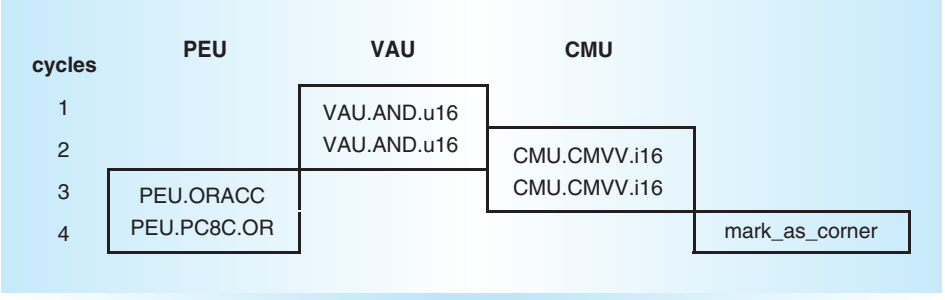

Figure 6. SHAVE very long instruction word mapping of GPU-Optimized FAST (kernel) to microarchitecture. Here, the parallelism using the predicated execution unit, vector arithmetic unit, and compare move unit allows eight of the 16 possible rotations about the center pixel to be evaluated in a single cycle, a complex sequence of operations that would require tens of cycles on a conventional machine.

high-definition image. This requires hundreds of instructions on a scalar processor, and performance optimization requires the use of machine learning and training to improve detector performance.[12]

GPU-Optimized FAST (GO-FAST) is an adaptation of FAST for SHAVE, making use of decomposition and code optimization to produce identical outputs efficiently on stream processors. The SHAVE assembler code shown in Figure 6 illustrates the operations required to find a contiguous segment of nine pixels on a circle surrounding a center pixel. On SHAVE, conversely, the massive parallelism allows the core of the FAST9 algorithm to be implemented in three lines of a VLIW assembler using a brute-force implementation that evaluates 8× rotations per VLIW packet using three VLLIW slots (PEU, VAU, and CMU). The result is that a single SHAVE processor running at less than 200 MHz can implement the FAST9 detector operating on 1080p input frames at over 60 Hz and detecting hundreds of feature points.

The following list shows the code implementing the $528(16 + 4 \times 128)$ equivalent

operations, resulting in a performance efficiency of 1,188 GOPS/W (16-bit), and using all 12 SHAVEs at the maximum 600-MHz clock rate, under typical process, voltage, and temperature conditions for a total power dissipation of 800 mW, including peripherals and stacked DRAM:

- PEU: 8.
- VAU: 128.
- CMU: 128.
- Operations/corner: 528.
- Operations/cycle: 132.
- MHz: 600.
- SHAVEs: 12.
- mW: 800.
- GOPS/W Myriad2: 1,188.

As we previously mentioned, applications such as augmented reality[10] require low latency. To this end, the Myriad 2 VPU has hard real-time support for low-latency computer vision, which is essential for line-sync-based, super-low-latency processing. This is enabled by deterministic data access due to 2 Mbytes of on-chip SRAM and the dedicated stacked DDR, which is dedicated to computer vision tasks without any CPU contention—as is typically seen in application processors with multiple ARM cores running Android in contention with DSPs and GPUs. Finally, the VPU has low-latency (less than 2 μs worst-case) interrupt handling with no contention with peripherals such
as USB or SPI, and 64-bit timestamp support. As a result, an OpenCV-compatible, multiscale Haar Cascade with 20 stages, computed using 12 SHAVEs and one hardware accelerator, can calculate 50,000 multiscale classifications for each 1080p resolution frame, in less than 7 ms.

The Myriad2 microarchitecture presented here is a first generation of a true VPU, which we believe will become as synonymous with vision processing as a GPU is with graphics processing. The future challenges in terms of evolving the VPU concept will be in achieving yet higher processing efficiency per milliwatt. Improving absolute efficiency will rely on improved analysis and microarchitectural enhancements to reduce power dissipated per clock cycle. Larger gains in processing efficiency may come through using hardware-assisted event filtering in the always-on power domain to detect if the inertial measurement unit or other events occur and to power up the Myriad hardware accelerators and SHAVE processors only where it is highly likely that there is useful visual data to process. This will reduce the average power dissipation for many interesting use cases in wearable cameras, phones and tablets, and even small drones or robots. We also expect future Myriad variants to provide enhanced support for visual object detection and classification and support for path finding and artificial intelligence by providing enhanced hardware and software support for deep learning and convolutional neural networks. This will allow local low-latency decision making to be made on the basis of the output of the vision processing pipeline, which is essential for the safe operation of autonomous vehicles, robots, and drones that interact closely with humans. MICRO

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 611183 (EXCESS Project, www.excess-project.eu).

References

1. D. Moloney et al., "Myriad 2: 'Eye of the Computational Vision Storm,'" Hot Chips 26, 2014.
2. "Integral Photography: A New Discovery by Professor Lippmann," *Scientific American*, 19 Aug. 1911, p. 164.
3. D. Moloney, "1TOPS/W Software Programmable Media Processor," Hot Chips 23, 2011.
4. A. Lal Shimpi, "Intel: By 2020 the Size of Meaningful Compute Approaches Zero," *AnandTech*, 10 Sept. 2012; www.anandtech.com/show/6253/intel-by-2020-the-size-of-meaningful-compute-approaches-zero.
5. J. Lipsky, "ISSCC Keynote: No Silicon, Software Silos," *EE Times*, 5 Sept. 2014; www.eetimes.com/document.asp?doc_id=1320954.
6. R. Merritt, "Google Cloud Needs Lower Latency," *EE Times*, 10 Feb. 2014; www.eetimes.com/document.asp?doc_id=1320945.

7. Z. Or-Bach, "28nm—The Last Node of Moore's Law," *EE Times*, 19 Mar. 2014; www.eetimes.com/author.asp?doc_id=1321536.
8. D. Bates et al., "Exploiting Tightly-Coupled Cores," *J. Signal Processing Systems*, Aug. 2014, doi: 10.1007/s11265-014-0944–6.
9. J. Shotton et al., "Real-Time Human Pose Recognition in Parts from Single Depth Images," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2011, pp. 1297–1304.
10. J. Toledo et al., "Augmented Reality," *Design of Embedded Augmented Reality Systems*, S. Maad, ed., InTech, 2010.
11. D. Scaramuzza and F. Fraundorfer, "Visual Odometry: Part I—The First 30 Years and Fundamentals," *IEEE Robotics and Automation Mag.*, vol. 18, no. 4, 2011, pp. 80–92.
12. E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," *Proc. European Conf. Computer Vision*, 2006, pp. 430–44.

Brendan Barry is vice president of engineering at Movidius, where he leads development of hardware and design tools and contributes to the Myriad and SHAVE microarchitecture and physical design. His research interests include hardware computer-vision acceleration, processor, and SoC architecture. Barry has a BEng in electronic engineering from Dublin City University. Contact him at brendan.barry@movidius.com.

Cormac Brick is vice president of software engineering at Movidius, where he works on integrating Movidius Myriad technology to enable new computer vision and imaging products. His research interests include computer vision, sensor fusion, and artificial intelligence. Brick has a BEng in electronic engineering from University College Cork. Contact him at cormac.brick@movidius.com.

Fergal Connor is a staff engineer at Movidius, where he is working on the next generation of the SHAVE processor. His research interests include power-efficient processor architectures and multicore processor design. Connor has an MEng in electronic engineering from Dublin City University. Contact him at fergal.connor@movidius.com.

David Donohoe is an embedded software architect at Movidius, where he leads the research, development, and software specification activities around high-throughput pixel-processing pipelines, targeting embedded and mobile applications. Donohoe has a BEng in electronic engineering from Dublin City University. Contact him at david.donohoe@movidius.com.

David Moloney is the chief technology officer of Movidius. His research interests include processor architecture, computer vision algorithms and hardware acceleration and systems, hardware and multiprocessor design for DSP communications, high-performance computing, and multimedia applications. Moloney has a PhD in mechanical engineering on high-performance computer architecture from Trinity College Dublin. Contact him at david.moloney@movidius.com.

Richard Richmond is a staff engineer at Movidius. His research interests include low-power hardware acceleration of computationally intensive image signal processing and computer vision algorithms. Richmond has an MEng in electronic engineering from Edinburgh University. Contact him at richard.richmond@movidius.com.

Martin O'Riordan is the chief compiler architect at Movidius, where he leads the SHAVE processor compiler development based on LLVM infrastructure and also worked on the SHAVE instruction set architecture and Myriad microarchitecture. O'Riordan has a BSc in computer science from Trinity College Dublin. Contact him at martin.oriordan@movidius.com.

Vasile Toma is a design engineer at Movidius. His research interests include chip-card security, SoC and processor architecture, DSP, and computer vision. Toma has a BEng electronic engineering from the Politehnica University of Timisoara, Romania. Contact him at vasile.toma@movidius.com.