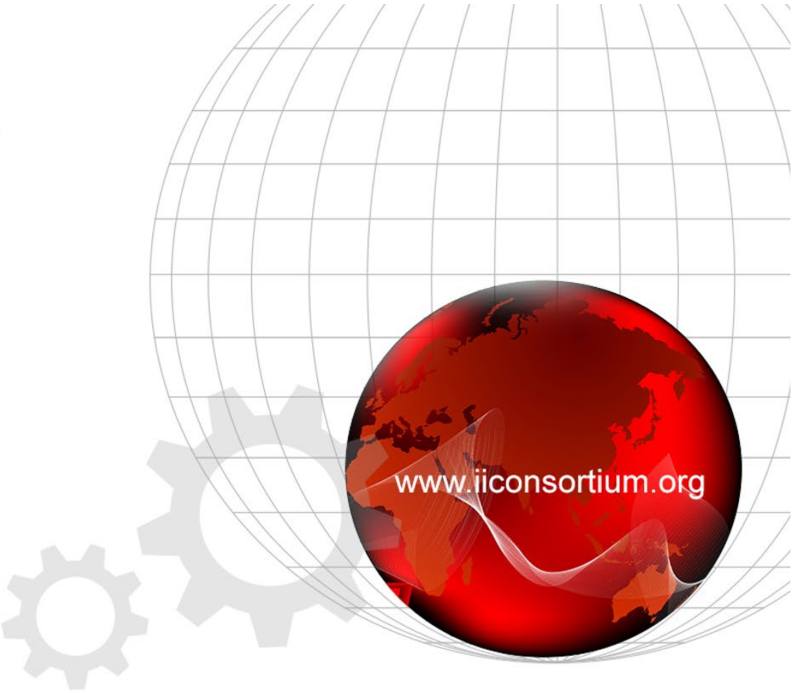


Heterogeneous Computing in the Edge

Authors:

Charles Byers
Associate Chief Technical Officer
Industrial Internet Consortium
byerschuck1@gmail.com

INTRODUCTION

Heterogeneous computing is the technique where different types of processors with different data path architectures are applied together to optimize the execution of specific computational workloads. Traditional CPUs are often inefficient for the types of computational workloads we will run on edge computing nodes. By adding additional types of processing resources like GPUs, TPUs, and FPGAs, system operation can be optimized.

This technique is growing in popularity in cloud data centers, but is nascent in edge computing nodes. This paper will discuss some of the types of processors used in heterogenous computing, leading suppliers of these technologies, example edge use cases that benefit from each type, partitioning techniques to optimize its application, and hardware / software architectures to implement it in edge nodes.

Edge computing is a technique through which the computational, storage, and networking functions of an IoT network are distributed to a layer or layers of edge nodes arranged between the bottom of the cloud and the top of IoT devices[1]. There are many tradeoffs to consider when deciding how to partition workloads between cloud data centers and edge computing nodes, and which processor data path architecture(s) are optimum at each layer for different applications.

Figure 1 is an abstracted view of a cloud-edge network that employs heterogenous computing. A cloud data center hosts a number of types of computing resources, with a central interconnect. These computing resources consist of traditional Complex Instruction Set Computing / Reduced Instruction Set Computing (CISC/RISC) servers, but also include Graphics Processing Unit (GPU) accelerators, Tensor Processing Units (TPUs), and Field Programmable Gate Array (FPGA) farms and a few other processor types to help accelerate certain types of workloads.

Many of the capabilities of the cloud data center are mirrored in the heterogenous computing architecture of the edge computer node. It includes modules for multiple processor types, including CISC/RISC CPUs, GPUs, TPUs, and FPGAs. The compute workloads can not only be partitioned between edge and cloud (see the companion article in this issue[2]), but also partitioned between the various heterogenous processing resources on both levels.

[1] Industrial Internet Consortium, "The Industrial Internet of Things Distributed Computing in the Edge", OCT 2020, IIoT Distributed Computing in the Edge (iiconsortium.org)

[2] C. Byers, "Key Criteria to Move Cloud Workloads to the Edge," IIC Journal of Innovation, June 2021

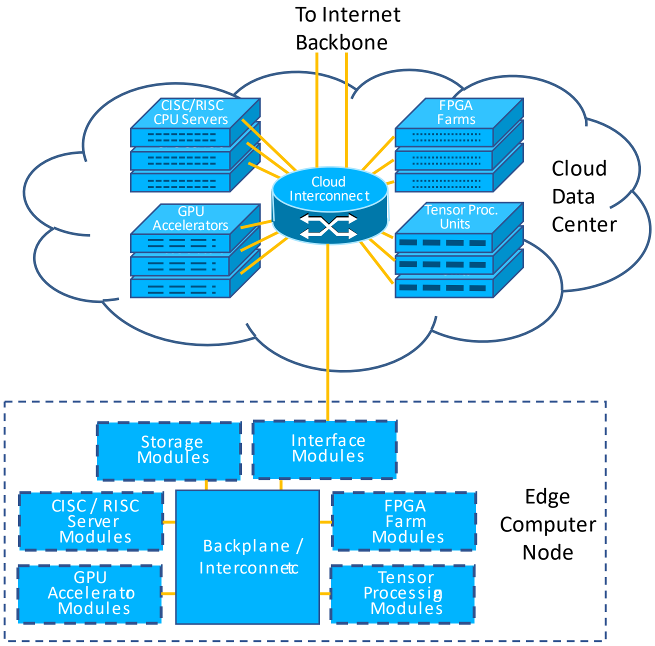


Fig. 1: Heterogeneous Computing in the Cloud - Edge Hierarchy.

Edge computing nodes in common deployment today typically use a homogenous computing base, that is all of the processing done on an edge node runs on the same type of CPU. CISC / RISC processors are by far the most popular solutions. Smaller edge nodes may have a single core CPU chip providing their processing power, often using X86 or ARM architectures. Larger edge nodes include multicore processors, with between two and about 32 X86, ARM, or RISC-V cores, or include multiple CPU chips of the same type.

Processor vendors such as Intel[3], AMD[4], and the ARM ecosystem[5] have defined conventional CISC / RISC processor chips that are optimized for these edge computing applications, by virtue of their specialized data path architectures, buses, power dissipation profiles, I/O structures, memory and storage subsystems, physical size, and environmental specifications. Those processor vendors provide hardware reference designs to aid implementers in the application of these processor chips to edge node applications.

[3] Intel | Data Center Solutions, IoT, and PC Innovation

[4] Welcome to AMD l High-Performance Processors and Graphics

[5] Artificial Intelligence Enhanced Computing – Arm

Software infrastructure optimized for conventional CISC / RISC processors is evolving to meet the specific needs of the edge. Examples of edge SW platforms include Microsoft Azure Edge[6], Amazon Greengrass[7], VMware Edge[8] and open source edge software projects from the Eclipse Foundation[9], EdgeX Foundry[10], and the Linux Foundation[11] (among others).

These software packages manage the operating system infrastructure, configuration, security, orchestration, management, etc. of CISC / RISC processors in edge nodes. Once one of these software infrastructure packages is up and running on the processor chip of an edge node, algorithms, protocol stacks, and application software can be loaded on top to complete the functionality of the edge system.

These systems serve the requirements well for systems whose computational workloads primarily use single thread architectures and run-to-completion models. However, they start to break down for applications that use massive parallelism, or do not map efficiently to standard CISC/RISC CPU data paths. Different types of processors, such as GPUs, TPUs, and FPGAs can supplement the system's CISC/RISC processors to optimize its key performance attributes. That is where heterogenous processors come in.

KEY PERFORMANCE ATTRIBUTES OF EDGE COMPUTING

There are several key performance attributes that can be used to judge the suitability of a certain processing architecture to a specific set of applications. These attributes relate to performance, efficiency, scalability, density, cost, and many similar areas.

Throughput is probably the highest priority attribute. It is about how much of a specific model of processing can be accomplished by a given edge node. Measuring this is very application dependent. Throughput could be quantified using measures like sessions / users / devices supported, link bandwidth processed, latency, model complexity evaluated, transactions / inferences / operations per second, and similar measures. This is often the numerator of a performance ratio in the form of:

[6] IoT Edge | Microsoft Azure

[7] AWS IoT for the Edge - Amazon Web Services

[8] What Is Edge Cloud | Edge Cloud Solutions | VMware

[9] Edge Native Working Group | The Eclipse Foundation

[10] Welcome (edgexfoundry.org)

[11] LF Edge - LF Edge

Performance Ratio = Some throughput measure / Some cost measure.

In general, heterogenous computing in the edge seeks to increase the throughput values and / or reduce the cost values so the overall performance ratio of the system is optimized. Of course, optimizing these values for a certain application does not guarantee that it will be optimized for other applications. This implies that different configurations and complements of processor types may be required for different application sets, and no single configuration will serve all applications.

One important cost measure is the dollar cost to purchase the processor hardware. Getting more throughput per dollar of system purchase price is one important way of optimizing the total lifecycle cost of ownership of an edge system. The amount of throughput a dollar will buy is highly dependent upon system architecture, the capabilities of the processor chips, the efficiencies of the hardware and software infrastructure, and the requirements of the software and algorithms.

Another cost metric is energy used. Electrical power needed to continuously operate an edge node is usually the largest component of its ongoing operational expense. In many edge node applications, this power is supplied by batteries (at least during the times when AC power is unavailable), and adequate battery capacity for sustained operation is very costly. Also, the electrical energy that enters a processor chip is almost completely converted to heat that must be removed from the system, and the necessary cooling infrastructure is a strong contributor to the purchase and operational costs. Power and cooling can create absolute limits on the throughput of edge computers.

Space is another cost driver. Edge computers are located outside of traditional cloud data centers, in facilities like huts at the base of cell towers, roadside cabinets, underground vaults, micro data centers in shipping container-like enclosures, and mobile enclosures riding on vehicles or carried by humans. As processors get physically larger, the cost associated with providing that space grows rapidly. Often, it is impossible to support edge computer designs for a given network deployment model if their physical size exceeds a certain value, and this can constrain system throughput.

Weight is the final cost driver we will discuss. In certain deployment situations, especially aerospace, maritime or human portable deployments, there are strong constraints to the maximum weight of an edge node. The choices of processor technologies can have a strong influence on the overall weight of the system. If the weight of an edge node exceeds a certain limit, it may not meet the system requirements, and it will have to be redesigned or its functionality will have to be adjusted.

HETEROGENOUS PROCESSOR TYPES

There a number of processor architectures that can be used to replace, or more likely supplement, traditional CISC / RISC processors in edge nodes. Some of these processor types are well suited to single thread performance, some are well suited to massively parallel execution, and others are highly optimized for specific applications (like Artificial Intelligence / Machine Learning or signal processing). This section will describe each of the main types of processors that could be used in heterogeneous edge nodes, and their potentials for application to common edge computing use cases.

CISC / RISC CPUs

Traditional CISC / RISC CPUs are by far the most common processor infrastructure for edge computers. The history of this architecture dates back at least 75 years to ENIAC at the University of Pennsylvania[12], which many consider the first programmable digital computer. There are several excellent histories of the development of CISC and RISC processors including one from Computerworld[13].

A watershed event here is the introduction of the Intel 8086 CPU in 1978, which is the instruction set heritage of most of Intel's and AMD's current product lines. These X86 microprocessors are widely used in cloud and edge computing. Their mature software development ecosystem makes them the leader in diverse computer applications. X86 processors have been successfully deployed in hundreds of edge computing designs. Advanced chips from Intel and AMD integrate 32 or more X86 cores into a server chip.

Beyond X86, the ARM processor architecture is heavily used in edge computers. The ARM1 architecture was finalized in 1985. The ARM architecture is owned and maintained by ARM Holdings Ltd., which licenses IP cores to chip manufacturers including Microchip, TI, NXP, ST Micro, Nvidia, Apple, and many others. The ARM architecture continues to evolve, with the recent announcement of the ARM9[14].

Of special interest to the edge computing market is the so-called big.LITTLE architecture, where two different types of ARM CPUs are integrated into the same chip[15]. This supports many edge applications, where, for example a relatively modest, low power ARM CPU monitors sensors or

[12] ENIAC at Penn Engineering (upenn.edu)

[13] Timeline: A brief history of the x86 microprocessor | Computerworld

[14] Armv9: The Future of Specialized Compute - Arm Blueprint

[15] big.LITTLE – Arm

communications traffic, and when certain events are detected the much more powerful companion ARM core wakes up to perform advanced computations. In 2020, NVIDIA purchased ARM Holdings.

Rounding out the CISC / RISC CPU category is the relatively recent RISC-V architecture[16], introduced in 2015. Unlike the X86 and ARM architectures, which are owned and licensed by commercial companies, RISC-V is an entirely open source implementation. This means that manufactures of a System on Chip (SoC) can freely import, adapt and modify RISC-V without concern for royalties or intellectual property.

The ecosystem is rapidly developing surrounding RISC-V, with dozens of manufacturers of core intellectual property, SoCs, development boards, and systems. RISC-V may have advantages over X86 and ARM for future edge computing systems by virtue of its highly flexible architecture, expanding supply chain, and potentially lower cost of ownership commercial model.

There are a number of additional CISC / RISC CPU architectures competing for relevance in the edge computing market. For the sake of brevity, I didn't include an exhaustive list here.

A common attribute of all the CISC / RISC processor architectures is their CPU cores are optimized for single thread performance, and they generally do not perform as efficiently on algorithms that can use massive parallelism, or would benefit from specialized data path designs. CISC/RISC CPUs are the obvious choice for control planes, configuration engines, log file maintenance, management processes, and similar applications, and since these are essential to most systems, at least a modest CISC/RISC supervisory subsystem is helpful in most edge computing applications.

Larger multicore CISC/RISC CPUs can be applied to some parallel execution environments, but in general they start to lose efficiency for applications with more than 50-100 parallel execution threads. Beyond that, the addition of some other type of processing resource supporting more massive parallelism or specialized data paths makes sense, and that is the opportunity for heterogenous processors in the edge. That opportunity is the focus of the remainder of this paper.

GPUs

Graphics Processing Units (GPUs) are specialized processor architectures that were originally designed to accelerate graphical computing operations like geometric transformation, windowing, clipping, texture mapping, ray tracing, shading, and rendering. They achieve orders-of-magnitude performance improvement over CISC/RISC CPUs for the specialized graphical

[16] About RISC-V - RISC-V International (riscv.org)

computing tasks they are designed for. This is accomplished through the use of massively parallel arrays of relatively simple processors (for example, the NVIDIA "Ampere" microarchitecture used in their A40 product has 10752 processor cores[17]). GPUs also have advanced memory subsystems capable of moving data in and out of their associated memories many times faster than CISC/RISC CPUs.

NVIDIA and AMD are leading suppliers of GPU processors. They have been in a competitive race for over a decade, and this has fueled rapid innovation. Intel and ARM also provide GPUs, but theirs are mostly integrated with conventional CPUs for applications like laptop computers, and this model may not be optimal for modular edge nodes. Recently, both NVIDIA[18] and AMD[19] have introduced GPU products optimized for edge / IoT applications.

The subset of edge computing applications that can be optimized through the use of GPUs tend to follow certain design patterns, for example those workloads that are "embarrassingly parallel"[20]. These sorts of problems are common in graphics, image analysis and visual computing, where a scene can be broken down into pixels or regions, and each can be processed independently of all others, with minimal inter-thread communication. Other examples of problems that are amenable to parallel execution include database searches, matrix math, and some problems in artificial intelligence (such as neural network execution). They all continue to scale linearly with thousands of processors acting on the same problem in parallel. These workloads are common on edge computing, and the addition of GPUs to edge nodes can greatly improve the efficiency of systems.

The programing environment for GPUs is somewhat different from the traditional software development models used on CISC/RISC CPUs. In order to partition the algorithms effectively across the large numbers of GPU processors, special software constructs are needed so the programmer can help direct the parallelism, and special compilers are needed to help optimize the execution. The NVIDIA CUDA[21] (Compute Unified Device Architecture) is one of the most capable toolkits for parallel algorithm development on GPUs. There is a large ecosystem of developers and a thriving application marketplace for CUDA. The OpenCL (Open Compute

[17] NVIDIA A40 datasheet

[18] Edge Computing Solutions For Leading Technologies | NVIDIA

[19] AMD Ryzen™ Embedded Family | AMD

[20] EmbarrassinglyParallel Design Pattern (ufl.edu)

[21] CUDA Zone | NVIDIA Developer

Language) framework is a more standardized alternative to CUDA, and has some advantages for edge computing.

There are several commercially-available GPU hardware modules optimized for edge computing applications. One example is the NVIDIA "Jetson" TX2 module, capable of 1.3TFLOPs of computation throughput, with 256 CUDA cores, 8GB of DDR4 memory, less than 15 Watts power dissipation, and a physical size of 87mm x 50mm[22]. Their cost is less than $500 in small quantities.

TPUs

Tensor Processing Units (TPUs) are specialized computing architectures whose data paths are optimized for Artificial Intelligence / Machine Learning / Deep Learning (AI/ML/DL) operations. They can accelerate both the learning phase (where training data is digested into an AI model) and the inference phase (where AI models are executed on live data to produce insights). They offer extremely large performance / cost improvements compared to CISC/RISC CPUs, and significant improvements over GPUs for many workloads. They consist of massively parallel arrays of relatively simple processors intimately connected to high performance memory subsystems. A highly parallel matrix multiplier is at their core (capable of performing tens of trillions of multiplies per second)[23], which is an essential operation in machine learning.

TPUs were pioneered by Google, and have been offered as a service on their cloud network since 2016[24]. They are integrated into their cloud computing network in large arrays called "pods", and are offered as an alternative to CISC/RISC servers to accelerate AI/ML/DL workloads. NVIDIA announced their Deep learning Accelerator TPU as an open architecture[25]. Intel announced their Neural Compute Stick that includes some TPU functions[26].

As in GPUs, TPUs require specialized programming environments. Google has a proprietary system called TensorFlow, Nvidia has PyTorch, and Intel has an AI Analytics Toolkit to provide the software development environments for these specialized processor architectures.

TPUs at the edge present some unique challenges. The training data and AI models can be large, which can tax the network bandwidth and storage available at edge nodes. Edge nodes can be power, size, and weight constrained, making it difficult to install cloud-optimized TPU solutions.

[22] Jetson Modules | NVIDIA Developer

[23] First In-Depth Look at Google's TPU Architecture (nextplatform.com)

[24] Cloud TPU | Google Cloud

[25] NVIDIA Deep Learning Accelerator (nvdla.org)

[26] Intel® Neural Compute Stick 2

One response to this challenge is Google's "Coral" series of edge TPU products. Their Dev Board Mini has a throughput of 4TOPS, uses 2 watts of power, has 2GB of DDR3 RAM, has a physical size of 64mm x 48mm, and costs $100 in small quantities[27]. It uses a simplified programming environment called TensorFlow Lite.

FPGAs

Field Programmable Gate Arrays (FPGAs) are an important processing technology for edge computing. FPGAs consist of a large array of programmable logic blocks interconnected by an array of programmable data paths to implement custom hardware that is ideally matched to specific computational problems. The programmable logic blocks consist of a look-up table that implements logic functions, a storage element for intermediate results, and connections to the chip's programmable interconnect network. Many modern FPGAs expand the programmable logic blocks to also include multipliers, memory blocks, specialized I/O structures, and RISC processors. The programmable logic blocks and interconnect are initialized by a configuration file that is shifted in from external storage and stored in configuration RAM within the FPGA. Most FPGAs can be reconfigured dynamically, so it is possible to change the hardware data paths on the fly as different applications may require.

Two companies dominate the FPGA world, and both have been acquired by semiconductor giants. Altera was acquired by Intel in 2015[28], and AMD recently announced their acquisition of Xilinx[29]. The combination of a large processor company and a leading FPGA company creates a powerful combination in heterogenous computing. In addition to Altera and Xilinx, there are a number of smaller players in the FGPA space, including Lattice, Microchip, and QuickLogic.

FPGAs are programmed using a hardware design process and high level HW design language. This means the design skills needed to effectively produce the configuration files is closer to a hardware designer than a software programmer. However, the design automation systems for FPGAs continue to mature, making FPGA programming accessible to audiences with less HW experience[30].

FPGAs are applicable to many different workloads at the edge. Since the FPGA designs are highly optimized for specific applications, their performance and power can be well suited to the constraints of edge computing. Some examples include software defined radio subsystems, high

[27] Dev Board Mini datasheet | Coral

[28] Intel Completes Acquisition of Altera | Intel Newsroom

[29] AMD to Acquire Xilinx | AMD

[30] Xilinx Developer

performance computing, instrumentation, medical, and financial algorithms. FPGA manufacturers have been producing parts optimized for edge computing, for example the Xilinx Kria product portfolio[31].

DSPs

Digital Signal Processors (DSPs) are specialized computation elements optimized for signal processing workloads. They tend to have parallel arrays of multiply-accumulate engines that are valuable in implementing functions like digital filtering, convolution, FFTs, compression, waveform synthesis, and audio algorithms. Leading processor companies in the DSP space include TI[32], Analog Devices, Microchip, STMicroelectronics and NXP.

The programming environment for DSPs is often a variant of the environments used in CISC/RISC CPUs. However, knowledge of the specific data paths in the target DSP chip is often required to produce high performing code.

DSPs have some utility at the edge, especially in radio signal and audio processing. However, the algorithms that are traditionally implemented on DSPs are moving to GPUs and FPGAs for many edge implementations, where they can be run at greater efficiency by most measures.

ASICs / ASSPs / SoCs

Rounding out the heterogenous computing at the edge technology universe are Application Specific integrated Circuits (ASICs), Application Specific Standard Products (ASSPs) and Systems on Chips (SoCs). These represent full custom hardware implementations of specific computational capabilities. This is similar to the full custom data paths found in FPGAs, but they are not reconfigurable at the gate level. This makes them faster, smaller, and lower power, but their lack of programmability limits their versatility in diverse edge applications.

ASICs are made for a specific application, typically by the company who is building the system or box-level product. They represent the most optimized, best possible throughput and cost profile for the specific function they are designed for. However, ASICs have serious drawbacks, because it often takes over a year to design one, more than $10M to create the masks needed to fabricate the chip, and they are impossible to modify once built. These concerns limit the applicability of ASICs in networks, especially the highly dynamic environments found at the edge.

ASSPs are ASICs designed by semiconductor houses for a certain set of applications, and sold to all of their customers. This eliminates many of the risks associated with full-custom ASICs, but

[31] Kria Adaptive System-on-Modules (xilinx.com)

[32] Digital signal processors (DSPs) | Overview | Processors | TI.com

doesn’t offer product differentiation. Some examples of where ASSPs are valuable include video systems, wireless interfaces, routers, automotive, military, and industrial automation.

SoCs are a type of ASSP that integrates many different functions that are traditionally provided by a collection of chips onto a single die. These functions can include one or more cores of CISC/RISC CPUs, memory, I/O interfaces, and processor accelerators like GPUs, TPUs, or FPGAs. Their high levels of integration can greatly improve the size and weight of solutions, and that is why they are common in smartphones. SoCs are becoming an important implementation technology for edge nodes, especially smaller ones closest to the IoT devices.

HETEROGENOUS EDGE NODE HARDWARE ARCHITECTURES

The previous section described the various heterogenous processor types that can be combined into an edge node. But, how do we configure them in a way that enables the construction of edge nodes that are optimized for a specific set of workloads? Modularity is the key – different modules including the various types of CPUs, GPUs, TPUs, FPGAs, etc., are plugged into a modular infrastructure enabling an exact match between the needs of the computational workloads and the heterogeneous processors that support it.

Modular edge nodes have a logical architecture that partitions different components onto modules that can be individually configured, expanded, upgraded, and replaced. This typically involves some sort of board-and-backplane or stacking configuration. Figure 2 shows the important components of one type of modular edge node based upon a backplane and a collection of different types of boards implementing the heterogenous computing functions.

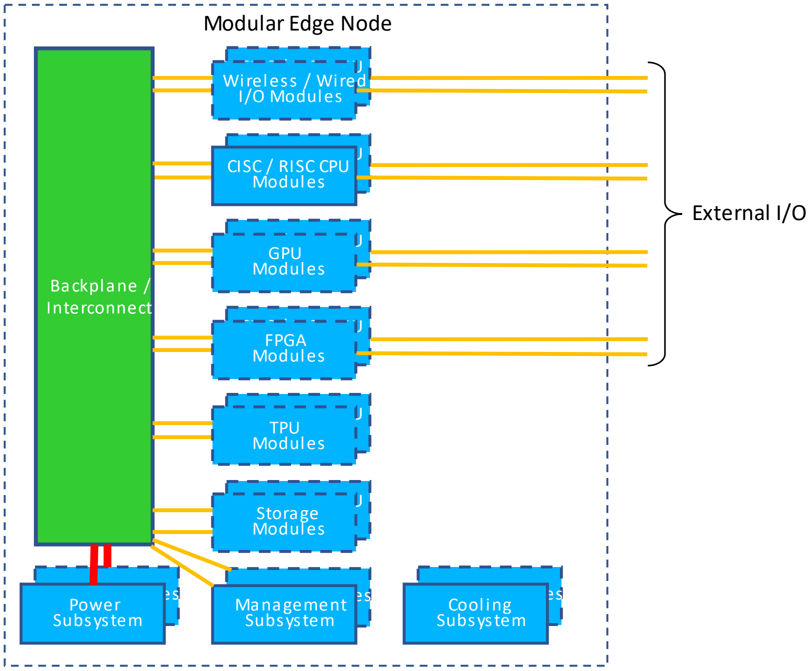


Fig. 2 - Modular Edge Node Architecture.

The common infrastructure subsystems required to “feed” and manage the modular edge node components appear at the bottom of the figure. A power subsystem accepts energy from the AC grid or a battery, converts it to the voltages required by the modular boards, and distributes it across the node.

A cooling subsystem provides a means of extracting waste heat from the modules. This is most often forced air using fans, but could be passive convection, heat pipes, conduction, or liquid cooling. A management subsystem configures and monitors all elements of the modular edge node via a dedicated internal management network. Finally, a rugged enclosure provides mechanical support, environmental isolation, and physical security for all the modules.

The interconnection between the modules is a key aspect of the modular edge node architecture. This is often implemented as a backplane with circuit board connectors to accept the modules. Conductors within the backplane provide paths between the various circuit boards. Individual board channels are often implemented as a differential pair signaling at 10-50Gb/s in each direction, and many such channels can be used in parallel to achieve the desired interconnect bandwidth.

Two topologies make sense for edge backplanes: star where one or more central switch fabric boards accept the traffic from all other modules and forward it to its intended destination module; and full mesh, where each module has dedicated channels to all other module positions on the backplane. In addition to the primary interconnect, the backplane carries an overlay network for management traffic, and the distribution paths for the power subsystem.

At least one CISC/RISC CPU module will be present in most modular edge nodes. These typically will include a single socket X86 or ARM CPU chip, along with associated memory, storage and I/O structures. The CISC/RISC CPU chip could be a server-class CPU with power dissipation over 100 Watts, but more likely it will be a laptop-class CPU with power ratings in the 40-Watt range. Memory is typically a few DRAM modules configurable in the 16-256GB range for edge applications. Storage is typically solid-state drives, with capacities in the 1-20TB range. It is interconnected to the backplane using a number of PCI-e or Ethernet channels, for a total usable bandwidth of tens of Gb/s

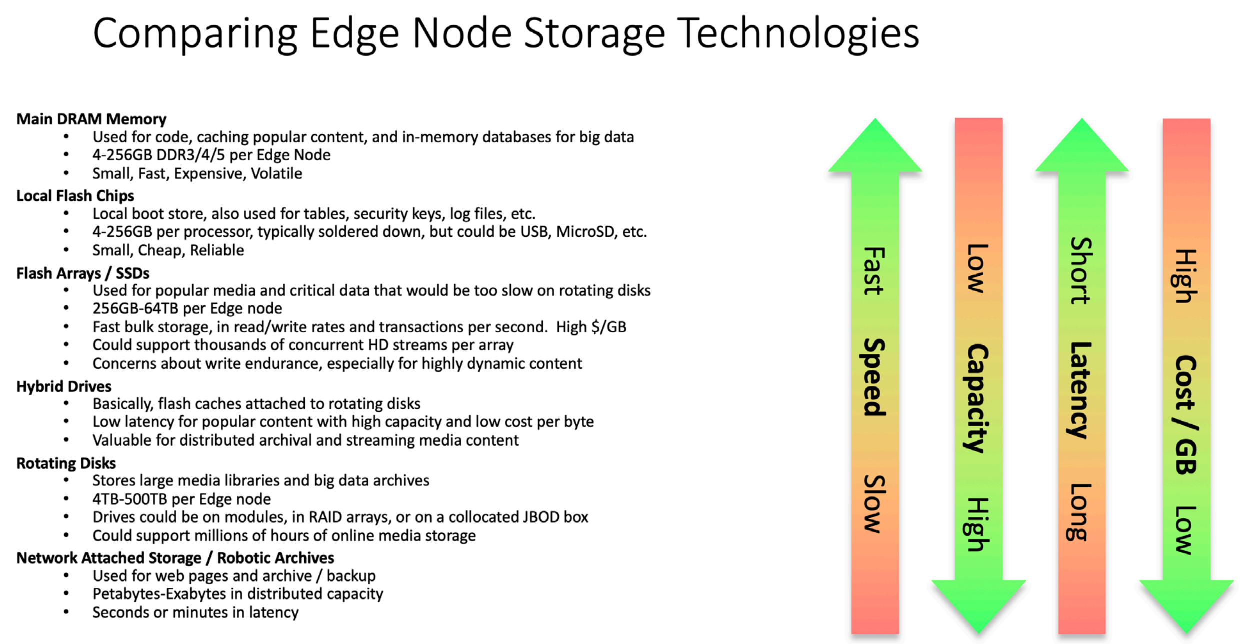
GPU modules will often have a modest CISC/RISC CPU as a host processor, and a fairly powerful GPU chip as the main processor. This chip may have thousands of GPU cores, and connect to 4GB+ of fast memory. It communicates with the backplane over many channels of high-speed interconnect, with a total bandwidth of several tens of Gb/s. As GPU chips are power hungry, this board may also feature advanced thermal management technologies like vapor chambers, heat pipes or local booster fans.

FPGA modules may be configured as farms of 4-16 FPGA chips, managed by a small host processor. Each FPGA will likely have one or two DIMM sockets for DRAM, but some applications may not require them and leave those sockets empty. Some of the rich I/O structures on the FPGAs will interconnect with their on-board peers over a mesh network, while other I/O channels will go to the backplane connector, and still others leave the modular edge node as external I/O links.

The architecture of TPU modules will be similar to the GPU modules. It will typically include a small host processor, and one or more large TPU chips. Each TPU will have 4-16GB of fast DRAM. Some TPUs may have local HDDs or SSDs to store the large datasets associated with AI/ML/DL training and model data. TPU modules communicate with the backplane over tens of Gb/s of channel bandwidth.

DSPs, ASICs, ASSPs, and SoCs all have potential applicability as modules for this edge node architecture. For certain applications, these will be parts of highly efficient implementations. However, for many edge applications, especially multi-tenant edge applications where multiple computational workloads from multiple customers are supported concurrently on the same edge node, some mix of the more versatile and higher performing technologies of CISC/RISC CPUs, GPUs, TPUs, and FPGAs will likely dominate, so those are emphasized on Figure 2.

Storage is another important aspect of modular edge node design. As shown in the sidebar, storage will usually be implemented as a hierarchy, involving main memory, solid state drives, rotating drives and network storage. Different types of modules, for example flash arrays and rotating disk arrays, can be configured on the same backplane as required by the applications.



Comparing Edge Node Storage Technologies

Main DRAM Memory
- Used for code, caching popular content, and in-memory databases for big data
- 4-256GB DDR3/4/5 per Edge Node
- Small, Fast, Expensive, Volatile

Local Flash Chips
- Local boot store, also used for tables, security keys, log files, etc.
- 4-256GB per processor, typically soldered down, but could be USB, MicroSD, etc.
- Small, Cheap, Reliable

Flash Arrays / SSDs
- Used for popular media and critical data that would be too slow on rotating disks
- 256GB-64TB per Edge node
- Fast bulk storage, in read/write rates and transactions per second. High $/GB
- Could support thousands of concurrent HD streams per array
- Concerns about write endurance, especially for highly dynamic content

Hybrid Drives
- Basically, flash caches attached to rotating disks
- Low latency for popular content with high capacity and low cost per byte
- Valuable for distributed archival and streaming media content

Rotating Disks
- Stores large media libraries and big data archives
- 4TB-500TB per Edge node
- Drives could be on modules, in RAID arrays, or on a collocated JBOD box
- Could support millions of hours of online media storage

Network Attached Storage / Robotic Archives
- Used for web pages and archive / backup
- Petabytes-Exabytes in distributed capacity
- Seconds or minutes in latency

Fig. 3 - Sidebar - Comparing Storage Technologies.

Orchestrating Heterogenous Edge Processes

The modular edge node described above requires sophisticated orchestration to make efficient use of its heterogeneous resources. Orchestration systems follow the requirements of the various workloads served by a modular edge node, and map portions of those workloads to the different modular processing components.

Orchestration manages the configuration of the modules, and generates alerts if additional modular resources are needed at any given edge node. Edge orchestration systems not only manage the workload division between the various heterogenous processor components of a given edge node, they also manage the distribution of workloads across different edge nodes, and between edge and cloud processors. Recent innovations in cloud-edge orchestration apply AI techniques to help optimize workload partitioning[33].

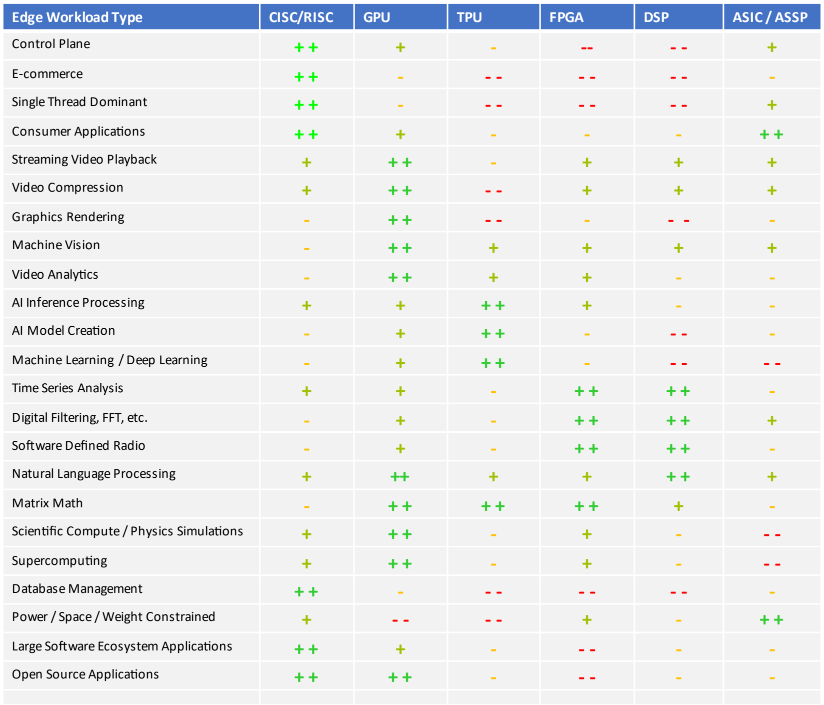
Orchestration systems often make use of containers to coordinate the software execution environments. For the Kubernetes ecosystem, KubeEdge is one framework for edge

[33] Cloud-Edge Orchestration for the Internet-of-Things: Architecture and AI-Powered Data Processing | IEEE Journals & Magazine | IEEE Xplore

orchestration[34]. The Linux Foundation's project Eve[35] also addresses edge orchestration. The Eclipse Foundation has several edge projects that include orchestration capabilities, including IoFog 2.0[36]. These frameworks can help automate and optimize the application of heterogenous processing resources in edge nodes.

ARCHITECTURAL TRADE-OFFS

Now that we have explored some of the alternatives for heterogenous computing in the edge, how do we decide which type of processor resource is optimum for a given workload or application? Figure 4 shows a set of exemplary edge processor workloads, and rates the various heterogeneous processor types on their suitability for each workload on a four-level scale (+ +, +, -, or - -).



Edge Workload Type	CISC/RISC	GPU	TPU	FPGA	DSP	ASIC / ASSP
Control Plane	+ +	+	-	--	- -	+
E-commerce	+ +	-	- -	- -	- -	-
Single Thread Dominant	+ +	-	- -	- -	- -	+
Consumer Applications	+ +	+	-	-	-	+ +
Streaming Video Playback	+	+ +	-	+	+	+
Video Compression	+	+ +	- -	+	+	+
Graphics Rendering	-	+ +	- -	-	- -	-
Machine Vision	-	+ +	+	+	+	+
Video Analytics	-	+ +	+	+	-	-
AI Inference Processing	+	+	+ +	+	-	-
AI Model Creation	-	+	+ +	-	- -	-
Machine Learning / Deep Learning	-	+	+ +	-	- -	- -
Time Series Analysis	+	+	-	+ +	+ +	-
Digital Filtering, FFT, etc.	-	+	-	+ +	+ +	+
Software Defined Radio	-	+	-	+ +	+ +	-
Natural Language Processing	+	++	+	+	+ +	+
Matrix Math	-	+ +	+ +	+ +	+	-
Scientific Compute / Physics Simulations	+	+ +	-	+	-	- -
Supercomputing	+	+ +	-	+	-	- -
Database Management	+ +	-	- -	- -	- -	-
Power / Space / Weight Constrained	+	- -	- -	+	-	+ +
Large Software Ecosystem Applications	+ +	+	-	- -	-	-
Open Source Applications	+ +	+ +	-	- -	-	-

Fig. 4 - Heterogeneous Processor Selection.

Of course, the requirements for edge workloads are highly varied, and it is impossible to anticipate the exact fit for heterogeneous processor types for specific algorithms or applications.

[34] KubeEdge

[35] EVE - LF Edge

[36] Eclipse ioFog

The table is a useful starting point, and hopefully one or more of the example workloads given are related to the closely enough to the target workloads you are considering for an edge computing deployment to provide a reasonable starting point for your processor technology selection decisions.

CONCLUSIONS

The performance, cost and efficiency of edge computing can be optimized through careful selection of various types of heterogeneous processors to run various aspects of the edge workloads. Different processor types, including CISC/RISC CPUs, GPUs, TPUs, and FPGAs can be combined into modular implementations that are optimized for the offered workloads. Modular orchestration systems dynamically adapt the heterogenous processor infrastructure to match the requirements of the offered load. Heterogeneous processor techniques are especially valuable in edge computing, as they can greatly improve the efficiency, throughput and cost measures in resource constrained edge nodes.

ACKNOWLEDGEMENTS

The views expressed in the *IIC Journal of Innovation* are the contributing authors' views and do not necessarily represent the views of their respective employers nor those of the Industrial Internet Consortium.

➢ Return to IIC Journal of Innovation landing page for more articles and past editions