



mitronics

White Paper

Low Power Hybrid Computing for Efficient Software Acceleration



mitrion™
by mitronics

Contents

Low Power Hybrid Computing for Efficient Software Acceleration	3
Hybrid Computing	3
Accelerators and Hybrid Computing Systems	4
Processing Sweet Spots in Hybrid Computing	4
Benefits of Hybrid Computing	5
FPGA-based Accelerators	6
FPGA Basics	6
Configuring the FPGA	7
Synthesis, Place and Route	7
FPGAs Compared to Microprocessors	7
Power Consumption	8
FPGA-based Hybrid Computer Systems	8
Developing Accelerated Applications	9
Algorithm Development for FPGAs	10
Hardware Design	10
The Mitrion Virtual Processor	10
Programming the Mitrion Virtual Processor	11
Software Development Cycle and Portability	11
Putting FPGA Accelerators to Work	12
Mitrion Applied	12
Genome Informatics	12
Internet Data Processing	12
Business Process Optimization	13
Finance	13
Computational Chemistry	13
Digital Content Creation and Image Processing	13
Random Number Generation	14
Evaluating Your Algorithms for FPGA Acceleration	14
Conclusion	16

Low Power Hybrid Computing for Efficient Software Acceleration

The good old days, when you could sit back and relax, waiting for a new generation of CPUs to effectively double the performance of your existing software are gone. Lately, CPU clock frequencies have reached a practical limit at around 3GHz, and your software needs a major rewrite to take advantage of the capabilities of today's many-core CPUs. The good news is you can turn this effort to your advantage by making use of software accelerators. These can provide an order of magnitude faster execution for certain computations, and be combined with traditional CPUs to form Hybrid Computing systems that deliver 10-100 times the performance of a non-accelerated solution.

Hybrid Computing

For decades, we have enjoyed constant performance improvements to our existing software as the clock speeds of CPUs in practice has been doubled every 18 months. However, higher clock frequency also means higher power consumption, and processor heat dissipation has put an end to free performance improvements with each new generation of semiconductor manufacturing processes.

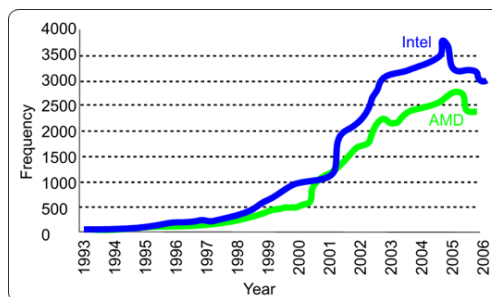


Figure 1 CPU Clock Frequency 1993-2006. Source: Tom's Hardware Guide, "The Mother of all CPU Charts"

CPU manufacturers now use the improved processes to fit more and more CPU cores onto each device, producing generations of many-core processors, each running at about the same clock frequency as their predecessors. This means that if you are looking for faster execution, you must actively seek alternative ways to speed up your software.

Accelerators and Hybrid Computing Systems

Purely sequential software will not see any further performance improvements, and will have to be rewritten exploit any available parallelism and distribute the computation among several processing resources – for example running on multiple CPU cores. However, the fact that the effort of making software run in parallel must be made opens up the playfield for new types of processing resources to complement the traditional CPU architecture.

Today, cost-efficient *accelerators* are available from several vendors as common off-the-shelf (COTS) products. Accelerators are specialized processors that can be used to speed up specific processing tasks. By combining accelerators with CPUs, you get a *hybrid computing* system, where each processing resource executes the parts of the software for which it delivers the best performance. This results in greatly increased system performance.

The main contenders for the COTS accelerator market are *Field Programmable Gate Arrays* (FPGAs) and *Graphics Processors* (GPUs). Unlike earlier accelerator technologies, such as vector processors or custom ASICs, these devices both have strong mass markets outside the high performance computing fields:

- Field Programmable Gate Arrays are integral parts of many advanced electronic devices and continuously evolved to cater to the needs of the developers of electronic products.
- Graphics Processors are used in most PCs and their development is driven by the requirements of the gaming market. When used for accelerated computing, these are referred to as general purpose GPUs, GPGPUs.

This allows cost-efficient high performance computing accelerators to be built, and ensures that the future development of the technology is well funded.

Processing Sweet Spots in Hybrid Computing

The performance gains available from hybrid computing are based on leveraging the strengths of many-core CPUs, FPGAs and GPGPUs used in tight integration. Compared to the two others:

- Many-core CPUs are an order of magnitude faster for command and control operations.
- FPGAs are an order of magnitude faster for non-floating point operations. They provide very good performance and power efficiency in processing integer, character, binary or fixed point data. They can also deliver competitive performance for complex floating-point operations such as exponentials or logarithms.
- GPGPUs are an order of magnitude faster on floating point operations.

Benefits of Hybrid Computing

Getting more compute performance from a server has a number of benefits. Most importantly, it reduces the number of servers needed for a specific workload. This means that Hybrid Computing delivers:

- Reduced system cost
- Greener computing and lower power cost, through lower power consumption and less energy spent on cooling
- Smaller system footprint

Alternatively, Hybrid Computing can deliver capacity for larger workloads within given budgets.

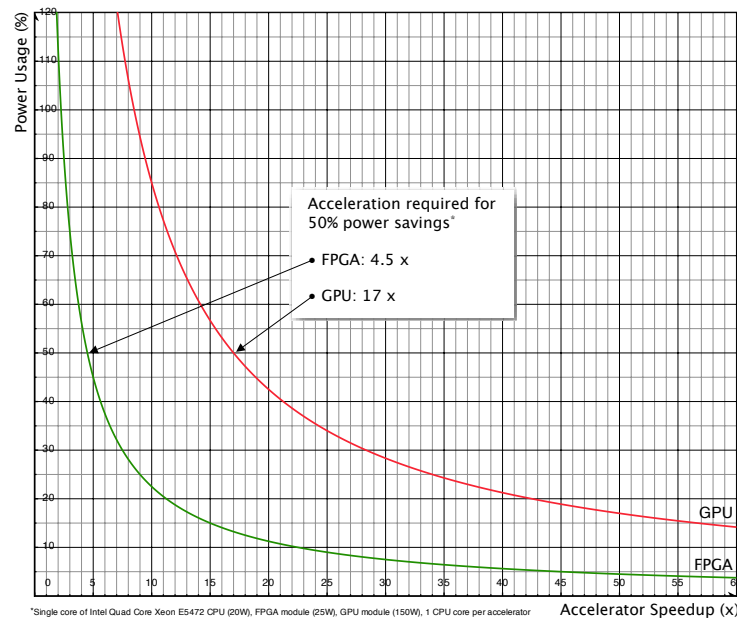


Figure 2 Power usage dependency on accelerator speedup.

FPGA devices typically use less power than CPUs and GPGPUs, and the remainder of this white paper will focus on Hybrid Computing using FPGAs. However, for power savings, the most important factor is the attained acceleration. It is therefore important to target the accelerator that provides the best acceleration for your algorithm.

FPGA-based Accelerators

Field Programmable Gate Arrays (FPGAs) are versatile configurable electronic components that are utilized in accelerators to implement tailored computational logic specific to the application being executed. In the hybrid computer system, the FPGA acts as a configurable co-processor to a CPU, allowing applications take advantage of application-specific hardware. The FPGA component can be reconfigured any number of times for new applications, making it possible to utilize the hybrid computer system for a wide range of tasks.

FPGA Basics

FPGAs can be used to build digital logic circuitry – including tailor made co-processors for a particular algorithm to be used as part of a hybrid computing system. Even though the P in FPGA stands for *Programmable*, it is important to realize that this is not in the software sense of programmable. For an FPGA it simply means that a circuit design can be loaded.

The bulk of an FPGA is made up from a large number of identical *configurable logic blocks, CLBs*. Each CLB consists of a number of *slices* which in turn consist of a number of (typically two or four) *logic cells* that can be configured to perform basic logic functions (such as **and**, **or**, **not**) on digital signals using the *lookup table, LUT*. The CLBs are interconnected through *programmable switch matrixes, PSM*, to form units that perform more complex functionality.

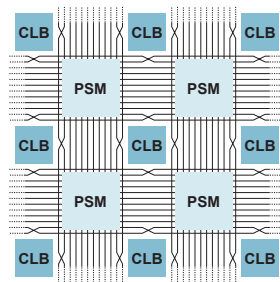


Figure 3 Organization of an FPGA

FPGAs also provide internal RAM memory banks and specialized multiplier logic blocks, *Multiply-accumulate circuits, MACs*, for efficient multiplication and addition. FPGAs may have other block with specialized functions to serve purposes such as digital signal processing.

Configuring the FPGA

Before the FPGA can perform any functionality whatsoever, all the configurable elements must be set up. This is done by “programming” the FPGA by uploading a binary configuration file known as a *bitstream* or *bit file*. This file holds the configuration settings for each CLB, PSM, MAC, I/O and other configurable element of the FPGA.

Synthesis, Place and Route

To create a bitstream that configures the FPGA to perform a certain function, a circuit design for the electronic circuit that is to perform the functionality is needed. This design is created in a *hardware description language, HDL*, such as *VHDL* or *Verilog*, or, at a slightly higher level of abstraction, using any of numerous *Electronic System Level, ESL* design tools. Alternatively, the Mittrion Software Acceleration Platform, offered by Mittrionics, is capable of producing the circuit design in VHDL from programs written in the high-level Mittrion-C programming language.

The HDL circuit design is run through a process known as *synthesis*, where the HDL is translated into interconnected basic logic functions. The output from the synthesis is fed to *place and route* software. The place and route process maps the logic functions to the configurable logic on the FPGA and calculates how the logic should be interconnected to meet constraints such as timing of electric signals between logic cells.

FPGAs Compared to Microprocessors

FPGAs represent a computer architecture ideally suited to exploiting parallel execution. They are able to run some algorithms 10-100 times faster than a microprocessor. It is, however, important to note that:

- An FPGAs is *not* a better microprocessor, nor a microprocessor substitute.
- FPGAs will *not* accelerate existing applications without significant porting efforts, and not all algorithms are suitable for FPGA acceleration.

Compared to microprocessors, FPGAs have a couple of major performance disadvantages:

- The maximum clock frequency for FPGAs is a few hundred MHz, while microprocessors run at a few GHz.
- The FPGAs configurability comes at the cost of a large overhead, leaving 10-100 times less logic available to the user compared to a microprocessor of similar size.

The reasons FPGAs are still able to outperform microprocessor for some algorithms are:

- The FPGA is used to implement a circuit specialized for a specific task.
- All the logic on the FPGA can be utilized to perform that task.
- FPGAs deliver vast amounts of parallelism.
- FPGAs offer huge memory bandwidth through configurable logic, block RAM and local memories. See figure below.

FPGA	Microprocessor
Configurable logic	On Chip Registers
<ul style="list-style-type: none"> • TB/sec • 10's KB 	<ul style="list-style-type: none"> • 100's GB/sec • 1-10's KB
Internal RAM	L1-L3 Cache
<ul style="list-style-type: none"> • 100's GB/sec • 100's KB 	<ul style="list-style-type: none"> • 100's GB/sec • 1-10's KB
Local Memories	
<ul style="list-style-type: none"> • ~10GB/sec • 10-1000's MB 	
System Memory	System Memory
<ul style="list-style-type: none"> • 100-1000's MB/sec • Terabytes 	<ul style="list-style-type: none"> • 100-1000's MB/sec • Terabytes

Figure 4 Memory hierarchies of FPGA and microprocessor systems.

Power Consumption

The power consumption of FPGAs compare very favourably to microprocessors. A large FPGA running at full speed typically uses less than 25 W, compared to 100 W or more for a modern microprocessor. Even at moderate acceleration, a FPGA system will use dramatically less electric power than an all microprocessor system with the same performance.

FPGA-based Hybrid Computer Systems

FPGA-based hybrid computing systems are typically built using *FPGA modules*. These FPGA modules are connected to a hosting computer system through a variety of system bus architectures, depending on the system and module. In addition to control and interface logic, the modules hold one or more FPGAs for user algorithms and usually have local memory attached directly to the FPGAs. Some modules allow the FPGA to read and/or write directly from or to the host system's memory.

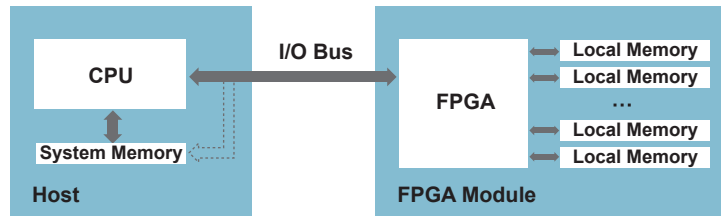


Figure 5 FPGA module in a hybrid computer system

Applications control the FPGA modules through an *Application Programming Interface, API*, to perform tasks such as:

- Upload bitstreams to the FPGA.
- Start and stop algorithm execution on the FPGA.
- Transfer data to and from the FPGA and/or locally attached memories.

Developing Accelerated Applications

Development of FPGA-accelerated software adds two steps to the software development cycle:

- Deciding on the partitioning of the application between the host CPU and the FPGA.
- The design and implementation of the tailored co-processor circuit that will run the accelerated algorithm on the FPGA.

The Mitrion Software Acceleration Platform addresses the later through the *Mitrion Virtual Processor*, a design for an adaptable parallel processor capable of efficiently running software on the FPGA.

With the Mitrion Platform, the steps of deploying FPGA-acceleration to an application are:

- Identify the computationally intense routines of the application and decide on a partitioning of tasks between CPU and FPGA.
- Rewrite the routines that are to run on the FPGA in the high level Mitrion-C programming language.
- Replace the routines in the original program with calls to the FPGA.
- Use the Mitrion Software Development Kit to debug the Mitrion-C routines and their interaction with the host program.
- Use the Mitrion SDK and FPGA vendor tools to generate configuration files for the FPGA.
- Install and run on target system.

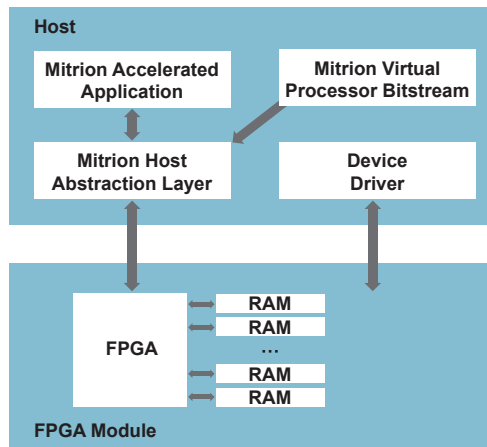


Figure 6 Running accelerated applications with the Mitrion Software Acceleration Platform

Algorithm Development for FPGAs

Hardware Design

As mentioned in the previous section, accelerating algorithms on FPGAs requires that a circuit is designed to perform the algorithm on the FPGA. Since FPGAs originated as versatile components for implementing electronic hardware, numerous hardware design tools are available to hardware designers specialized in creating circuits for FPGAs. They can choose to develop their design in VHDL, Verilog or using an ESL design tool.

Regardless of the tools chosen, hardware design requires a solid knowledge of electrical engineering and the understanding of subjects of FPGA circuit design such as gates, wires, CLBs, multipliers, signal timing and clocking. From the perspective of software development, there is a big difference between the software source code (an instruction stream for a processor), and the gates and wires of the FPGA, making it a challenge to map software algorithms to the FPGA.

The Mitrion Virtual Processor

The Mitrion Software Acceleration Platform has been developed for the sole purpose of allowing software developers to benefit from FPGA-based software acceleration, without having to deal with the complexities of hardware design.

What makes the Mitrion Software Acceleration Platform unique is that it introduces a processor as an abstraction layer, the *Mitrion Virtual Processor* (MVP) between the FPGA hardware and the software that is to be run. While other programming solutions for FPGAs attempt to produce circuit designs directly from a high level language, Mitronics adds the MVP, which executes the software on the FPGA. The benefit is that the processor gives a complete separation of software from hardware, and the software developer is isolated from all aspects of FPGA hardware design. With the MVP, all the circuit design is already taken care of and is delivered as part of the Mitrion SDK.

Programming the Mitrion Virtual Processor

The Mitrion SDK includes a compiler for Mitrion-C, a C family programming language used to program the Mitrion Virtual Processor. Mitrion-C is a high level programming language that is designed to let programmers take advantage of ultra-fine grained parallelism and synchronization together with tightly coupled microprocessor/FPGA co-processing. At the same time, it hides all aspects of FPGA circuit design.

The performance of a Mitrion Virtual Processor programmed in Mitrion-C is comparable to hand designed circuits at the same clock speed.

Software Development Cycle and Portability

A major advantage of the Mitrion Platform is that the development of accelerated applications follows a software life cycle. When porting to other FPGA-based hybrid computer systems, the programmer will only need to change the details pertaining to the specifics of the machine organization. This lets users immediately exploit FPGA evolution with regard to sizes, speeds, and on-chip memory.

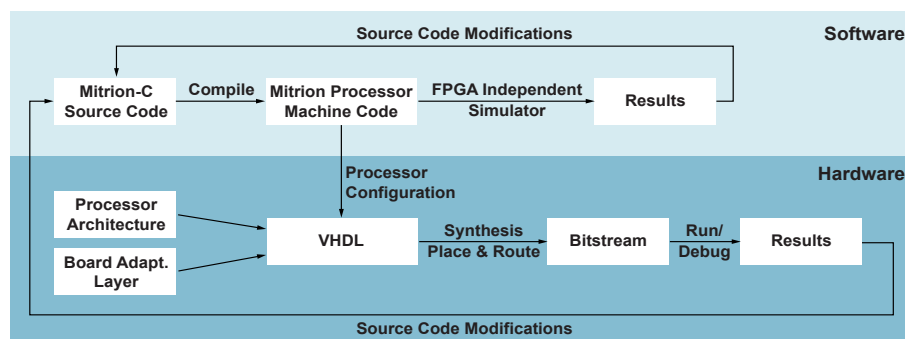


Figure 7 Mitrion-C development cycle

Putting FPGA Accelerators to Work

In hybrid computing systems, FPGA accelerators provides very good performance for applications that process integer, character or bit data. They can also provide exceptional performance for more complex floating point operations such as exponentials and logarithms.

Mitrion Applied

Mitronics has successfully deployed the Mitrion Virtual Processor to accelerate applications and algorithms within a number of areas.

Genome Informatics

Today, some genome computing tasks take days, weeks or even months. In the future, the amount of data to process will increase while at the same time, these task must be completed in seconds or minutes in order to be used in medical decisions.

The challenge is to shorten genome computing processes by several orders of magnitude, while at the same time keeping system size and power consumption.

Genome data is typically encoded using two bits per nucleotide base pair. The Mitrion Virtual Processor supports flexible data types that allow highly efficient processing of the nucleotide data. The Mitrion Platform is used to accelerate applications and algorithms such as:

- **NCBI BLAST-N** – NCBI BLAST is the most used sequence alignment application available. Within the framework of the Mitrion-C Open Bio Project, Mitronics has accelerated the BLAST-N nucleotide search algorithm to provide up to 60 times the performance of an unaccelerated solution.
- **Phylogenetic tree** – Evolutionary history is a central problem in genomic research. With Mitrion, Markov Chain Monte Carlo Simulations for phylogenetic tree research have been accelerated by a factor of 15.

Internet Data Processing

Internet search engines, as well as new services such as video-on-demand and Internet telephony, adds to the demand for ultra-scale data centres to power these services. It is a challenge to increase power efficiency of ultra-scale data centres to allow for introduction of new services and scaling to more users. FPGAs have been proven efficient at processing large amounts of text based data. Applications could be search, filtering, character set conversions etc.

- **Text search** – A simple text search algorithm has been implemented on the Mitrion Virtual Processor that is capable of simultaneously searching for any of 4000 strings in stream of data corresponding to a 10Gbit Ethernet link.
- **Grep** – The Mitrion Virtual Processor has been used to accelerate the standard UNIX grep function, providing FPGA-accelerated regular expression searches.

Business Process Optimization

Today, many businesses have workflows that require technology beyond standard enterprise solutions. This can involve applications that operate on very large datasets, for example data mining large databases, or that are very computationally intensive, for example performing statistical modelling or process simulations. Some applications, such as market feed processing, need to perform advanced operations on complex data in near real-time. The Mitrion Virtual Processor can be used to expand the availability of high performance computing resources throughout organizations and deliver the performance required for critical workflows within constrained power envelopes and server footprints.

Finance

Within the finance sector, the Mitrion Virtual Processor makes it easy to implement high-performance fixed and floating point binary coded decimal arithmetic, required by law for some financial applications. The Mitrion Virtual Processor also delivers massive performance for exponentials and logarithms required for calculations such as Black-Scholes option pricing.

- **Black-Scholes** – The Mitrion Virtual Processor has been successfully demonstrated running Black-Scholes option pricing, producing 100M results per second on a single FPGA.

Computational Chemistry

Computational Chemistry algorithms that depend on computing bit-patterns are well suited for the Mitrion Virtual Processor.

- **Tanimoto coefficients** – Determining the similarity between two molecules is facilitated by calculating molecular fingerprints that represent the composition and structure of the molecules. The fingerprints are binary sequences in which each bit reflects some aspect of the molecule. The similarity can be measured using Tanimoto Coefficients, calculated by counting and comparing bits set in the molecular fingerprints. With the Mitrion Virtual Processor, databases of all chemicals known in the world can be searched in seconds.

Digital Content Creation and Image Processing

The efficient bit-manipulation capability of the Mitrion Virtual Processor makes for the creation highly efficient accelerated compression algorithms and codecs.

- **Discrete cosine transform (DCT)** – Discrete Cosine Transform is a core part of many audio, video and image compression algorithms, such as JPEG, MPEG and DV. The Mitrion platform has been demonstrated to accelerate the DCT by an order of magnitude.
- **Rice coding** – Rice coding can be used for lossless compression of audio and video data. Rice coding on the Mitrion Virtual Processor has been shown to attain a speedup of 7x.
- **Convolution** – In image processing, convolutional filtering is a common operations, for example in edge detection or gaussian blurring. The Mitrion platform comes with example code that performs efficient image convolution.
- **Thin plate splines** – Thin plate splines is an interpolation method that finds a smooth surface that passes through all given points. It is used for image alignment and shape matching. The Mitrion Virtual Processor provides a 10-fold acceleration over non-accelerated solutions.

Random Number Generation

High quality random numbers are required by many algorithms. The Mitrion Virtual Processor can be used to efficiently implement a wide range of pseudo-random number generators.

- **Mersenne twister** – The Mersenne twister algorithm has been implemented to provide 800 million uniform or Gaussian random numbers per second while at the same time leaving plenty of room for other algorithms on the FPGA

Evaluating Your Algorithms for FPGA Acceleration

To determine the potential for acceleration, your candidate algorithm must be carefully analysed. Important factors determining the acceleration potential are:

- Does the application have one, (or possibly a few) key routines which dominate computing time?
- Computational intensity: is there enough work per word of data to offset the overhead to move the data to and from the FPGA module?
- Is the data organized in a way that allows it to be transferred to and from the FPGA module without too much data marshalling overhead?
- Do the accelerated parts of the application require floating point arithmetic or transcendental math functions?

In determining the expected performance it is important to realize that only parts of the application is being accelerated. Some share of the execution time will be spent on the non-accelerated parts of the application running on the host CPU. Also, implementing acceleration will introduce additional overhead for data transfers to and from

the FPGA, data re-ordering and re-formatting. The illustration below shows the actual acceleration obtained if 90% of the execution time can be cut by a factor 10 using the FPGA, and assuming a 5% overhead copying data to and from the FPGA.

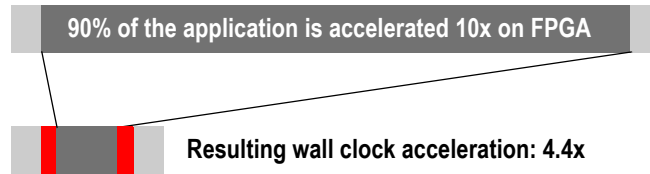


Figure 8 Wall clock acceleration compared to FPGA acceleration.

The application speed-ups will follow Amdahl's law. The diagram below illustrates the attainable total speed-ups by accelerating a certain percentage (out of the execution time) of an application.

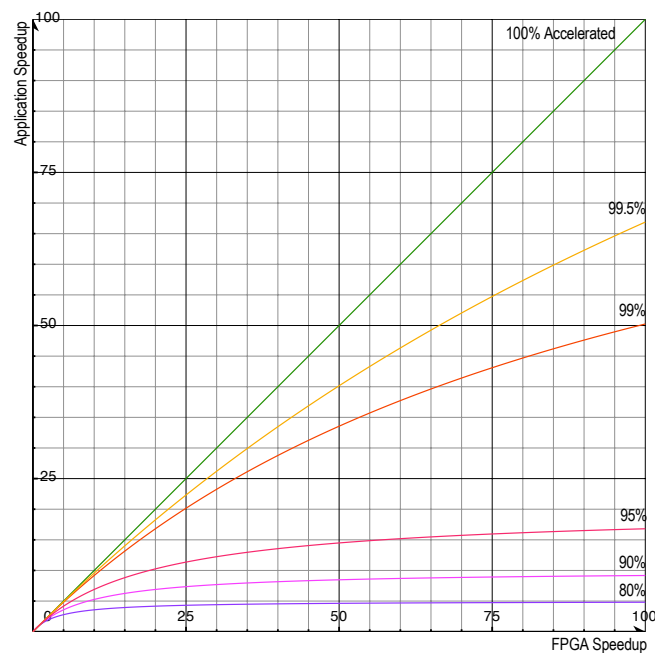


Figure 9 Total application speed-up related to FPGA speed-up.

Conclusion

Today, new COTS accelerators based on GPU and FPGA technology are readily available from several vendors. These can be used to build highly efficient systems that significantly reduce power cost and server footprint. For applications that process integer, text and binary data, FPGAs deliver a low power solution, and with the Mitrion Software Acceleration Platform, software development for these FPGA accelerators does not require any hardware design skills.

For more information on FPGA-based hybrid computing and details of Mitrionics Hybrid Computing Development Systems, please see Mitrionics website:

www.mitrionics.com