Instruction Set Innovations for Convey's HC-1 Computer

THE WORLD'S FIRST HYBRID-CORE COMPUTER.

Hot Chips Conference 2009

tbrewer@conveycomputer.com

Introduction to Convey Computer

- **Company Status**
  - Second round venture based startup company
  - Product beta systems are at customer sites
  - Currently staffing at 36 people
  - Located in Richardson, Texas
- **Investors**
  - Four Venture Capital Investors
    - Interwest Partners (Menlo Park)
    - CenterPoint Ventures (Dallas)
    - Rho Ventures (New York)
    - Braemar Energy Ventures (Boston)
  - Two Industry Investors
    - Intel Capital
    - Xilinx

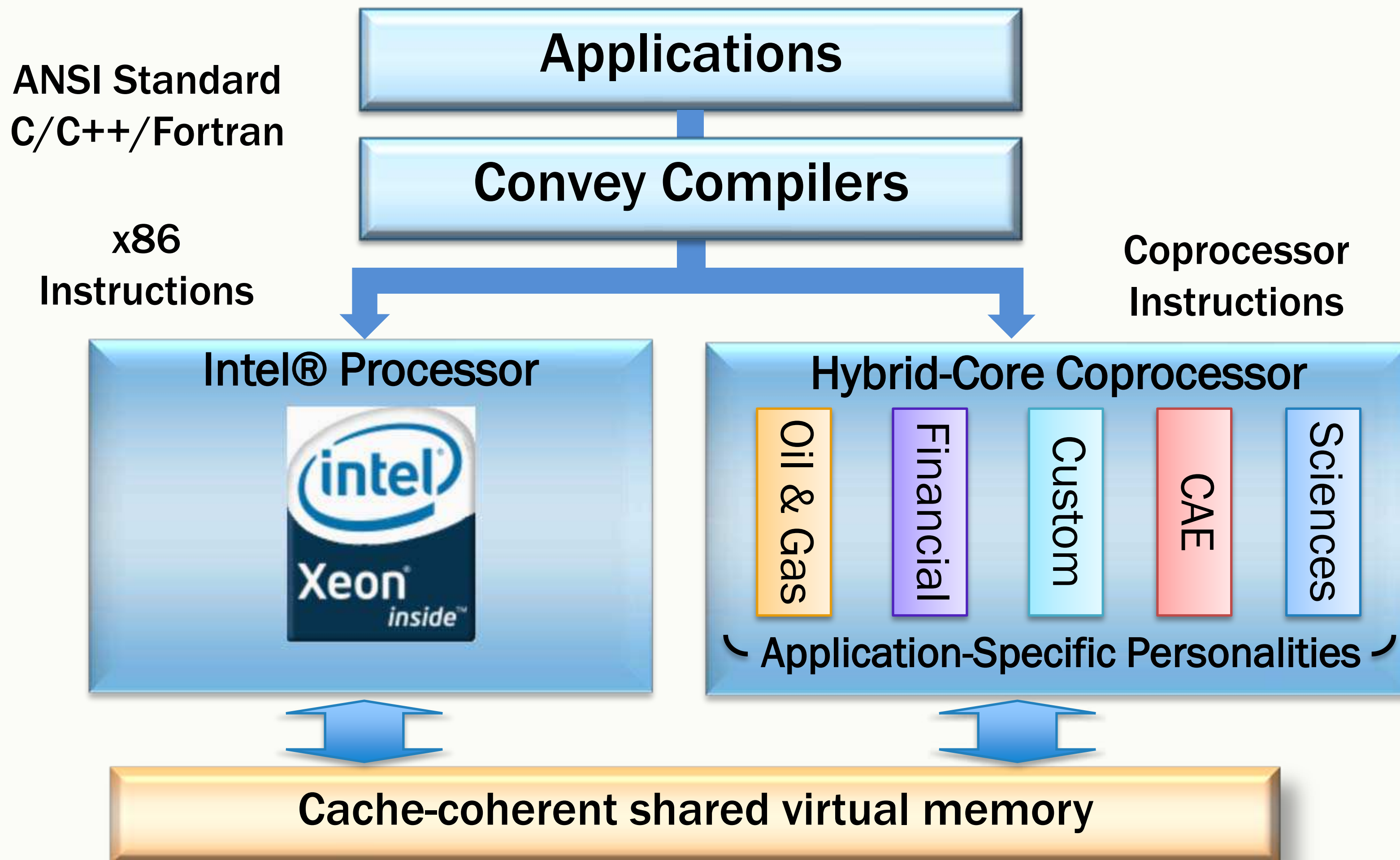
Presentation Outline

- Overview of HC-1 Computer
- Instruction Set Innovations
- Application Examples

CONVEY computer™

What is a Hybrid-Core Computer ?

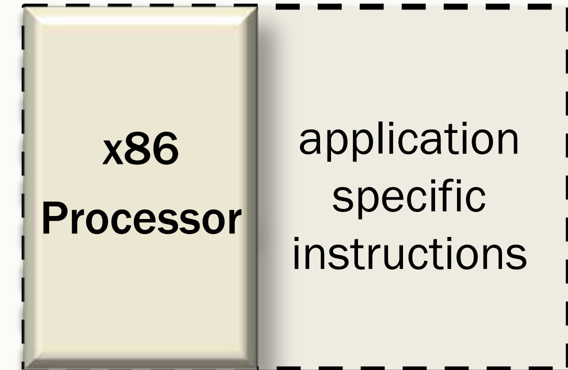
A hybrid-core computer improves application performance by combining an x86 processor with hardware that implements application-specific instructions.



CONVEY computer™

What Is a Personality?

- **A personality is a reloadable set of instructions that augment the x86 instruction set**
  - Applicable to a class of applications or specific to a particular code

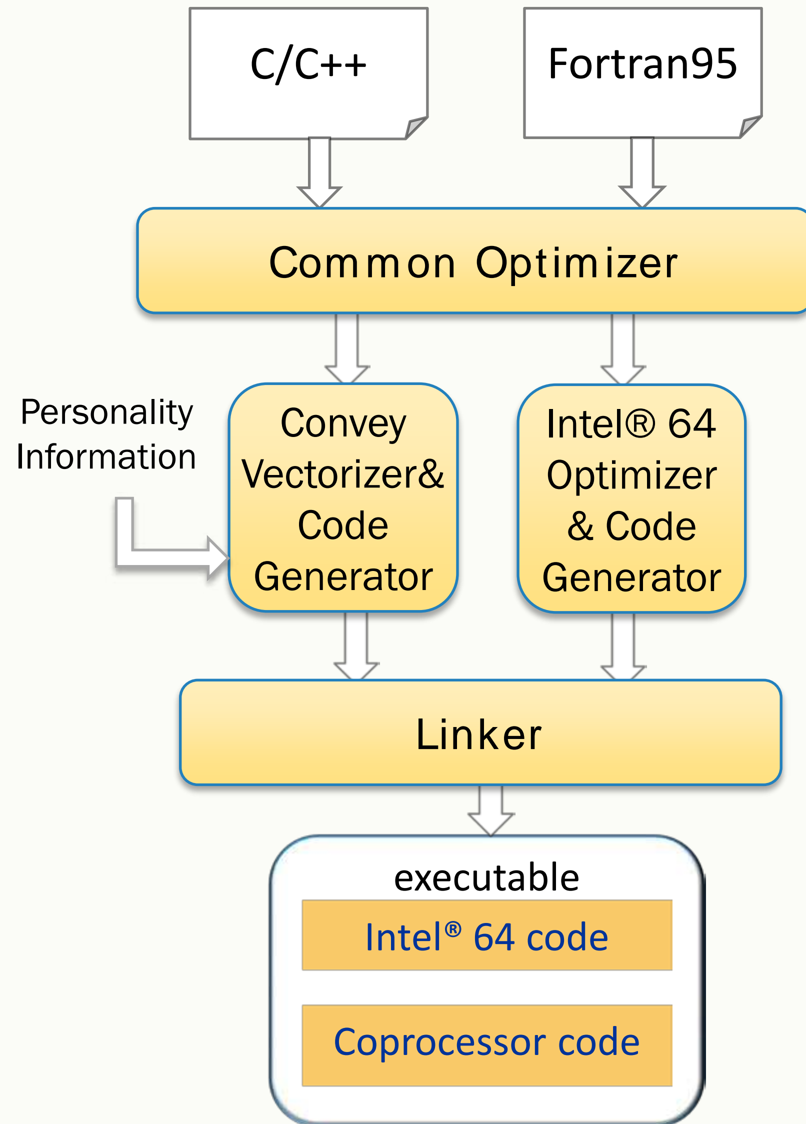


- **Each personality is a set of files that includes:**
  - The bits loaded into the Coprocessor
  - Information used by the Convey compiler
    - List of available instructions
    - Performance characteristics for compiler optimization
  - Information used by the GNU Debugger (GDB)
    - Instruction disassembly information
    - Information allowing machine state formatting and modification

CONVEY computer™

How do you Program using Personalities?

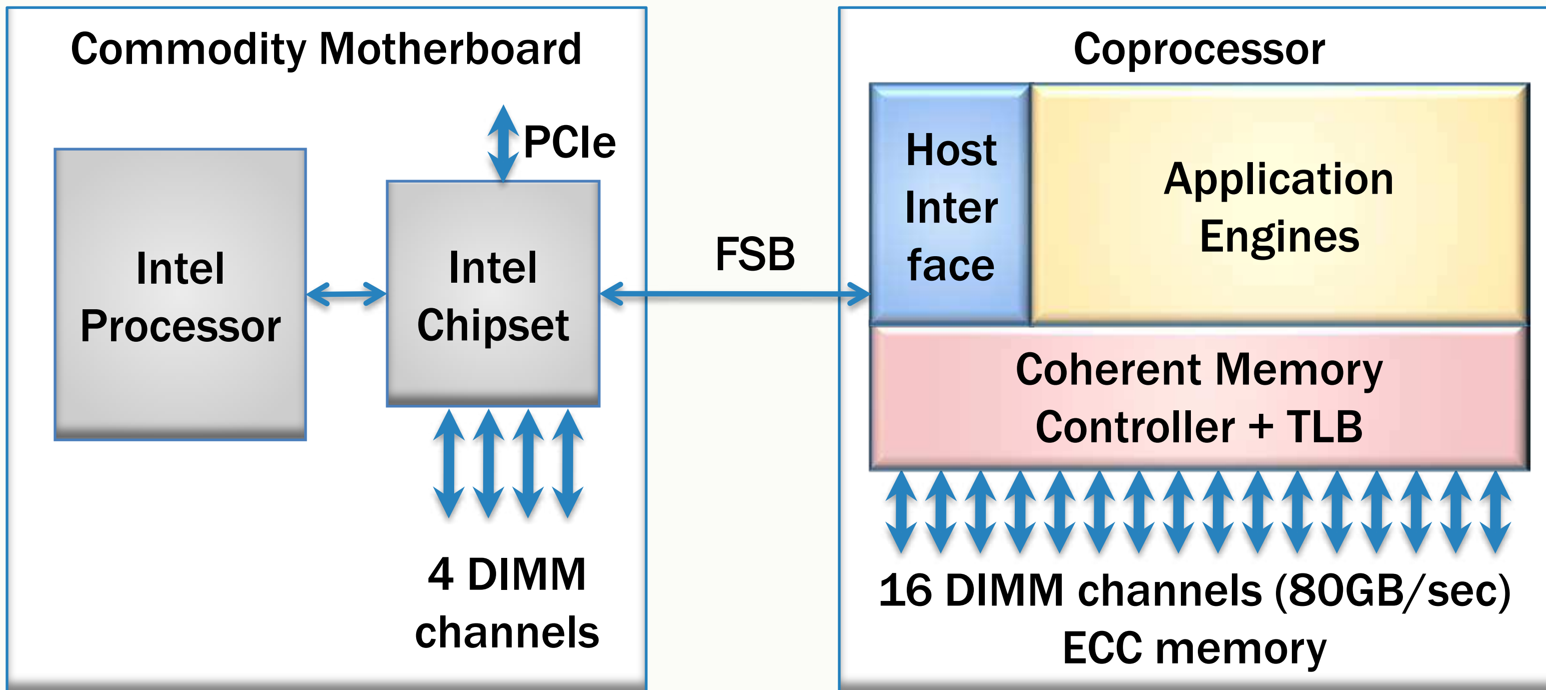
- Program in ANSI standard C/C++ or Fortran
- Unified compiler generates x86 and coprocessor instructions
- Compiler generates x86 code for full application and coprocessor code where data parallelism exists
- Run time checks determine if sufficient parallelism exists to use coprocessor
- Executable can run on x86 nodes or on Convey Hybrid-Core nodes



CONVEY computer™

Convey System Architecture

Intel x86
Linux ecosystem



Shared physical and virtual memory provides coherent programming model

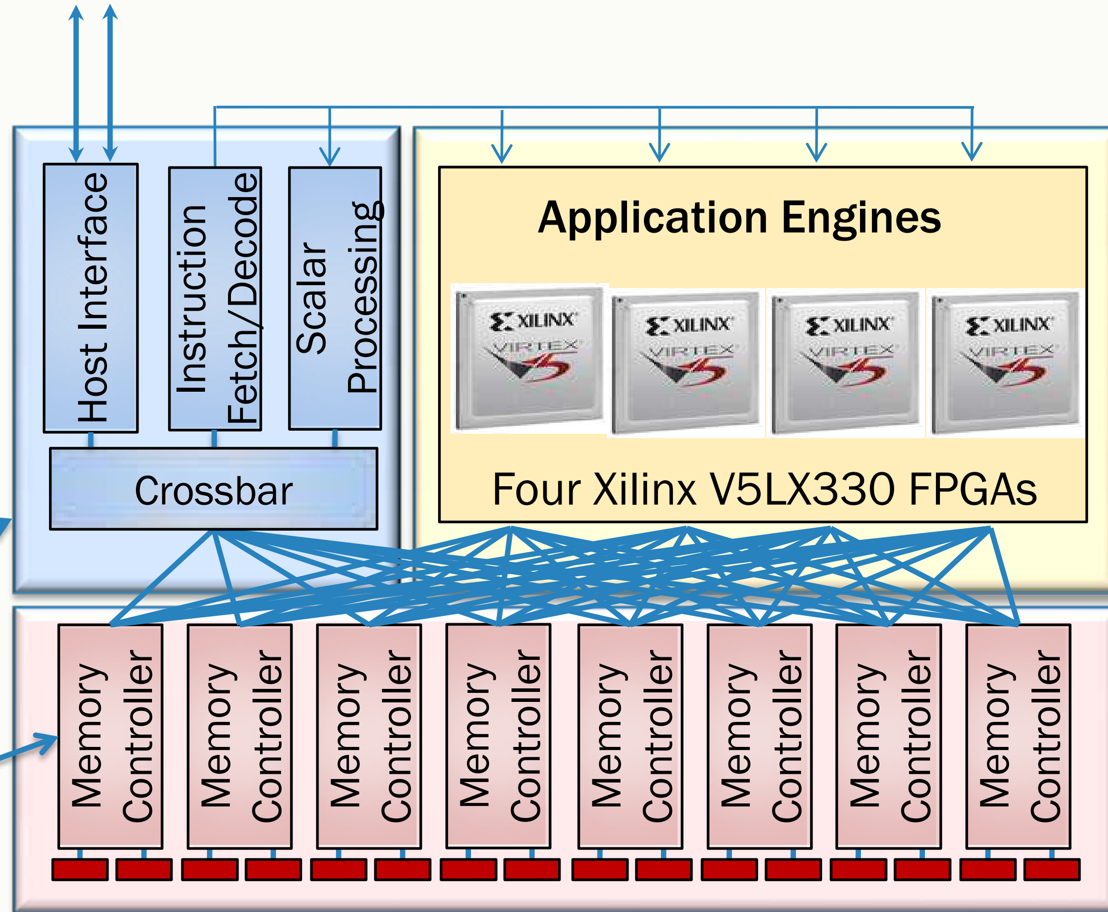
CONVEY computer™

Inside the Coprocessor

Host interface and memory controllers implemented by coprocessor infrastructure

Implemented with Xilinx V5LX110 FPGAs

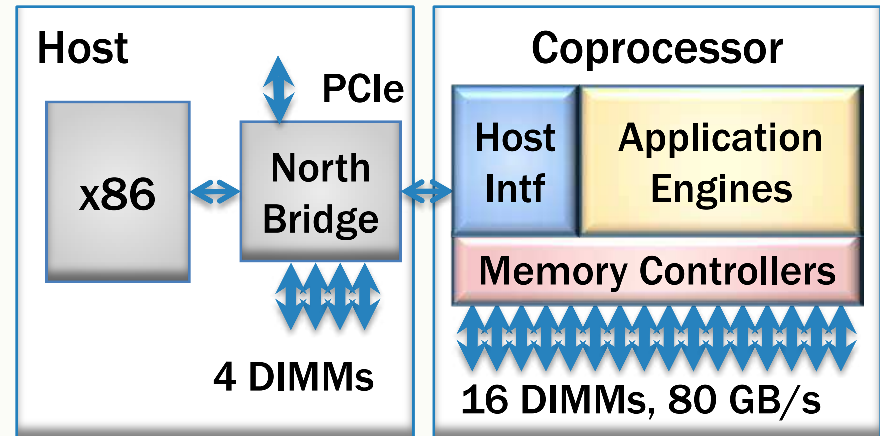
Implemented with Xilinx V5LX155 FPGAs



16 DDR2 memory channels
Standard or Scatter-Gather DIMMs
80GB/sec throughput

CONVEY computer™

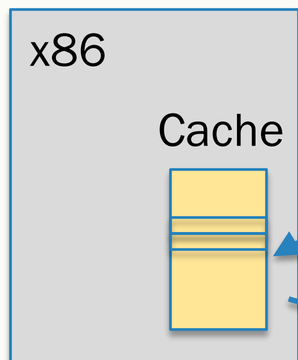
High Performance Memory Subsystem



- Mixed access mode
  - cache line or quad word
- x86 cache coherency support
  - without impacting coprocessor bandwidth
- Prime number interleave (31)
  - full memory bandwidth at all strides (except a multiple of prime number)
- Convey designed Scatter / Gather DIMMs or Standard DIMMs
  - Scatter / Gather DIMMs provide 8-byte accesses at full bandwidth
- Large virtual page support (4MB)
  - entire virtual address of process is resident within the coprocessor TLBs
- Page Coloring within OS
  - interleave pattern to span entire virtual address space of application

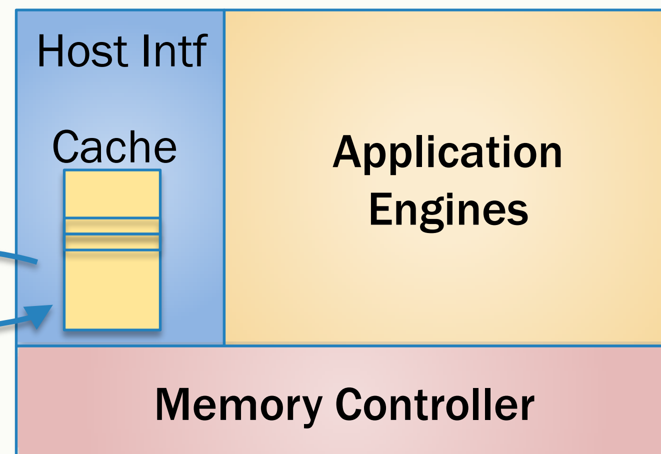
CONVEY computer™

How is the Coprocessor Started?



Start Sequence:

1. x86 writes "Start Message" to a predefined 64-byte memory line. Ownership for line is moved from coprocessor cache to x86 cache.



2. x86 writes start bit in a second memory line.

3. Coprocessor detects "Start Bit" memory line has been invalidated from its cache and immediately retrieves the "Start Message" and "Start Bit" memory lines.

4. Coprocessor checks that the "Start Bit" is set and starts coprocessor.

Start Message

Personality
Start Addr.
Param 1
Param 2
Param 3
Param 4
Param 5
Param 6

Mechanism works with:
- Host thread time sharing
- Application paging
- Front Side Bus or QPI

Application can start coprocessor synchronously, or asynchronously

CONVEY computer™

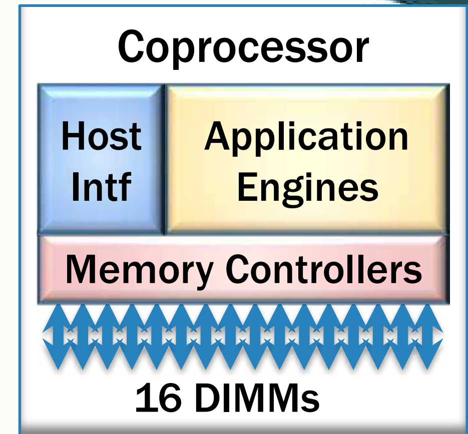
A view from the Inside

- Top half of platform is the Coprocessor
- Bottom half is the Intel Motherboard

2U Platform

CONVEY computer™

Architecture for a Dynamically Loadable Instruction Set



- Three classes of instructions are defined with separate machine state for each

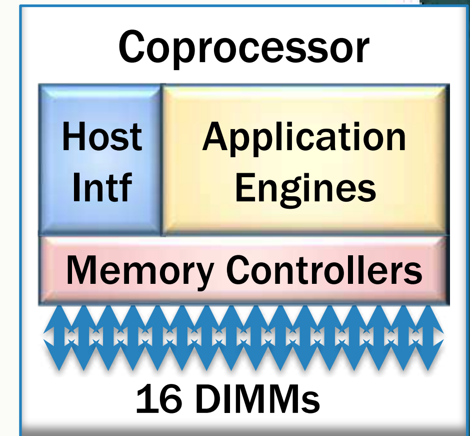
Instruction Class / Examples	Machine State	Pre-defined?	Location Executed
Address CALL Routine	A Registers	Yes	Scalar Processor
Scalar S5 ← 2.0 * S6	S Registers	Yes	Scalar Processor
Application Engine (SIMD) V2 ← S6 + V4	Defined by Personality	No	Application Engines

- Address and Scalar instructions are available to all personalities

Note: The Scalar Processor is implemented within the Host Interface

CONVEY computer™

How does the Scalar Processor Interact with AE Instructions?



Problem: The Scalar Processor functionality is fixed (I.e. does not depend on the personality loaded on the AEs). How does the Scalar Processor know how to interact with user defined AE instructions?

Solution: The Application Engine instruction encoding space is partitioned into regions. Each region has predefined interactions with the Scalar Processor machine state (A and/or S Registers).

Example Application Engine Instructions	Instruction Operation	A/S Register Interaction
V4 ← S2 + V3	Add a scalar (S2) to each element of a vector (V3) and write the result to a vector (V4).	Send S2 to AE
A5 ← VL	Move vector length (VL) to an address register (A5).	Receive VL from AE

- **A personality must select encodings for AE instructions based on the interactions between A/S registers and the AE instructions.**

CONVEY computer™

A Personality's Machine State can be Saved/Restored to Memory

- **Each personality defines its own machine state**
  - Machine state is simply the instruction visible register state.
  - The coprocessor's machine state is only saved to memory at instruction boundaries (I.e. once all executing instructions have completed).

- **A common mechanism is used to save / restore machine state for all personalities**
  - All personality machine state is mapped to AEG registers (Application Engine Generic registers).
  - A few pre-defined AE instructions allow access to AEG registers for the purpose of machine state save and restore to/from memory.
  - The following predefined AE instructions allow user defined machine state to be saved/restored:

```
MOV     AEGcnt, At      ; Obtain the number of defined AEG registers
MOV     Sa,Ab,AEG       ; Restore an AEG[Ab] register from an S register
MOV     AEG,Ab,St       ; Save an AEG[Ab] register to an S register
```

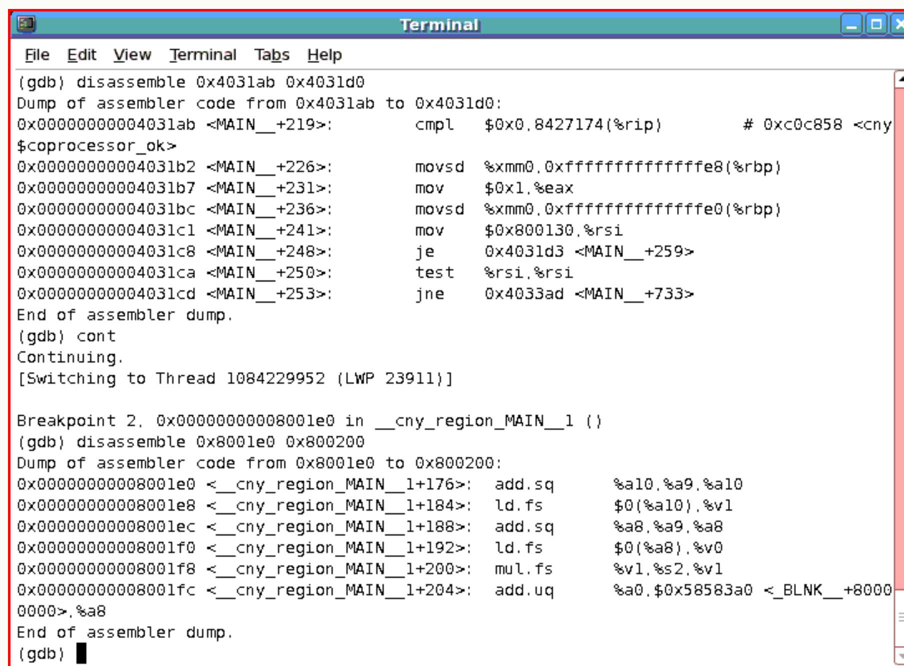
  - The operating system has one machine state save / restore routine used by all personalities.

CONVEY computer™

Machine State in Memory enables Application Debugging

- Coprocessor machine state can be accessed (read and written) by the GNU Debugger (GDB).

- GDB can perform single step, run and stop at breakpoint instruction.



CONVEY computer™

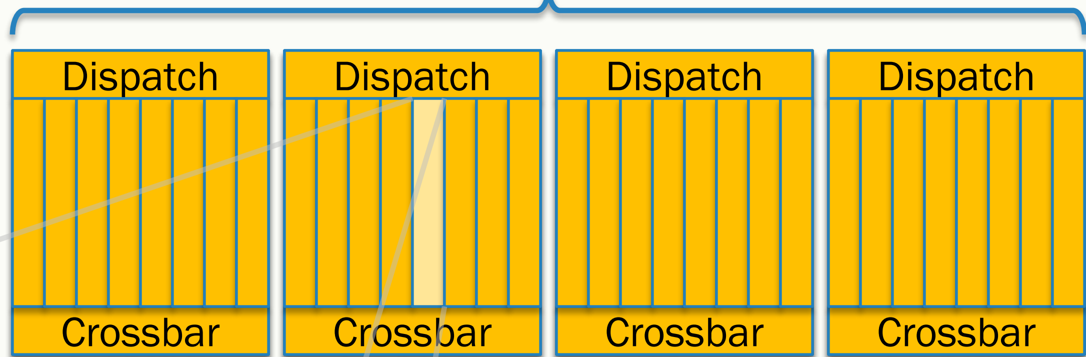
Single Precision Vector Personality

Load-store vector architecture with modern latency-hiding features

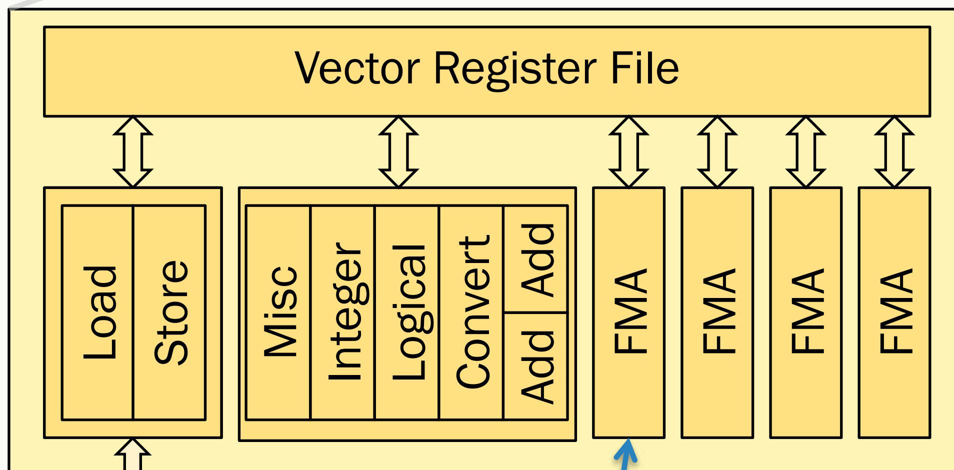
Optimized for Signal Processing applications

4 Application Engine FPGAs / 32 Function Pipes

vector elements distributed across function pipes



1-of-32 Function Pipes



To Crossbar

FMA - Fused Multiply Add

A function pipe is the unit of data path replication

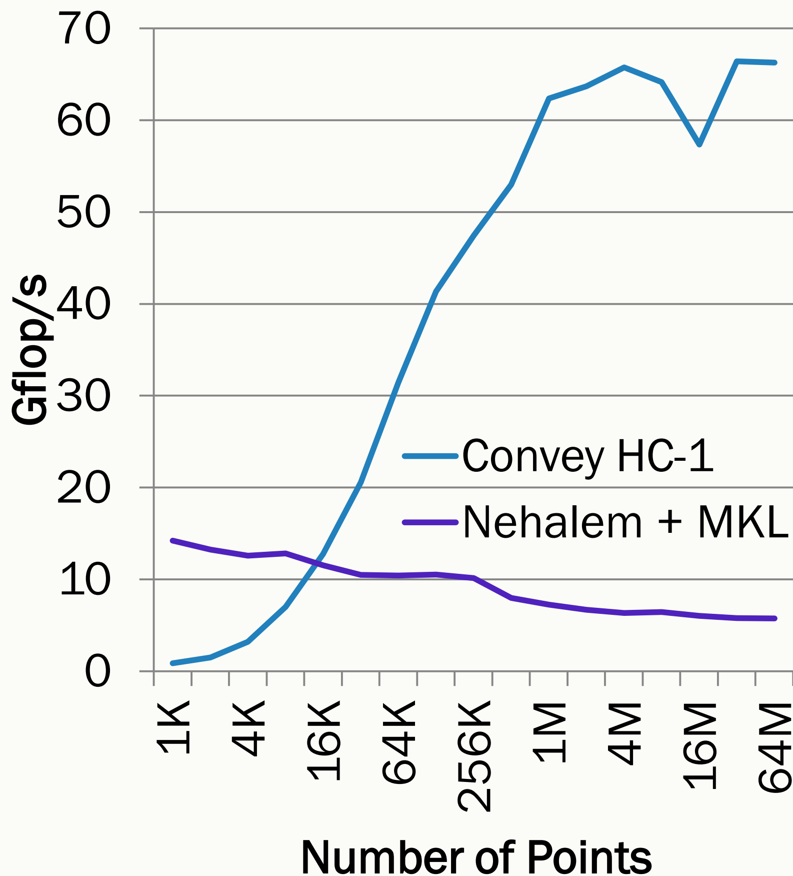
Same instructions sent to all function pipes (SIMD)

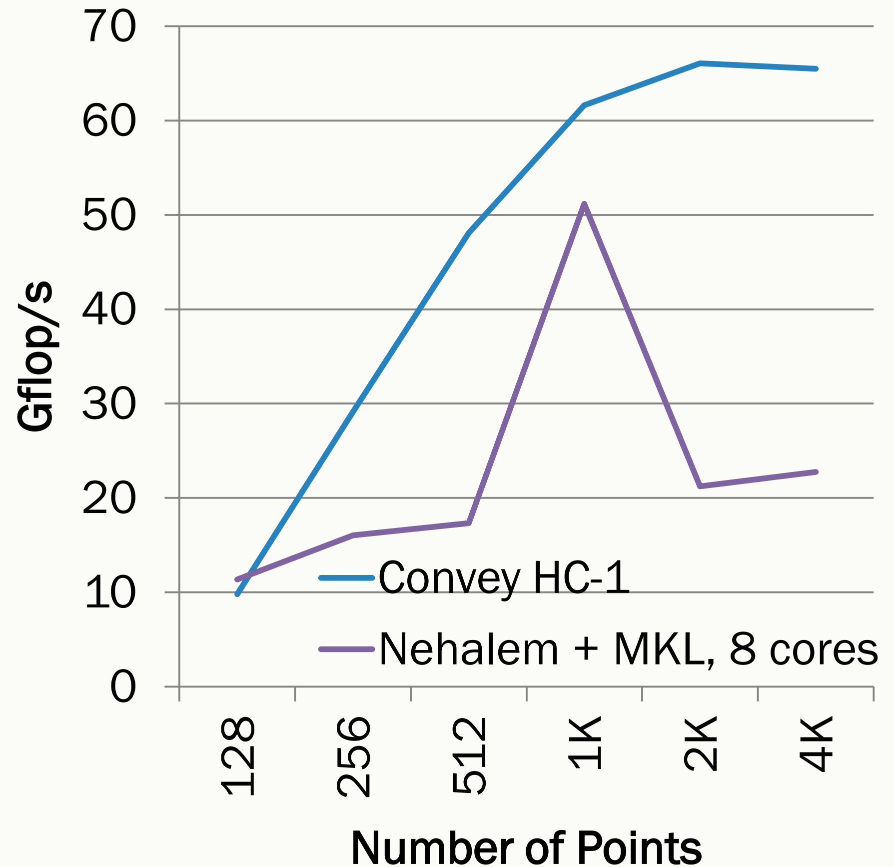
Each function pipe supports:

- Multiple functional units
- Out-of-order execution
- Register renaming

CONVEY computer™

Single Precision Vector Personality FFT Performance

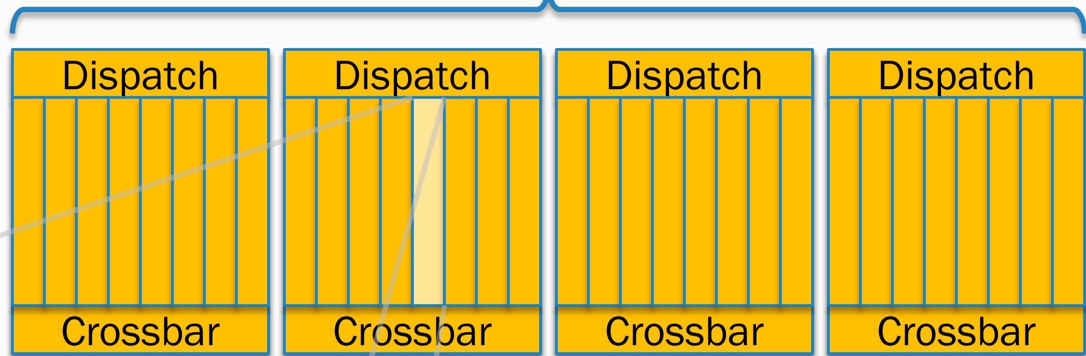


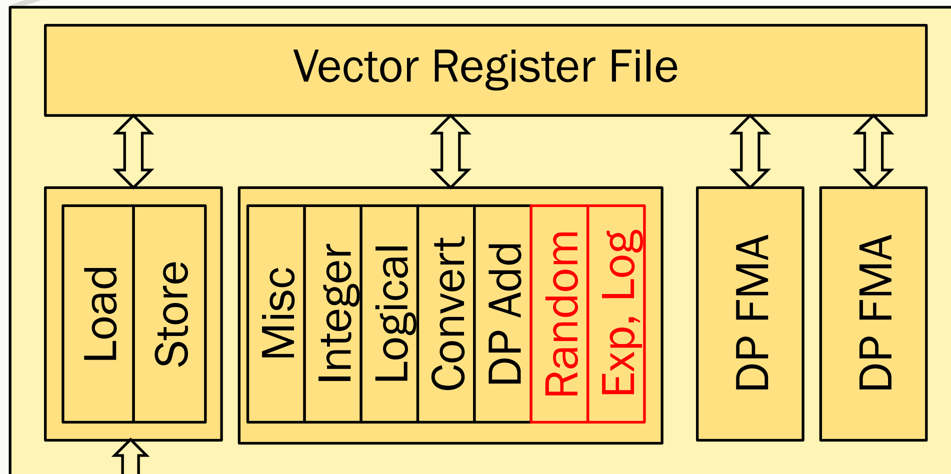


CONVEY computer™

Financial Vector Personality

Pairs of single precision functional units replaced by double precision units





Functional units for *log, exp, random number generation*

Supported by the compiler as vector intrinsics

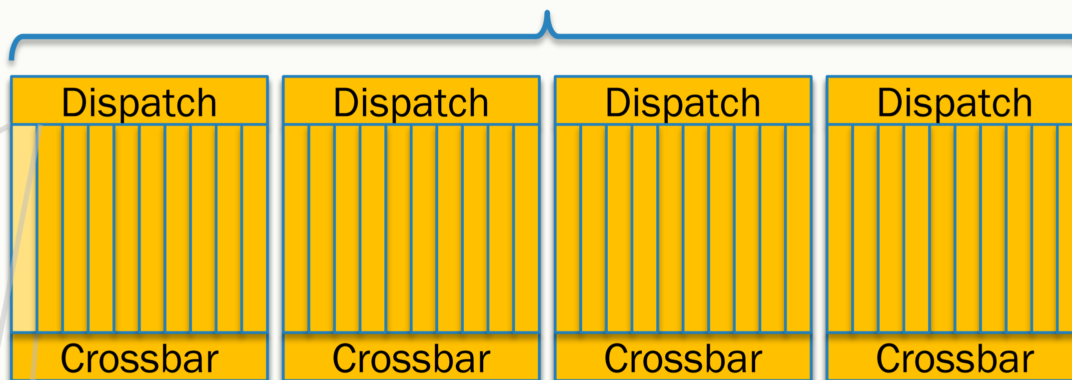
Allows compile and run model for financial analytic applications

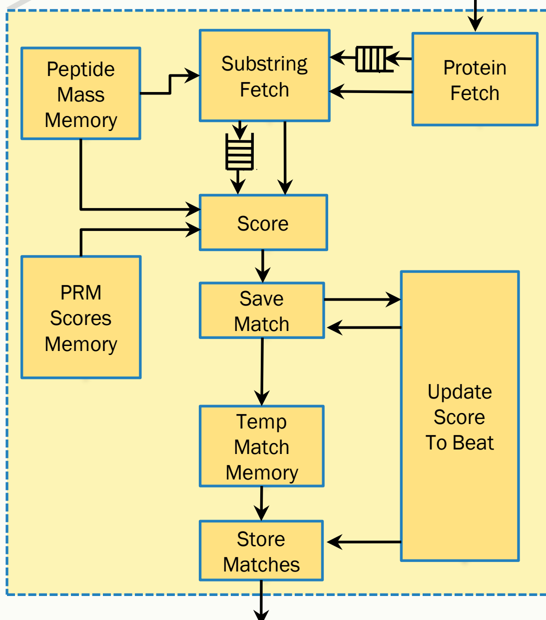
CONVEY computer™

Protein Sequencing Personality

Inspect Application ported to Convey HC-1 as joint project with UCSD

Application performs protein sequencing





Entire "Best Match" routine implemented as state machine

Multiple state machines for data parallelism (MIMD)

Operates on main memory using virtual addresses

Executes greater than 100 times faster than a single Nehalem core

CONVEY computer™

In Summary

- **The Convey Approach Provides:**
  - Virtual, cache coherent, global address space
  - Instruction based FPGA compute model
  - ANSI Standard C/C++/ Fortran high level language support
  - Integrated debugging environment using GDB (text or GUI based)

- **The Convey system allows users to develop applications as one would for standard CPU based systems.**

CONVEY computer™