Convey MX Series

Architectural Overview
High-performance Analytics

High performance computing (HPC) grew out of the need for scientists and engineers to observe phenomena that is either too fast or too slow, or too big or too small, to observe empirically. The general rule in HPC is “the more compute power, the better the answers.” Today however, there is a new and growing class of HPC-like problems that have grown from being computationally intensive to being data intensive—affectionately known as "big data" problems. Challenges include problems in genomics, graph analytics, social network analytics, fraud detection, and security.

Convey highlights a subset of big data applications as "high-performance analytics" (HPA)—problems that are irregular in nature, usually require results in realtime, and deal with large volumes of data. HPA problems are the antithesis of those that are computationally-intensive. Many HPA algorithms use pointer-based data structures such as unbalanced trees, unstructured grids, and graphs. As a result, they tend to have a much lower compute-to-memory reference ratio, and memory accesses often dominate performance (or lack thereof).

HPA algorithms also tend to randomly access large amounts of physical memory. Any time a data set can be resident in physical memory (as opposed to on secondary storage) the performance of the application will increase. However, randomly accessing that memory is a performance-killer for today's commodity systems. Not only is memory bandwidth limited, but the number of parallel operations that can be “in flight” is also limited.

Finally, there is often little computation involved in data-intensive applications. Data types might include floating-point, integer and character, and operations are often simple compares or references to other data (pointers). The complexities of memory reference patterns combined with the “lightness” of computation makes data-intensive computing extremely inefficient on current commodity system architectures.

Solution Characteristics

Several challenges arise when developing architectures that will economically and efficiently solve data intensive problems. First, power limitations limit future increases in clock rates of commodity processors—meaning performance of a single thread of general-purpose instructions (e.g. the x86_64 instruction set architecture) has essentially flattened. That means that future computational models must leverage a larger number of computational units.

Secondly, a different approach has to be taken to address the disparity between processor performance and memory latency. Today’s general-purpose processors operate in the ~3GHz range—much faster than typical DRAM-based memory
subsystems. Computer architects have gone to great lengths to develop memory hierarchies to circumvent this disparity (e.g. multi-level caches, prefetching, etc.). However, the memory subsystem is the primary roadblock to increased performance of data-intensive applications.

To summarize: architecture elements that support data intensive computing include:

• Large, randomly accessible and latency-tolerant memory
• Support for many (000’s) of threads
• Fine-grained synchronization (Full/Empty tagged memory)
• High aggregate memory bandwidth
• Scalability of memory and processing elements

**Hybrid-Core Computing**

Convey introduced the world's first hybrid-core computer system, the HC Series, in 2008. Since then, Convey customers worldwide have employed hybrid-core systems to accelerate their most critical applications well beyond the performance levels available from commodity systems. Now Convey is launching another industry first—a smarter computer architecture tailored to the growing needs of big data applications. These types of applications demand a computing architecture that efficiently handles extremely large, hard to partition problems.

Hybrid-core computing extends a commodity instruction set architecture (e.g. x86_64) with application-specific instructions to accelerate application performance. Convey hybrid-core computers tightly integrate an FPGA-based, reconfigurable coprocessor with industry standard Intel® based systems to enhance performance without sacrificing the flexibility and ease of use of a general-purpose system (Figure 1). When used as nodes in a high performance computing cluster, Convey systems deliver higher performance for a given number of nodes, providing substantially better performance per dollar or watt than conventional clusters.

The Intel processors (the "host" system) run industry-standard Linux, support standard networking and interconnect fabrics, and, in a clustered environment, make the entire system appear as any other commodity server. The coprocessor’s FPGAs execute specific operations (called personalities, see below) that represent a large component of an application’s runtime, reducing time-to-solution for the entire application.

Figure 1. A hardware view of hybrid-core computing.

The ability to support different instruction sets in a common hardware platform allows the implementation of new instruction sets in months instead of the years required to design and introduce a new application-specific integrated circuit. The huge reduction in implementation time makes it practical to develop instruction sets tailored to specific applications and algorithms. Convey provides personalities for key application areas in industries like life sciences, governmental programs, and "big data" analytics, and provides the ability for customers to create their own personalities tailored to their particular applications.
An application executable contains both Intel and coprocessor instructions, and those instructions execute in a common, coherent address space (Figure 2). Coprocessor instructions can therefore be thought of as extensions to the x86 instruction set—they execute in the same address space and on the same data as x86 instructions.

**Figure 2. Hybrid-core computing as it appears to an application.**

**MX Architecture**

The MX™ Series is a new computer architecture that directly addresses the computing needs of high-performance analytics applications. The Convey MX Series is a smarter computer architecture tailored solving extremely large, irregular and hard to partition problems.

The new architecture features the capability to run tens of thousands of threads of execution coupled with a smart memory system that can atomically perform “in-memory” arithmetic operations. Architected to scale up to 32 Terabytes of physical memory, the MX Series is designed to run big data applications that can benefit from massive parallelism and large physical memory.

The MX architecture is based on the original hybrid-core computing pioneered by Convey. The first MX Series system, the MX-100, is shown in Figure 3. Key components of the MX architecture are:

- An Intel processor based host system,
- An FPGA-based reconfigurable coprocessor,
- A highly parallel, latency-tolerant memory subsystem on the coprocessor,
- Globally shared memory,
- “Personalities”—instruction sets developed to accelerate individual algorithms within applications.

Each of these is described in more detail below.

**Figure 3. Convey MX-100 architectural overview.**
Hosts and Host Operating System
A key component of hybrid-core computing is the integration of a complete x86 ecosystem. The host subsystem presents a standard interface to other components of an installation's infrastructure—including a standard operating system, interconnect fabrics, job scheduling software, and development tools.

As such, a standard host server is tightly coupled (see "HCGSM" above) to the FPGA-based coprocessor. A selection of host servers are available, depending on application needs and are described in "Appendix A. HC-2 Host Specifications."

The hosts utilize CentOS (currently 6.2) with some modifications. From centos.org: "CentOS is an Enterprise Linux distribution based on the freely available sources from Red Hat Enterprise Linux. Each CentOS version is supported for 10 years (by means of security updates). A new CentOS version is released approximately every 2 years and each CentOS version is periodically updated (roughly every 6 months) to support newer hardware. This results in a secure, low-maintenance, reliable, predictable and reproducible Linux environment."

Convey makes some changes to CentOS to support the HCGSM interface, larger page sizes, the data mover, and the coprocessor runtime interface.

The MX-100 Coprocessor
The coprocessor has three major sets of components, referred to as the Application Engine Hub (AEH), Application Engines (AEs), and the Memory Subsystem.

Application Engine Hub (AEH)
The AEH is the central hub for the coprocessor. It implements the interface to the host processor, routes memory requests from the host processor to the appropriate coprocessor memory locations, fetches and decodes instructions, and executes Convey canonical instructions. Canonical instructions are executed in the AEH, while extended instructions are passed to the AEs for execution.

Reconfigurable Application Engines (AEs)
The reconfigurable AEs are the heart of the coprocessor and implement the extended instructions that deliver performance for a personality. There are 4 AEs, connected to the AEH by a command bus that transfers opcodes and scalar operands, and connected via a network of point-to-point links to each of the memory controllers. Each AE instruction is passed to all four AEs. How they process the instructions depends on the personality.

While the clock rate for the FPGAs used to implement the coprocessor is lower than that of a commodity processor, each AE will have many functional units operating in parallel. This high degree of parallelism is the key to the coprocessor's performance. By implementing just those operations that are needed for a particular workload, many more of those units can be packed into each chip.

Memory Subsystem
The Convey MX memory subsystem complements the coprocessor's highly parallel reconfigurable computing elements, and has the following key features:

- A large physical memory that allows thousands of concurrent access from the computational elements
- Cacheless, doubleword (64-bit) optimized design to efficiently support random (scatter-gather) transfers
- Hybrid-core Globally Shared Memory (HCGSM) that supports virtual-to-physical translation and provides an identical (coherent) view of memory between the host and coprocessor
- Coprocessor atomic operations that provide single instruction read-modify-write capabilities for lightweight test and set operations
- Coprocessor full/empty tag bits that provide support for locking or unlocking individual 8 byte memory locations with a single instruction
Memory Crossbar and Memory Controllers

Within a node, the coprocessor memory system implements a distributed crossbar over a network of point-to-point links. Each Application Engine (AE) FPGA is connected to each Memory Controller (MC) by four full duplex channels, each consisting of a pair of high speed serial links. An additional set of links connects each MC to the Host Interface. The MCs translate virtual to physical addresses on behalf of the AEs, and route references to host memory to the host via the AEH.

Most modern microprocessors transfer whole cache lines to and from memory, and waste bandwidth when executing programs that perform non-unit strides or random accesses to individual words. The degradation can be as much as 8x for an application that only uses 8 bytes from a 64-byte cache line. In contrast, the Convey memory system operates on individual 8-byte doublewords. The Scatter-Gather DIMMs are optimized for transfers of 8-byte bursts, and provide near peak bandwidth for non-sequential 8-byte accesses. The coprocessor therefore not only has a much higher peak bandwidth than is available to commodity processors, but also delivers a much higher percentage of that peak for non-sequential accesses.

Together the AEH and the MC's implement features that are present in all personalities. This ensures that important functions such as memory protection, access to coprocessor memory, and communication with the host processor are always available.

Memory requests from the AEs use virtual addresses, and are queued on the appropriate MC link interface. Each AE supports 1024 outstanding requests to each MC, for a total of 32,768 (2^15) outstanding requests across all four AEs. The system as a whole supports 16Gt/s (billions of doubleword transfers per second) between the AEs and the Memory Controllers.

Each MC supports 4 Scatter-Gather DIMM slots, for a total of 32 SG-DIMMs. Depending on memory technology employed, a DIMM slot can be populated with 2, 4, 8, 16, or 32 GB DIMMs, resulting in 64, 128, 256, 512, and 1,024 GB memory configurations respectively.

Synchronization Primitives

Because of the multi-threaded architecture of the MX systems, it is important to have very lightweight, fine-grained synchronization primitives. Such primitives permit thousands of concurrent threads of operation to increase performance and minimize the effects of memory latency. The MX architecture implements two different kinds of instructions: atomic memory operations and memory-based synchronization via full/empty tag bits.

Atomic memory operations. Atomic memory operations are operations that allow indivisible in situ operations on a specific memory location, without the requirement of a typical lock-load-<operation>-store-unlock sequence. These operations are accessible from both host and coprocessor code regions via compiler intrinsics. Atomic memory operations guarantee a thread’s ability to perform an operation on memory without colliding with another thread.

Full/empty tag bits. Full/empty tag bits are a memory-controller level implementation of the classic producer-consumer construct. In addition to its native storage, each 8-byte doubleword in coprocessor memory has a tag-bit. This tag-bit indicates the locked, or full, and the unlocked, or empty, state of the associated address. Memory operations have the ability to read or write and lock or unlock the tag-bit with single instructions. The result is very fine-grained, hardware-based producer-consumer synchronization. Memory requests that utilize full/empty tag bits are accessible from host and coprocessor code regions via compiler intrinsics.

Hybrid-core Globally Shared Memory (HCGSM)

An important part of any heterogeneous architecture is the ease of developing applications. The ideal programming model is one in which applications are free from the intricate details of the underlying hardware—generally, a "load/store" programming
model is the easiest target development environment. Application developers can simply reference data constructs naturally, without worrying about using I/O commands or other constructs to communicate with the coprocessor. In addition, by using standard virtual memory translation mechanisms, a process (either on the host side or the coprocessor side) is securely constrained to its own address space, and can’t corrupt or disable memory outside of that address space.

The Convey hybrid-core systems feature a globally addressable shared memory (HCGSM) architecture. Although there are two physical memory subsystems (one for the multi-processor host, and one for the multi-AE coprocessor), all memory is accessible to all compute elements. Application code in both the host and the coprocessor deals with virtual addresses, and the system hardware is responsible for determining the physical location of that memory. Code generated by the compilers on the x86 host system can assume a load/store model, making code generation easier and more efficient.

Given the NUMA memory architecture, data locality, and thus data movement between nodes, becomes a critical performance issue. A data mover engine is built into the coprocessor and used for data migration, page zeroing, and data movement between host and coprocessor. Data mover functions are accessible via library calls or compiler directives.

HCGSM in the MX system is physically implemented via a PCIe Gen2 connection, and utilizes the MMIOH function of the x86 processor to map coprocessor memory in the host physical address space. The use of PCIe as the interface allows for flexible choice of hosts, giving customers the flexibility to match the system configuration to the application requirements.

Programming Model
Acceleration of applications on the Convey hybrid-core systems is by means of reloadable personalities—hardware bitfiles that describe performance-critical parts of an application directly in hardware. Personalities are custom instruction set architectures (ISAs) that are extensions to the x86 ISA.

Personalities
A personality is the custom logic that resides on the coprocessor and implements the extended instruction set tailored to accelerate a given algorithm. Personalities include a precompiled FPGA bit files, a description of the machine state model sufficient for the compiler to generate and schedule instructions, and an ID used by the application to load the correct image at runtime. A system can contain multiple personalities that can be dynamically loaded, but only one personality is loaded at any one time. Each personality supports the entire canonical instruction set, plus extended instructions that may be unique to that personality. Extended instructions are designed for particular workloads, and may include only the operations that represent the largest portion of the execution time for an application.

All personalities have some elements in common, however:

• Coprocessor execution is initiated and controlled via instructions.
• All personalities use a common host interface to dispatch coprocessor instructions and return status.
• Coprocessor instructions use virtual addresses and coherently share memory with the host processor. The host processor and I/O system can access coprocessor memory and the coprocessor can access host memory. The virtual memory implementation provides protection for process address spaces as in a conventional system.
• All personalities support the canonical instruction set, and the Convey compilers assume that the canonical instructions can be generated and executed.

These common elements ensure that compilers and other tools can be leveraged across multiple personalities, while still allowing customization for different workloads. A
personality therefore implements a computer architecture customized for a particular type of algorithm or workload.

**Custom Personalities**

Many emerging applications in fields such as bioinformatics and data analytics incorporate algorithms that don’t fit classic architectural models. Convey provides a Personality Development Kit (PDK) that allows custom instructions to be implemented to support such applications. Custom instructions might be as simple as a single instruction that instructs the AEs to process a large in-memory data structure.

The PDK includes logic blocks that implement the interfaces between the AEs and the other components of the coprocessor, tools to package bitfiles produced by the Xilinx FPGA development tools into a personality, a simulator for debugging, and system and compiler APIs to allow execution of user-defined instructions. The custom logic is loaded only in the AEs, however the system interface, virtual memory, and canonical instruction set implementation in the AEH and Memory Controllers is unchanged (Figure 4).

![Figure 4](image.png)

Figure 4. The PDK includes logic for memory, management, and dispatch functions.

An architected set of instructions transfer data between the canonical register set in the AEH and the AEs, and initiate execution by the custom logic. This allows the user to develop FPGA logic that executes within a process address space using virtual memory addresses. The FPGA logic therefore can operate directly on the same data structures created by the application code running on the host processor. Other processes and system memory are protected from invalid accesses by the virtual memory protection system.

The Convey Personality Development Kit supports a stepwise process that simplifies the process and ensures correctness (Figure 5)

1. Identify key algorithms or areas of an application that consume the majority of compute time, which can be as small as a few loops, or as large as a whole routine that implements a computational kernel.
2. Develop a functional model that represents the behavior of the instruction. This model is based on the original application source code and is called from the application as if it were a hardware instruction.
3. Implement the instructions as logic design using Verilog or VHDL, using the logic libraries included in the PDK to connect to the other hardware components in the coprocessor. The logic design can then be verified using the Convey system simulator to drive an external logic simulator.
4. The application can now be compiled using the Xilinx ISE tools and packaged as a personality to be run on the Convey hybrid-core system.
The ability to create new instructions that can be loaded dynamically allows the implementation of highly parallel instructions that are specific to algorithms and data structures not well served by classic scalar and vector architectures. As these applications evolve, the instruction set architecture can evolve along with new instructions.

**Turnkey Applications and Personalities**

The Convey hybrid-core architecture lends itself well to development and distribution of personalities that accelerate popular algorithms across a wide range of disciplines. Both Convey, Convey customers, and 3rd party developers have developed personalities that are sold and supported to accelerate applications. Specifically, personalities have been developed for four major market segments: Bioinformatics, "Big Data" Analytics, Government/Defense, Research and Development. Please check the Convey website for the latest information on personalities that are available.

**Convey Hybrid-core OpenMP (CHOMP) Programming Environment**

One of the personalities supported on the MX architecture is CHOMP (Convey Hybrid-core OMP). CHOMP is a personality family that encompasses support for the OpenMP programming model and API (Figure 6). OpenMP is ideally suited for accelerating applications that can be highly parallelized.

The CHOMP personality family consists of multiple levels of software hierarchy. At the basis of the CHOMP software hierarchy is the CHOMP RISC instruction set. Most users will write application code in a high-level language such as C or C++. However, it is definitely permissible to write optimized functions and/or application kernels in CHOMP ISA assembly language.¹

The next level in the software hierarchy is the Convey Lightweight Runtime Library, or CLR library. The CLR library is designed to function as a machine-level runtime library that permits exacting control over the dispatch, runtime, locality and degree of parallelism within the target CHOMP personality. The CLR library is analogous to the POSIX Thread library [Pthreads]. It is designed for users seeking to develop other high-level programming models and/or runtime interfaces for CHOMP.

¹For more info on the CHOMP instruction set, refer to the CHOMP Architecture Reference document.
The Convey compiler suite serves as the next level in the CHOMP software hierarchy. The compiler has the ability to generate dual-target code for both the x86_64 instruction set and the CHOMP RISC instruction set within the same object file. This ability to generate multiple target code streams permits the Convey-generated binaries to be both high performance and highly portable.

The final level in the software hierarchy is the high-level runtime and programming model. Convey has developed a runtime library based upon the OpenMP 3.1 programming model and runtime specification that successfully implements the entire specification on both the host and CHOMP instruction sets. This hybrid OpenMP permits the user to develop very portable versions of the same applications kernels that scale on both commodity x86_64 platforms and Convey hybrid core platforms.

Summary
The MX Architecture is ideally suited for highly threaded parallel applications that require access to a large, shared memory. Such applications typically do not run well on clusters of commodity servers.

Convey hybrid-core computer systems provide increased scalability and cost effectiveness by delivering higher performance per node for compute intensive workloads. They leverage reconfigurable logic to support specialized architectures optimized for specific workloads. These specialized architectures are integrated into an industry standard Intel 64 system—leveraging commodity components and allowing easy integration into an existing environment.

The Convey systems maximize productivity by delivering prebuilt personalities for important applications and a unified development environment based on standard ANSI C/C++ and Fortran. Users can create new personalities, allowing new instruction sets to be innovated for emerging applications.
### Appendix A. MX Host Specifications

<table>
<thead>
<tr>
<th>System Model</th>
<th>Coprocessor Options</th>
<th>Host Options</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>MX-100</td>
<td>4x Virtex®-6 HX-565T</td>
<td>16 Slots 1/2/4 GB SG-DIMMs (64 GB max)</td>
<td>Host-110 2x Intel® Xeon® Processors X5670 6-core 2.93GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>12 Slots 4/8/16 GB DDR3 DIMM (192 GB max)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>6x 2.5&quot; SATA disk 1 Intel IOM 3U 1570 W Redundant Power Supplies</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Host-210 2x Intel® Xeon® Processor E5-2609 4-core 2.4GHz</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>24 Slots 4/8/16/32 GB DDR3 DIMM (768 GB max)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>8x 2.5&quot; SATA disk 1 PCIe 3.0 x16 1 Intel IOM 3U 1670 W Redundant Power Supplies</td>
</tr>
</tbody>
</table>

### Appendix B. MX-100 Specifications

<table>
<thead>
<tr>
<th>System Model</th>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
</table>
| MX-100       | Dimensions | • 5.19 inches (131.8 mm) high  
• 16.93 inches (430 mm) wide  
• 27.25 inches (692 mm) deep  
• 48 pounds (21.8 kg) - max chassis weight |
|              | Operating Temperature | 10°C to 35°C (50°F to 95°F) |
|              | Storage Temperature | -40°C to 70°C (-40°F to 158°F) |
|              | Humidity | 10 to 90% non-condensing |
|              | Operating Altitude | -16 to 1800m (-50 to 5,900 feet) at 35°C inlet  
1,800 to 3,048m (5,901 to 10,000 feet) @ 30°C inlet |
|              | Storage Altitude | -16 to 10,600m (-50 to 35,000 feet) |
|              | External Interfaces | Rear Panel  
• PCIe x8 Connector  
• RJ-45 10/100 Ethernet Port  
• RJ-45 Serial Port  
• Two IEC320-C14 Power Plugs |
|              | LEDs and Controls | Front Panel  
• Reset Switch  
• Chassis ID  
• Power OK Indicator  
• Co-Processor Status  
• 4 Fan Fail LEDs  
Rear Panel  
• 5 LED group "port good" indicator  
• 4 LED group status indicator |

‘32 GB DIMMS available 4Q2012