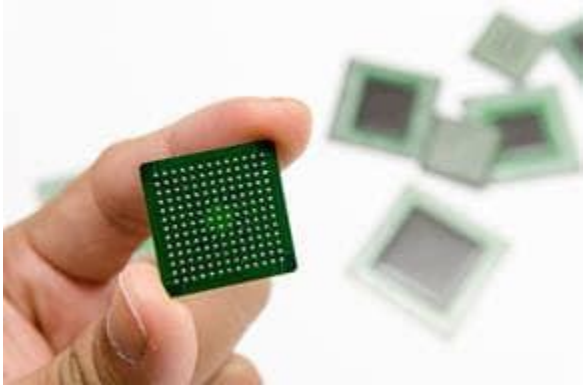
FPGA vs CPU vs GPU vs Microcontroller: How Do They Fit into the Processing Jigsaw Puzzle?

5 Oct 2018



Decades ago, **central processing units (CPUs)** were implemented in discrete transistors and, later, in integrated-circuit logic devices. That all changed when the first microprocessor, Intel's 4-bit 4004, made its debut in 1971. For many years, products from Intel and its competitors were the only choice for engineers who wanted programmable processing power in their designs.

CPU vs MPU

Now the CPU is a component in a larger system. A standalone **microprocessor unit (MPU)** bundles the CPU with peripheral interfaces such as **DDR3** & DDR4 memory management, PCIe, serial buses such as USB 2.0, USB 3.0, Ethernet and more, so these designs are flexible and versatile and are designed to run multi-tasking high-level operating systems (OSes) such as Windows, iOS, Linux, etc. For more compact designs, a **microcontroller unit (MCU)** combines the CPU core with

internal memory, many peripherals on a single integrated circuit or into a single package, where they typically run a stripped-down OS.

The microprocessor holds sway in laptop and desktop computing, and microcontrollers are ubiquitous in embedded applications such as automotive instrument panels, motor controllers or smart cards. The CPU hardware architecture of these devices is intended for general-purpose use but often includes specialized blocks, such as floating-point units (FPUs) for mathematical operations. Low-end CPUs execute operations in a sequential manner, but multiple processing cores are now standard in high-end microprocessors and microcontrollers. Intel's Xeon has up to 22 cores.

FPGA and ASIC

Although these CPUs are adequate for general-purpose computation, a slew of computationally demanding applications has emerged in recent years that require more specialized architectures. Examples include high-speed search; machine learning and artificial intelligence (AI); high-performance computing (HPC) in data centers; real-time graphics processing, including virtual reality and video gaming; industrial products such as digital motor control and automotive-related applications such as advanced driver assistance systems (ADAS) and, soon, autonomous vehicles.

Designers in these fields can draw upon three additional processing choices: the graphics processing unit (GPU), the **field-programmable gate array (FPGA)** and a custom-designed **application-specific integrated circuit (ASIC)**.

GPU vs FPGA

The GPU was first introduced in the 1980s to offload simple graphics operations from the CPU. As graphics expanded into 2D and, later, 3D rendering, GPUs became more powerful. Highly parallel operation is highly advantageous when processing an image composed of millions of pixels, so current-generation GPUs include thousands of cores designed for efficient execution of mathematical functions. Nvidia's latest device, the Tesla V100, contains 5,120 CUDA cores for single-cycle multiply-accumulate operations and 640 tensor cores for single-cycle matrix multiplication. Many algorithms in other fields lend themselves to parallel execution, so GPUs

have spread far beyond their initial application. Many of the world's fastest supercomputers, for example, include thousands of both GPUs and CPUs.

The FPGA consists of internal hardware blocks with user-programmable interconnects to customize operations for a specific application. In contrast to the other devices mentioned, the connections between blocks can readily be reprogrammed, changing the internal operation of the hardware and allowing an FPGA to accommodate changes to a design, or even support a new application, during the lifetime of the part. This flexibility makes the FPGA a great choice for applications in which standards are evolving, such as digital television, consumer electronics, cybersecurity systems and wireless communications.

At the other end of the spectrum, an ASIC is designed specifically for its intended application. It has only the blocks required for optimum operation, including a CPU, GPU, memory and so on. Although designers can incorporate third-party IP cores such as the Arm Cortex CPU—or predesigned blocks for standard functions such as an Ethernet physical layer—an ASIC is a ground-up design. It's best-suited for high-volume applications. The Tensor Processing Unit (TPU), for example, is an accelerator developed by Google specifically for neural-network machine learning. Google also offers TPU access to outside companies through its cloud-computing unit.

The Future?

Increasingly, we're seeing a convergence of the three categories as vendors search for the optimum feature set for emerging applications. SoC FPGAs come with hard- or soft-IP CPUs, GPUs and DSP blocks. CPUs include hardware accelerators and ASICs for cryptographic functions, and **NVIDIA**'s Tesla T4 GPU includes embedded FPGA elements for AI inference applications.

Regardless, this is a lot of information. To help you keep track of everything, we've created a couple of tables that summarize the main features of the four devices, together with some of their relative strengths and weaknesses.

To download the CPU vs FPGA vs GPU vs ASIC Cheat Sheet, click here.

	CPU	FPGA	GPU	ASIC
Overview	Traditional sequential processor for general-purpose applications	Flexible collection of logic elements and IP blocks that can be configured and changed in the field	Originally designed for graphics; now used in a wide range of computationally intensive applications	Custom integrated circuit optimized for the end application
Processing	Single- and multi-core MCUs and MPUs, plus specialized blocks: FPU, etc.	Configured for application; SoCs include hard or soft IP cores (e.g., Arm)	Thousands of identical processor cores	Application-specific: may include third-party IP cores
Programming	OSes, APIs run huge range of high-level languages; assembly language	Traditionally HDL (Verilog, VHDL); newer systems include C/C++ via openCL & SDAccel	OpenCL & Nvidia's CUDA API allow general-purpose programming (e.g., C, C++, Python, Java, Fortran)	Application-specific: TensorFlow open-source framework for Google's TPU; CPU manufacturers (e.g., Intel) include tools with new ASIC releases
Peripherals	Wide choice of analog and digital peripherals in MCUs; MPUs include digital bus interfaces	SoCs include many transceiver blocks, configurable I/O banks	Very limited; e.g., only cache memory	Tailored to application: may include industry-standard functions (USB, Ethernet, etc.)

Strengths	Versatility, multitasking, ease of programming	Configurable for specific application; configuration can be changed after installation; high performance per watt; accommodates massively parallel operation; wide choice of features: DSPs, CPUs	Massive processing power for target applications—video processing, image analysis, signal processing	Custom-designed for application with optimum combination of performance and power consumption
Weaknesses	OS capability adds high overhead; optimized for sequential processing with limited parallelism	Relatively difficult to program; second-longest development time; poor performance for sequential operations; not good for floating-point operations	High power consumption, not suited to some algorithms; problems must be reformulated to take advantage of parallelism, but API frameworks provide abstraction	Longest development time; high cost; cannot be changed without redesigning the silicon

It's also worth considering how these choices stack up in some common applications. As shown in the table, designers can often use any or all of the options either alone or, more likely, in combination.

Applications	CPU	FPGA	GPU	ASIC	Comments
Vision & image processing		✔	✔	✔	FPGA may give way to ASIC in high-volume applications

AI training			✔		GPU parallelism well-suited for processing terabyte data sets in reasonable time
AI inference	✔	✔	✔	✔	Everyone wants in! FPGAs perhaps leading; high-end CPUs (e.g., Intel's Xeon) and GPUs (e.g., Nvidia's T4) address this market
High-speed Search	✔	✔	✔	✔	Microsoft's Bing uses FPGAs; Google uses TPU ASIC; CPU needed for coordination & control
Industrial motor control	(✔)	✔		✔	Many motor-control MCUs and ASICs available; FPGAs offer a quick-turn ASIC alternative
Supercomputer HPC	✔		✔		Majority of TOP500 supercomputers uses some combination of CPUs and GPUs
General-purpose computing	✔		(✔)		CPU most versatile, flexible option; GPUs beginning to perform some tasks
Embedded control	✔	✔		✔	CPUs (-> MCU) dominant in low-cost, space-constrained, low-power, mobile applications
Prototyping, low-volume		✔			FPGAs best choice for low-volume, high-end applications; also pre-silicon validation, post-silicon validation and firmware development

•

•

The Future of Single Board Computers and Artificial Intelligence

2 years ago