Tackling Defense Apps by Harnessing the Intel Core i7's Integrated GPU

Read About

Graphics Processing Units (GPUs)

General Purpose Graphics Processing Units (GPGPUs)

Display Rendering

Math Acceleration

Floating Point Processing

Introduction

Defense electronics has embraced COTS technology. Riding waves of innovation driven by huge, worldwide markets, defense system designers are using multi-core processors, high capacity memory chips, wide bandwidth fabrics and open software – all developed for commercial markets. These designers must constantly evaluate new commercial innovations and determine if, and how, they can be employed to craft a superior defense solution.

This white paper will discuss a recent innovation, the integrated, on-chip Graphics Processing Units (GPUs) within a new generation of Intel® x86 architecture chips. The paper will cover the basic concepts of how these integrated GPUs function, look at what they do well, and not so well, and then present some Defense electronics use cases where they offer significant advantages.

Evaluating the design value of an integrated GPU

Early in the design cycle, system engineers must make decisions on what types of processing elements will best meet their requirements. There are lots of choices – Power Architecture®, ARM®, x86 architecture, FPGAs, GPUs – all with strengths and weaknesses.

Now there is a new variant in the list of processor choice, GPUs integrated inside Core i7 x86 silicon. Understanding that variant, its strengths and weaknesses, is one part of making good design decisions for a new generation of programs.

The Primary Functions of a GPU

A discussion on the value of an integrated GPU needs to begin with an understanding of how GPUs function, as a general class of processors. At a high level, GPUs lend themselves to two broad classes of processing, both of which are needed by defense applications:

(1) Rendering images for display

(2) Accelerated floating point math operations

Info

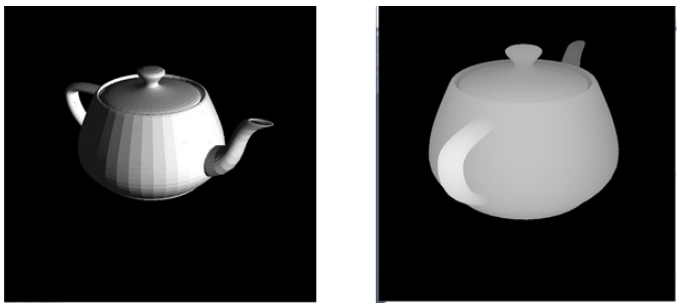
curtisswrightds.com

Email

ds@curtisswright.com

What is involved in rendering an image?

An image on a flat screen monitor can be derived via a 2D or 3D computer model. Creation of complex objects usually involves the utilization of very simple shapes, such as triangles, that serve as fundamental graphic building blocks. Then these objects are immersed in a scene that defines the overall lighting, surface textures, and point of view. One of the most popular techniques for immersing the objects within a given scene involves first object shading and then rasterization, which geometrically projects the 3-D objects and the scene onto a 2-D image plane, i.e. the monitor.



Greater shading capability enhances image quality

In contrast to these computer-generated images, Image Signal Processing, as well as Computer Vision, receives a continuous stream of imaging data from a video camera or electro-optic focal plane array. Two dimensional signal processing techniques are then applied for edge detection and motion blur correction, in addition to a myriad of other application specific algorithms. The enhanced image is then rendered to a display, where the human operator will hopefully have an easier experience in regard to extracting useful information.

How does a GPU make that happen?

Rendering images was the original intent of the Graphics Processing Unit (GPU) chip and remains as its core function in commercial applications to this day. Projecting 3D computer images in motion onto a 2D surface, so as to be convincing to the human eye, is a commonly used but highly compute intensive application. A moving image, for instance, will command a tremendous number of calculations per frame as the GPU renders simple polygons to create more advanced objects, maps textures to simulate surfaces, and then rotates these shapes within dynamically changing coordinate systems.

Image Signal Processing and Computer Vision perform different operations, but are equally intensive in terms of computation throughput requirements and use math algorithms highly similar to those used by image rendering.

Why is the GPU good at these operations?

Given the continuously streaming nature of rendering video frames, GPUs are by nature optimized for streaming processing. To manage streams of data they first boost throughput by parallelizing their compute engines via a massive pipeline architecture. Then they add high bandwidth, high capacity memory, which is central to this model given the relentless flow of high resolution imagery.

In recent years, this streaming processing capability has been further enhanced by the incorporation of “programmable shaders”. These shaders perform highly sophisticated shading effects (e.g. volumetric lighting, normal mapping, and chroma keying); they are optimized at the silicon level to execute the algorithms associated with this type of processing. Powerful new GPUs include large numbers of these shaders for accelerated rendering of highly complex, life-like images.

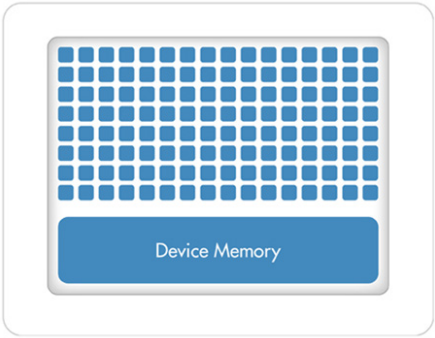
How, and why, does a GPU accelerate floating point math?

With the advent of programmable shaders that are inherently adept at vector and matrix math, using GPUs to work on problems outside of rendering graphics has become a reality. Vector and matrix data manipulation are fundamental to linear algebra, filtering, and transforms, all components of digital signal processing. Initially, the data formats were integer based, however the need for high dynamic range imaging saw the development of floating point data formats. This set the course in motion for GPUs to be used as a floating point math accelerators in a multitude of commercial and defense related high performance compute applications. The term General Purpose GPU or GPGPU was instantiated.

Why are GPUs really good at floating point math for streaming applications?

The sheer number of shaders in GPGPUs, coupled with a high throughput streaming pipeline architecture, allows for scalable, massive parallel processing that can far outpace the floating point performance of SIMD units within standard CPUs. Discrete GPGPUs are often coupled with high throughput, high capacity GDDR5, a type of synchronous

graphics RAM based on DDR3 SDRAM. This is not only required for buffering high frame rates of high resolution graphics data, but is highly advantageous for buffering the ingest and egress of wide band sensor data.



GPUs scale with hundreds of shader cores

Putting GPUs into systems

Until very recently, GPUs have only existed as unique chips. As discrete pieces of silicon, GPUs have traditionally relied on external host CPUs, typically Intel-based, as they themselves are not governed by an operating system. The host CPU directs the instructions to be executed as well as identifying the input source and output destination for data streams.

The physical implementation of GPUs has seen them on full length PCI cards, placed in Intel-based workstations, in addition to being incorporated directly onto CPU motherboards. MXM (Mobile PCI Express® Module) is a modular format for GPUs which, in addition to high-end consumer products, have seen applications in high performance embedded computing (HPEC) applications.

One example is a single MXM module on a 3U OpenVPX™ carrier board, another instantiates two MXM modules on a 6U OpenVPX carrier. These carriers are placed in adjunct slots alongside Intel SBC or Intel/FPGA DSP boards with a multi-lane PCIe super highway providing interconnect for streaming data ingest/egress. Rendered graphics destined for displays are output directly from the GPUs over display signal ports.

Recently, ARM cores that do run operating systems are finding their way into discrete GPUs. And, as will be discussed later, there are other architectural approaches that also provide new ways to integrate GPUs into systems.

Comparing a GPU to other math accelerators

How is a GPU like an FPGA? How is it different?

FPGAs have architectures that scale by the number of logic cells, a single cell being comprised of D flip-flops, LUTs and other elements. FPGAs scale these logic cells from a few hundred to thousands as the physical die footprint grows. GPUs scale shader processors in a similar fashion scaling from a few hundred in the smaller packages to a couple thousand. This range of sizes sees both FPGA and GPU technologies producing devices that are easily over 100W TDP (Total Design Power), which often excludes the largest high performers from being selected for designs destined for applications that have physical constraints such as size, weight, and power.

Neither FPGAs nor GPUs are well suited for handling the cognitive, or decision making, parts of an application; dynamic decision making functions are usually performed by a general purpose processor CPU cores. However, both FPGAs and GPUs excel at those parts of a high bandwidth, streaming application where the dataset and associated algorithms can be parallelized. The big difference between FPGAs and GPUs to date is that GPUs have a powerful floating point capability while FPGA's lack dedicated floating point processing, although this may change in the not-so-distant future.

FPGAs do have an advantage in that their I/O is user definable, whereas GPUs are solely dependent on PCI Express for data transport, in addition to supporting display port interfaces destined for monitors.

How is a GPU like an SIMD unit? How is it different?

The CPU cores within a device such as the Intel Core i7 (4th generation "Haswell") each have a SIMD (Single Instruction Multiple Data) unit, labeled Advanced Vector Extensions (AVX) 2. The AVX2, like GPU shaders, is a floating point engine. And, in fact, shaders are SIMD-like, with the same instruction operating in one cycle on multiple shaders.

A big point of differentiation lies in the fact that GPUs generally process elements independently, with no or little provision for static or shared data. Each GPU processor can only read from an input, perform an operation on it, and write it to the output. In contrast, the AVX2 SIMD units contain advanced register and memory manipulation functionalities, in addition to support for context switching, making for compute elements capable of cognitive applications.

This makes them suitable for more general purpose processing, such as scientific and financial applications that benefit from parallelism and high throughput.

Another major difference is the huge disparity in density, as GPUs have far more shaders than the number of SIMD processors in a Core i7. For instance, the NVIDIA® GTX 970M has 1280 shaders, as opposed to 4 SIMD units in the Core i7; this is partially offset by the fact that the individual AVX2 unit FLOPS benchmark is much higher than an individual GPU shader core, ~75 GFLOPS as opposed to ~2 GFLOPS.

Programming a GPU

CUDA™ and OpenCL™, pros and cons

Within the software domain, the most popular and ubiquitous API for drawing and rendering graphics is undoubtedly OpenGL, maintained by a not-for-profit consortium called the Kronos Group. However, when it comes to accelerating math, there are two main choices: CUDA from the GPU silicon manufacturer NVIDIA and something a bit newer and also from the Kronos Group, called OpenCL.

CUDA (Compute Unified Device Architecture) strictly targets NVIDIA GPUs for use as GPGPUs. CUDA is based on C and is predicated on a parallel computing programming model that looks to gain massive throughput by executing many threads concurrently. The big advantage to CUDA is that it exposes optimized, intrinsic functionalities buried within the NVIDIA GPU silicon. One example enables thread usage of shared memory regions, a previously cited omission of GPUs. In general, reviews from CUDA developers have been very favorable within the High Performance Computing (HPC) community. The big disadvantage with CUDA is that it is proprietary and closed to silicon outside of NVIDIA GPUs.

OpenCL (Open Computing Language) is also based on C and provides parallel computing constructs. The main differentiation lies in the fact that OpenCL targets and executes across a whole spectrum of heterogeneous platforms consisting of not just GPUs, but CPUs, DSPs, and even FPGAs. The big advantage with the open nature of OpenCL is portability and the ability to bridge heterogeneous computing elements within the same system or even the same silicon. The disadvantage can sometimes be experienced in the level of abstraction from the underlying silicon and optimization mileage will vary depending on OpenCL implementation by the associated silicon vendor. Great progress is being made by the various chip manufacturers and the community is growing at a much faster rate due to the variety of target silicon that is available.

GPU security concerns

Intel x86 and Power Architecture devices are General Purpose Processors (GPPs), well armored against security attacks but within a device the neighboring discrete GPU/GPGPUs are relatively unprotected and vulnerable. Discrete GPUs lack the integrated crypto capabilities and special security features, like secure boot, found in GPPs.

How GPUs have evolved

More and more cores

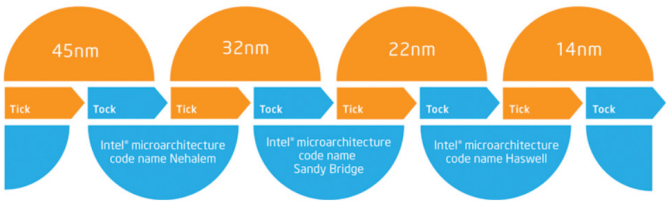
As is the case with other processing architectures, GPUs have seen, and continue to see, a steady increase in core count within a fixed power footprint, driven by lithography advances that increase die density. This creates an ever greater performance-to-power (i.e. FLOPS to Watt) ratio in all architectures, with GPUs currently in the lead.

Another key point, previously mentioned, is that CPU cores (e.g. ARM) are finding their way onto GPU dies. This provides a new level of autonomy to GPUs and also allows them to assume functions within a system beyond that of a mindless Giga/Tera FLOPS monster.

Intel and GPUs

For a number of years Intel Corporation has conducted a dual roadmap targeting two distinct macro markets. Xeon Server Class chips are effectively large homogeneous pools of x86 cores. These chips can be physically linked so that cache coherency scales across multiple devices, making them ideal for cloud computing, high performance computing servers, and applications that are extremely dynamic or multi-modal. Alternatively, Intel's Core i7/i5/i3 Mobile Class devices are geared towards portable devices such as laptops and ultrabooks, are therefore more environmentally rugged, and now have the requirement to sometimes render their own graphics.

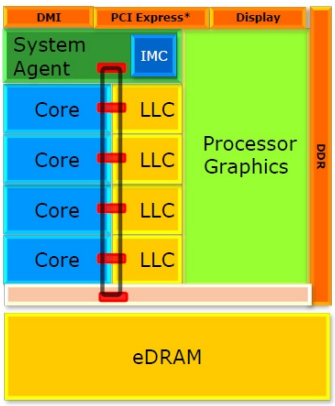
Intel has kept both of these product lines on a strict time cadence called "Tick-Tock" where Intel first creates a new microarchitecture in one generation and then shrinks the die geometry in the following generation. This results in the latest Core i7 Mobile Class device in particular having four x86 cores, each core with the latest AVX2 SIMD vector unit. Additionally, with the evolving microarchitecture and shrinking lithography, a tightly coupled GPU now resides on the same Core i7 die and this GPU will continue to advance in efficiency and density with the Tick-Tock march of future generations.



The Intel 'Tick-Tock' Development Model

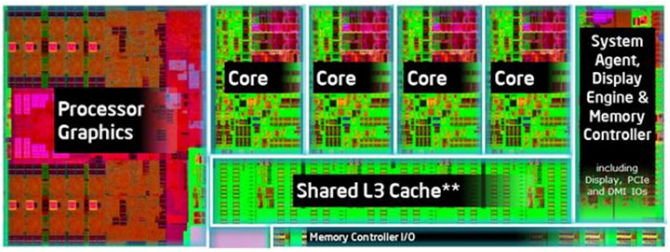
Historically, Intel CPUs have been coupled on motherboards with discrete GPUs from NVIDIA and AMD. The integration of the GPU device by Intel onto the Mobile Class die was partly spurred by marketplace contentions, but there are also definitive technical advantages to subsuming the GPU, such as removing PCB complexities and reducing thermal dissipation. The Aerospace Defense industry is a direct beneficiary of this development as it now has a very powerful heterogeneous CPU+GPU core where the GPU can be used to drive monitors or to process extreme sensor feeds.

Intel's integrated GPUs



Core i7 Architecture Diagram

The Intel Core i7 Gen 4 (Haswell) contains four x86 cores, each with its own dedicated AVX2 SIMD unit and each with an exclusive L1/L2 cache. There is a larger L3 cache (4-8 MB) that is shared among all four cores and the internal graphics processor (integrated GPU). This "sharing" is accomplished via a high speed ring bus (> 300 GBytes/sec) that serves as the data transport mechanism. Additionally, some of the Core i7 chip variants have additional embedded DRAM (eDRAM) that effectively functions as a Level 4 cache (victim cache to L3, up to 128 MB). Outside of external DDR3 memory, the method for getting into and out of the Core i7 is through lots of PCI Express Gen 3 lanes.



A Silicon View of the Core i7 Architecture

There are several different sizes available for the embedded GPU within a Core i7; one of the more popular ones is the GT2 (a.k.a HD Graphics 4600), which has 20 shader processors (Intel calls them Execution Units (EU)). These EUs produce >350 GFLOPS of single precision floating point processing. Additionally, the GT2 supports 3 display ports for rendering graphics.

Advantages for Defense electronics

Some applications will not be candidates for tossing aside a discrete NVIDIA or AMD GPU in favor of the embedded Intel Mobile Class GPUs. This includes applications that require TFLOPS of processing and/or GBytes of external GDDR5 DRAM, which are what the MXM modules with the larger discrete GPUs provide. However, there are other applications than can greatly benefit from the onboard GPU in the Core i7, where the GFLOPS performance, excluding CPU AVX2 units, meets the need and the memory capacity is sufficient.

- The first obvious advantage to Defense electronics is the removal of an entire adjunct GPU board (or more), which equates to an entire FRU slot for a deployed chassis, saving size, weight, power, and overall cost for the entire system.
- Another key advantage with the onboard GPU is the extremely low latency made possible by its proximity to the CPU cores. With all the CPU cores and the GPU interconnected by a lightning-fast ring bus and passing data at the caching level, latency benchmarks are greatly improved when compared to data transport to/from an Intel CPU device on one board and a discrete GPU on an second board interconnected via PCI Express.
- A third significant advantage to consider is the increased security surrounding the Intel devices as opposed to the large discrete GPUs, which currently

do not have adequate provisions for protection. Intel processors include capabilities for hardware-based secure key generation and boot integrity protection.

Example Use Cases

Graphics Processing Units continue to gain popularity in the Aerospace & Defense industry. They are utilized in both modes as previously described, as image renderers (GPUs) or math accelerators (GPGPUs). Both the discrete GPUs and the integrated GPUs are growing in terms of performance and memory capacity. Which type makes the most sense is really a question that needs to be answered by the requirements of the Program.

We are seeing GPUs being used to render displays for 360° situational awareness on modern military platforms. This enables the pilot or driver to effectively see through the ceiling, walls and floor of a vehicle. Even more popular, is GPU usage to capture image sensor data (e.g. via gigapixel cameras) followed by real-time image manipulation, optimization, and display. This latter application of GPUs is becoming particularly popular with Unmanned Arial Vehicles (UAVs) that are carrying multiple camera types (i.e. electro-optic and infrared) that need to be ortho-rectified and stitched together, just for starters.

Sensor data other than standard imagery is also a target for GPUs. STAP and SAR radar, which are hungry for FLOPS, are seeing pulse compression and Doppler processing occurring in GPUs. SIGINT applications requiring high throughput, wideband frequency domain analysis are being targeted by GPUs. The embedded GPUs are becoming particularly attractive for some applications given their proximity to the CPU cores all on the same die. Applications with stringent latency requirements, in addition to processing throughput, can greatly benefit from a heterogeneous processor such as the Intel Core i7, given the tightly coupled ring bus infrastructure between the embedded GPU and the AVX2 enabled x86 cores. Electronic warfare (EW) applications, such as Cognitive EW, that were dismissive of discrete GPUs due to high latency figures, are now considering the integrated device as it gives them the throughput of the GPU, the cognitive capabilities of the CPUs, and the low latency of the ring bus interconnected caching. OpenCL, as previously discussed, represents the software enabler that gives developers the access they need to realize the benefits of this heterogeneous device.

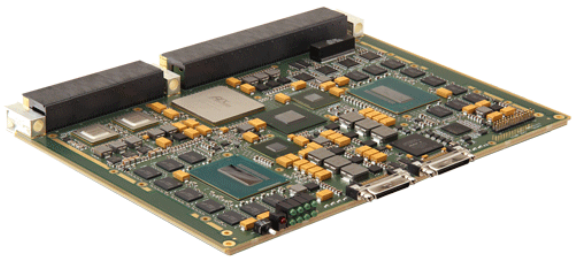
Choose the product that fits your design requirements

A choice of modules implementing integrated GPUs

Curtiss-Wright offers system designers a choice of performance points and form factors. All modules are based on the OpenVPX standard and ruggedized for deployment in harsh environments.

A Dual-node integrated GPU module – the CHAMP™-AV9

A prime example of a design implementation incorporating embedded GPUs is Curtiss Wright's CHAMP-AV9 Intel Core i7 Multiprocessor 6U OpenVPX DSP Board. This module brings two Core i7 processors onto a single PCB and is designed to withstand the severe environments that the Aerospace and Defense industry imposes upon it. The combined GFLOPS metric for just the AVX2 SIMD units on the two Haswell Core i7's equates to 614 GFLOPS. However, if we were to enlist the embedded GT2 GPUs as GPGPU math accelerators, we would see an overall total of 1,254 GFLOPs of processing power. Alternatively, if the GPUs were leveraged to render displays, the CHAMP-AV9 provides for a total of 6 individual display ports, 3 per Core i7 device.



The CHAMP-AV9 6U OpenVPX DSP Module

One real consideration to take into account with the Core i7 computational power house is the importance of providing ample memory throughput and capacity, as well as high speed I/O, and lots of it. Rarely are GPUs compute bound; they are much more likely to suffer from being memory bound or I/O bound. This is why the CHAMP-AV9 employs

up to 32 GB of DDR3 to tackle memory capacity in addition to the routing of all genres of high speed I/O on and off module.

Signal integrity over an extended temperature range is of particular importance as the CHAMP-AV9 routes 1,600 MHz memory lanes, PCIe Gen 3, DisplayPort™, and 40GigE/InfiniBand® in a very tight 6U OpenVPX footprint. The CHAMP-AV9 employs Curtiss-Wright Fabric40™ technology to address this issue and provide high signal integrity at the extreme signaling rates used by high bandwidth fabrics.

A 6U SBC powered by an integrated GPU – the VPX6-1958

Many applications have FLOPS requirements that are satisfied by a single powerful Core i7 processor; the VPX6-1958 Single Board Computer efficiently meets that need. This rugged, high performance 6U OpenVPX SBC employs the same Haswell Core i7 processor as the CHAMP-AV9, delivering 627 GFLOPs per module through the combination of AVX2 SIMD units across the four cores and the integrated GPU.

The VPX6-1958 6U OpenVPX Single Board Computer

With a high speed, dual-channel DDR3 memory subsystem supporting up to 32 GB of SDRAM, the VPX6- 1958 is able to maximize the throughput delivered by Core i7 processor, while a massive amount of memory, up to 128 GB of SATA SSD, make it an ideal SBC for handling applications with demanding storage, data logging and sensor processing requirements.

Configuration flexibility is a hallmark of the VPX6-1958, supporting a wide variety of mezzanine daughter cards on sites enabling two XMCs or one PMC and one XMC module.

The VPX6-1958 also has a host of standard I/O interfaces, including four independent Gigabit Ethernet ports, multiple RS-232, RS-422, SATA and USB ports, discrete DIO, DVI and dual-mode DisplayPort and VGA, and analog audio ports. It implements Curtiss-Wright's Fabric40 technology to deliver end-to-end support for both 10/40 Gbps Ethernet and InfiniBand fabrics, with proven performance OpenVPX Gen3 signaling and reduced signal integrity risks.

An integrated GPU for SWaP-constrained systems – the VPX3-1258

Designs facing severe SWaP constraints can still access the processing performance of an Intel integrated GPU. The VPX3-1258 is a 3U OpenVPX SBC featuring the latest 4th Gen Intel Core i7 (Haswell) processor. Pin-compatible with our previous generations of Intel SBCs, the VPX3-1258 offers the highest performance Intel processing in the space-efficient 3U form factor.

The VPX3-1258 3U OpenVPX Single Board Computer

The VPX3-1258 has the same Core i7 processor as its 6U cousins, delivering 627 GFLOPs. And, despite its compact size, the VPX3-1258 offers I/O flexibility to go with its processing performance. A local XMC mezzanine site supports an independent 8-lane PCIe Gen3 bus directly to the processor and standard I/O interfaces include 2 four independent Gigabit Ethernet ports, multiple RS-232, RS-422, SATA and USB ports, discrete DIO, DVI and dual-mode DisplayPort and VGA, and analog audio ports.

With up to 16 GB of dual-channel high speed ECC protected DDR3 memory, the VPX3-1258 provides up to 25.6 GB/s memory throughput, maximizing the capabilities of the processor. Using configurations that reach 32 GB of high speed SATA SSD memory, the VPX3-1258 is well-suited for complex applications with demanding sensor processing requirements, or high speed data processing, logging and storage needs.

Authors

Marc Couture
B.S and M.S.,
Electrical Engineering

Senior Product Manager
Curtiss-Wright Defense Solutions

Summary

System designers must regularly evaluate the many choices of processing elements available on the market, selecting the architecture that best meets the needs of a specific program. GPUs integrated within Intel's Core i7 processors are a recently available processing variant, with significant advantages to many types of Aerospace Defense applications. SWaP savings, high performance, low latency and enhanced security are all characteristics of these integrated GPUs.

Curtiss Wright will continue to choose the best and brightest processing elements for the Aerospace and Defense market. Expect to see new solutions utilizing the latest discrete GPU technologies. Additionally, the 5th generation 5 Intel Core i7 "tick" isn't far off in the future.

Learn More

Application Example: Using Intel-based COTS for On-deck Optical Threat Detection

CHAMP-AV9 6U OpenVPX DSP

VPX6-1958 6U OpenVPX SBC

VPX3-1258 3U OpenVPX SBC

White Paper: Massive SSD Storage on Intel Single Board Computers

White Paper: Intel Single Board Computers for Small to Mid-Size ISR Applications

White Paper: Understanding HPEC Computing – The Ten Axioms

White Paper: Fabric40 – An Introduction