Intel® Processor Architecture

January 2013
Agenda

• Overview Intel® processor architecture
• Intel x86 ISA (instruction set architecture)
• Micro-architecture of processor core
• Uncore structure
• Additional processor features
  – Hyper-threading
  – Turbo mode
• Summary
## Intel® Processor Segments Today

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Target Platforms</th>
<th>ISA</th>
<th>Specific Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel® ATOM™ Architecture</td>
<td>phone, tablet, netbook, low-power server</td>
<td>x86 up to SSSE-3, 32 and 64 bit</td>
<td>optimized for low-power, in-order</td>
</tr>
<tr>
<td>Intel® Core™ Architecture</td>
<td>mainstream notebook, desktop, server</td>
<td>x86 up to Intel® AVX, 32 and 64bit</td>
<td>flexible feature set covering all needs</td>
</tr>
<tr>
<td>Intel® Itanium® Architecture</td>
<td>high end server</td>
<td>IA64, x86 by emulation</td>
<td>RAS, large address space</td>
</tr>
<tr>
<td>Intel® MIC Architecture</td>
<td>accelerator for HPC</td>
<td>x86 and Intel® MIC Instruction Set</td>
<td>+60 cores, optimized for Floating-Point performance</td>
</tr>
</tbody>
</table>
Itanium® 9500 (Poulson)
New Itanium Processor

• Compatible with Itanium® 9300 processors (Tukwila)
• New micro-architecture with 8 Cores
• 54 MB on-die cache
• Improved RAS and power management capabilities
• Doubles execution width from 6 to 12 instructions/cycle
• 32nm process technology
• Launched in November 2012

Compatibility provides protection for today’s Itanium® investment
Intel® XEON™ Phi - The first product implementation of the Intel® Many Integrated Core Architecture (Intel® MIC)
Agenda

• Overview Intel® processor architecture
• Intel x86 ISA (instruction set architecture)
• Micro-architecture of processor core
• Uncore structure
• Additional processor features
  – Hyper-threading
  – Turbo mode
• Summary
X86: From Smartphones to ...
Motorola RAZR* i

• Launched September 2012
• RAZR i is the first smartphone that can achieve speeds of 2.0 GHz

Processor:

Intel® ATOM™ Z2460
X86: ... to Supercomputers
LRZ SuperMUC System

• Installed summer 2012
  – Most powerful x86-architecture based computer
  – #6 on Top500 list
  – More than 150000 cores

• Processor:
  - Intel® Xeon® E5-2680 („Sandy Bridge‟)
  - Intel® Xeon® E7-4870 („Westmere‟)
Intel Tick-Tock Roadmap for Mainstream x86 Architecture since 2006

**Intel® Core™ MicroArchitecture**
- **Merom**: NEW Micro architecture, 65nm, TOCK
- **Penryn**: NEW Process Technology, 45nm, TICK
- **Nehalem**: NEW Micro architecture, 45nm, TOCK
- **Westmere**: NEW Process Technology, 32nm, TICK
- **Sandy Bridge**: NEW Micro architecture, 32nm, TOCK
- **Ivy Bridge**: NEW Process Technology, 22nm, TICK

**Micro Architecture Codename “Nehalem”**
- **2006**: SSSE-3
- **2007**: SSE4.1
- **2008**: SSE4.2
- **2009**: AES
- **2011**: AVX
- **2012**: 7 new instructions

**2nd Generation Intel® Core™ Micro Architecture**
- **2008**: AES
- **2009**: AVX

**3rd Generation Intel® Core™ Micro Architecture**
- **2011**: AVX

**TICK + TOCK = SHRINK + INNOVATE**
To be continued ...

<table>
<thead>
<tr>
<th>4th Generation</th>
<th>22nm</th>
<th>14nm</th>
<th>14nm</th>
<th>10nm</th>
<th>10nm</th>
<th>7nm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel® Core™ Micro Architecture</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
</tr>
<tr>
<td>2013 AVX-2</td>
<td>&gt;= 2014</td>
<td>???</td>
<td>???</td>
<td>???</td>
<td>???</td>
<td></td>
</tr>
</tbody>
</table>

TICK + TOCK = SHRINK + INNOVATE
Registers State for Intel® Pentium® 3 Processor (1998)

IA32-INT Registers
- Fourteen 32-bit registers
- Scalar data & addresses
- Direct access to registers

MMX Technology / IA-FP Registers
- Eight 80/64-bit registers
- Hold data only
- Direct access to MM0..MM7
- No MMX™ Technology / FP interoperability

SSE Registers
- Eight 128-bit registers
- Hold data only:
  - 4 x single FP numbers
  - 2 x double FP numbers
  - 128-bit packed integers
SSE Vector Types

Intel® SSE

4x single precision FP

Intel® SSE2

2x double precision FP

16x 8 bit integer

8x 16 bit integer

4x 32 bit integer

2x 64 bit integer

plain 128 bit
AVX Vector Types

**Intel® AVX**
- 8x single precision FP
- 4x double precision FP

**Intel® AVX2 (Future)**
- 32x 8 bit integer
- 16x 16 bit integer
- 8x 32 bit integer
- 4x 64 bit integer
- plain 256 bit
X86 ISA: The Instruction Set

• The instruction set for the x86 architecture has been extended numerous times since the set supported by the 8086 processor

• Today, the “base” instructions set (“IA32 ISA”) is the one supported by the first 32bit processor - 80386

• Multiple, “smaller” extensions added then before SSE (1998 / Intel® Pentium® 3) like
  - MMX( 64 bit SIMD using the x87 FP registers)
  - Conditional move
  - Atomic exchange
# New Instructions in Haswell (2013)

<table>
<thead>
<tr>
<th>Group</th>
<th>Description</th>
<th>Count *</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVX2</td>
<td><strong>SIMD Integer Instructions promoted to 256 bits</strong>&lt;br&gt;Adding vector integer operations to 256-bit</td>
<td>170 / 124</td>
</tr>
<tr>
<td></td>
<td><strong>Gather</strong>&lt;br&gt;Load elements from vector of indices&lt;br&gt;vectorization enabler</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Shuffling / Data Rearrangement</strong>&lt;br&gt;Blend, element shift and permute instructions</td>
<td></td>
</tr>
<tr>
<td>FMA</td>
<td><strong>Fused Multiply-Add operation forms (FMA-3)</strong></td>
<td>96 / 60</td>
</tr>
<tr>
<td></td>
<td><strong>Bit Manipulation and Cryptography</strong>&lt;br&gt;Improving performance of bit stream manipulation and decode, large integer arithmetic and hashes</td>
<td>15 / 15</td>
</tr>
<tr>
<td>TSX=RTM+HLE</td>
<td><strong>Transactional Memory</strong></td>
<td>4 / 4</td>
</tr>
<tr>
<td>Others</td>
<td><strong>MOVBE: Load and Store of Big Endian forms</strong>&lt;br&gt;<strong>INVPCID: Invalidate processor context ID</strong></td>
<td>2 / 2</td>
</tr>
</tbody>
</table>

* Total instructions / different mnemonics
HSW Improvements for Threading
Sample Code Computing PI by Windows Threads

```c
#include <windows.h>
define NUM_THREADS 2
HANDLE thread_handles[NUM_THREADS];
CRITICAL_SECTION hUpdateMutex;
static long num_steps = 100000;
double step;
double global_sum = 0.0;

void Pi (void *arg)
{
    int i, start;
double x, sum = 0.0;

    start = *((int *) arg);
    step = 1.0/(double) num_steps;

    for (i=start;i<= num_steps; i=i+NUM_THREADS){
        x = (i-0.5)*step;
        sum = sum + 4.0/(1.0+x*x);
    }
    EnterCriticalSection(&hUpdateMutex);
    global_sum += sum;
    LeaveCriticalSection(&hUpdateMutex);
}

void main ()
{
    double pi; int i;
    DWORD threadID;
    int threadArg[NUM_THREADS];

    for(i=0; i<NUM_THREADS; i++)
        threadArg[i] = i+1;

    InitializeCriticalSection(&hUpdateMutex);
    for (i=0; i<NUM_THREADS; i++){
        thread_handles[i] = CreateThread(0, 0,
                    (LPTHREAD_START_ROUTINE) Pi,
                    &threadArg[i], 0, &threadID);
    }
    WaitForMultipleObjects(NUM_THREADS, thread_handles, TRUE,INFINITE);

    pi = global_sum * step;
    printf(" pi is %f \n",pi);
}
```

Locks can be key bottleneck – even in case there is no conflict
Intel® Transactional Synchronization Extensions (Intel® TSX)

Intel® TSX = HLE + RTM

HLE (Hardware Lock Elision) is a hint inserted in front of a LOCK operation to indicate a region is a candidate for lock elision
- XACQUIRE (0xF2) and XRELEASE (0xF3) prefixes
- Don’t actually acquire lock, but execute region speculatively
- Hardware buffers loads and stores, checkpoints registers
- Hardware attempts to commit atomically without locks
- If cannot do without locks, restart, execute non-speculatively

RTM (Restricted Transactional Memory) is three new instructions (XBEGIN, XEND, XABORT)
- Similar operation as HLE (except no locks, new ISA)
- If cannot commit atomically, go to handler indicated by XBEGIN
- Provides software additional capabilities over HLE
Agenda

• Overview Intel® processor architecture
• Intel x86 ISA (instruction set architecture)
• Micro-architecture of processor core
• Uncore structure
• Additional processor features
  – Hyper-threading
  – Turbo mode
• Summary
Core™ 2 Architecture (Merom)

Front End

- Instruction Fetch and Pre Decode
- Instruction Queue
- Decode
- Rename/Allocate
- Reservation Station
- Execution Units
- Retirement Unit (ReOrder Buffer)

Memory

- ITLB
- 32kB Instruction Cache
- 2/4/6 MB 2\textsuperscript{nd} Level Cache
- Front-Side Bus

Out-Of-Order Execution Engine

- DTLB
- 32kB Data Cache
## Peak FP Performance per Core & Cycle

<table>
<thead>
<tr>
<th></th>
<th>Single Precision</th>
<th>Double Precision</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nehalem</td>
<td>8</td>
<td>4</td>
<td>By SSE; MULT and ADD can start each cycle:</td>
</tr>
<tr>
<td>Sandy Bridge</td>
<td>16</td>
<td>8</td>
<td>AVX doubles all due to twice the vector length</td>
</tr>
<tr>
<td>Haswell</td>
<td>32</td>
<td>16</td>
<td>2 FMA instructions can start each cycle – doubling performance compared to SNB</td>
</tr>
</tbody>
</table>

For a 2-socket, 16-core Haswell server system running at 3 GHz, this will sum up to 1.5 terra flops SP FP peak performance (0.77 for DP)
Agenda

• Overview Intel® processor architecture
• Intel x86 ISA (instruction set architecture)
• Micro-architecture of processor core
• Uncore structure
• Additional processor features
  – Hyper-threading
  – Turbo mode
• Summary
Common Core, Modular Uncore

- **Common “core”**
  - Same core for server, desktop, mobile
  - Incremental improvements to u-arch of current Core architecture
  - Common target for SW optimization
  - Common feature set

- **Segment differentiation in the “Uncore”**
  - # of cores
  - # of QPI links
  - Size of L3 cache
  - # IMC channels
  - Frequency DDR3
  - Integrated graphics (GT)
  - ...

<table>
<thead>
<tr>
<th></th>
<th># of Cores</th>
<th>L3$ Size</th>
<th>Memory Controller</th>
<th>QPI Links</th>
<th>Graphic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop i5</td>
<td>2</td>
<td>4MB</td>
<td>2xDDR3</td>
<td>N/A</td>
<td>Yes</td>
</tr>
<tr>
<td>Desktop i3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Desktop i7 NHM</td>
<td>4</td>
<td>8MB</td>
<td>3xDDR3</td>
<td>1 x 4.8</td>
<td>Yes</td>
</tr>
<tr>
<td>Desktop i7 SNB</td>
<td>6</td>
<td>8MB</td>
<td>3xDDR3</td>
<td>1 x 6.4</td>
<td>Yes</td>
</tr>
<tr>
<td>XEON E5-2600</td>
<td>2x8</td>
<td>20MB</td>
<td>4xDDR3</td>
<td>2 x 8.0</td>
<td>No</td>
</tr>
<tr>
<td>XEON E7-8870</td>
<td>4x10</td>
<td>30MB</td>
<td>3xDDR3</td>
<td>4 x 6.4</td>
<td>No</td>
</tr>
</tbody>
</table>

*Other brands and names are the property of their respective owners.*
Level 3 Cache

- New 3\textsuperscript{rd} level cache
  - Also called LLC – Last Level Cache
- Shared across all cores of processor (socket)
- Size
  - NHM: 2MB/core (EX up to 3.0)
  - SNB: 2.5 MB/core (today)
- Latency:
  - NHM: \geq 35
  - SNB: 25-31
- Inclusive property
  - Cache line residing in L1/L2 must be present too in 3\textsuperscript{rd} level cache
QuickPath Interconnect

• Nehalem introduces new QuickPath Interconnect (QPI)

• **High bandwidth, low latency** point to point interconnect

• 4.8/6.4/8.0 GT/sec
  - E.g. 6.4 GT/sec -> 12.8 GB/sec each direction

• Highly **scalable** for systems with varying # of sockets
Remote Memory Access

- CPU0 requests cache line X, not present in any CPU0 cache
  - CPU0 requests data from CPU1; request sent over QPI to CPU1
  - CPU1’s IMC makes request to its DRAM
  - CPU1 snoops internal caches
  - Data returned to CPU0 over QPI
- Remote memory latency a function of having a low latency interconnect
  - Typical numbers: Local access 60ns, remote access 90ns
Non-NUMA (UMA) Mode

- Addresses interleaved across memory nodes by cache line
  - Some systems too support page size granularity
- Accesses may or may not have to cross QPI link

**UMA lacks tuning for peak performance but in general delivers good performance without any additional tuning effort**
NUMA Mode

- Non-Uniform Memory Access (NUMA)
- Addresses not interleaved across memory nodes by cache line.
- Each CPU has direct access to contiguous block of memory.

Combined with thread affinity (“pinning”) enables potential for peak performance but can degrade performance in case not taken care of.
Uncore Architecture: Sandy Bridge
Significant Bandwidth Increases over Prior Generation

- PCIe BW: \(~300\%\)
- Cache BW: \(~800\%\)
- On-Die Interconnect BW: \(~900\%\)
- Socket to Socket BW: \(~250\%\)
- DDR3 BW: \(~200\%\)
**SNB: Scalable Ring On-die Interconnect**

- **Ring-based** interconnect between Cores, Graphics, Last Level Cache (LLC) and System Agent domain

- Composed of **4 rings**
  - 32 Byte *Data* ring, *Request* ring, *Acknowledge* ring and *Snoop* ring
  - Fully pipelined at **core frequency** bandwidth, latency scale with cores

- Access on ring always picks the **shortest path** – minimize latency

- **Distributed arbitration**, sophisticated ring protocol to handle coherency, ordering, and core interface

- **Scalable to servers** with large number of processors
Agenda

• Overview Intel® processor architecture
• Intel x86 ISA (instruction set architecture)
• Micro-architecture of processor core
• Uncore structure
• Additional processor features
  – Hyper-threading
  – Turbo mode
• Summary
**Intel® Turbo Boost Improvements**

<table>
<thead>
<tr>
<th>Client</th>
<th>Nehalem/Westmere</th>
<th>Sandy Bridge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merom/Penryn (Mobile only)</td>
<td>Clarksfield Lynnfield/Clarkdale</td>
<td>Arrandale</td>
</tr>
</tbody>
</table>
| **Key New Capabilities** | • Turbo controlled within power limit  
• Multi-core turbo  
• More turbo if cores are asleep | • Graphics Dynamic Frequency  
• Driver controlled power sharing between IA and Graphics (Mobile) | • HW controlled power sharing between IA cores and Graphics  
• Dynamic Turbo provides high responsiveness  
• More Turbo headroom from Improved power monitoring and control |

**Turbo Behavior**

Illustrative only. Does not represent actual number of turbo bins.

![Turbo Behavior Diagram](image-url)
Dynamic Adaptation in Sandy Bridge

Buildup thermal budget during idle periods

"Next Gen Turbo Boost"

"TDP"

Sleep or Low power

C0 (Turbo)

Time

Power

After idle periods, the system accumulates "energy budget" and can accommodate high power/performance for a few seconds

In Steady State conditions the power stabilizes on TDP

Use accumulated energy budget to enhance user experience

P > TDP: Responsiveness

"Next Gen Turbo Boost"

Buildup thermal budget during idle periods

"TDP"

Sleep or Low power

C0 (Turbo)
Simultaneous Multi-Threading (SMT) “Intel Hyper-Threading – HT”

- Run 2 threads at the very same time per core
- Available on Nehalem (and successors) as well as Intel® ATOM Architecture
- Take advantage of 4-wide execution engine
  - Keep it fed with multiple threads
  - Hide latency of a single thread
- Most **power efficient** performance feature
  - Very low die area cost
  - Can provide significant performance benefit depending on application
  - Much more efficient than adding an entire core
- Nehalem advantages
  - Larger caches
  - Massive memory BW

Note: Each box represents a processor execution unit.
Performance Gain SMT enabled vs disabled

Floating Point 3dsMax* Integer Cinebench* 10POV-Ray* 3.7 3DMark* Vantage* CPU

Intel® Core™ i7

7% 10% 13% 16% 29% 34%

Floating Point is based on SPECfp_rate_base2006* estimate
Integer is based on SPECint_rate_base2006* estimate

SPEC, SPECint, SPECfp, and SPECrate are trademarks of the Standard Performance Evaluation Corporation.
For more information on SPEC benchmarks, see: http://www.spec.org

Source: Intel. Configuration: pre-production Intel® Core™ i7 processor with 3 channel DDR3 memory. Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, visit http://www.intel.com/performance/

*Other brands and names are the property of their respective owners.
Agenda

• Overview Intel® processor architecture
• Intel x86 ISA (instruction set architecture)
• Micro-architecture of processor core
• Uncore structure
• Additional processor features
  – Hyper-threading
  – Turbo mode

• Summary
<table>
<thead>
<tr>
<th>Platform</th>
<th>Memory Bandwidth</th>
<th>GFLOPs (DP) per core</th>
<th>FLOPS per DP Data Move to get Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td>NHM: 32GB/Socket 4 cores</td>
<td>8.00GB/core (3ch x 1333 x 8bytes)/4</td>
<td>12 4 x 3GHz</td>
<td>12.0</td>
</tr>
<tr>
<td>WSM: 32GB/Socket 6 cores</td>
<td>5.33GB/core (3ch x 1333 x 8bytes)/6</td>
<td>9.6 4 x 2.4GHz</td>
<td>14.4</td>
</tr>
<tr>
<td>SNB: 51GB/Socket 8 cores, SSE</td>
<td>6.40GB/core (4ch x 1600 x 8bytes)/8</td>
<td>9.6 4 x 2.4GHz</td>
<td>12.0</td>
</tr>
<tr>
<td>SNB: 51GB/Socket 8 cores, AVX</td>
<td>6.40GB/core (4ch x 1600 x 8bytes)/8</td>
<td>19.2 8 x 2.4GHz</td>
<td>24.0</td>
</tr>
<tr>
<td>Itanium 2 “Montecito” Dual core</td>
<td>5.40 GB/core (0.677Ghz x 16bytes)/2</td>
<td>6.4 4 x 1.6 Ghz</td>
<td>9.5</td>
</tr>
</tbody>
</table>

Tuning for memory bandwidth remains key challenge!
References

- Intel **Software Development and Optimization manual**
- Session from Intel Developer Forum on processor architecture – www.intel.com/idf
- Michael E. Thomadakis, Texas University, “**The Architecture of the Nehalem Processor** ...”
- Agner, “**The microarchitecture of Intel, AMD and VIA CPUs** ...”
- Wikipedia
  - x86
  - x86 assembly language
Legal Disclaimer

INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance. Buyers should consult other sources of information to evaluate the performance of systems or components they are considering purchasing. For more information on performance tests and on the performance of Intel products, reference www.intel.com/software/products.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and other countries.  
*Other names and brands may be claimed as the property of others.  
Copyright © 2012.  Intel Corporation.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2®, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804
Intel® Xeon Phi™ Overview

Shared Memory Controllers

GDDR  GDDR  ...  GDDR

Memory

PCle x16 Interface

Coherent L2-Cache

Multi-Threaded Wide SIMD Core

I$  D$

Processor

Multi-Threaded Wide SIMD Core

I$  D$

Standard IA Shared Memory Programming

Future options subject to change without notice.

*Other brands and names are the property of their respective owners
Intel® Xeon Phi™ Microarchitecture Overview

TD: Tag Directory
L2: L2-Cache
MC: Memory Controller

For illustration only.
Interleaved Memory Access
Intel® Xeon Phi™ Core

Intel® Xeon Phi™ co-processor core:
• Scalar pipeline derived from the dual-issue Pentium processor
• Short execution pipeline
• Fully coherent cache structure
• Significant modern enhancements
  - such as multi-threading, 64-bit extensions, and sophisticated pre-fetching.
• 4 execution threads per core
• Separate register sets per thread
• 32KB instruction cache and 32KB data cache for each core.

Enhanced instructions set with:
• Over 100 new instructions
• Wide vector processing operations, incl. gather/scatter and masking
• Some specialized scalar instructions
• 3-operand, 16-wide vector processing unit (VPU)
• VPU executes integer, SP-float, and DP-float instructions
• Supports IEEE 754 2008 for floating point arithmetic

Interprocessor Network
1024 bits wide, bi-directional (512 bits in each direction)
Intel Xeon Phi Processor Core

L1 TLB and 32KB Code Cache

Decode

Pipe 0

Pipe 1

uCode

L2 TLB

TLB Miss Handler

HWP

L2 Control

512KB L2 Cache

To On-Die Interconnect

L1 TLB and 32KB Data Cache

DCache Miss

TLB Miss

16B/Cycle (2 IPC)

4 Threads In-Order

VPU RF

X87 RF

Scalar RF

VPU 512b SIMD

X87

ALU 0

ALU 1

For illustration only.
Vector/SIMD High Computational Density

Future options subject to change without notice.
*Other brands and names are the property of their respective owners
Intel® Xeon Phi™ Coprocessor: Prorgaming Model:

Restricted Architectures

- Run restricted code
- Run restricted architectures

It’s a Supercomputer on a chip

- Operate as a compute node
- Run a full OS
- Run MPI
- Run OpenMP*
- Run x86 code
- Run offloaded code

Custom HW Acceleration

- GPU
- ASIC
- FPGA

Intel® Xeon Phi™ Coprocessor

Restrictive architectures limit the ability for applications to use arbitrary nested parallelism, functions calls and threading models.
Well, it is an SMP-on-a-chip running Linux*
Intel® Xeon Phi™ Environment

Physical View

Logical Views

NATIVE
- Linux
- IP
- SSH
- FTP
- NFS
- ...

OFFLOAD

Heterogeneous

Autonomous
Flexible Execution Models
Optimized Performance for different Workloads

- SINGLE SOURCE CODE
- Compilers, Libraries, Runtime Systems

- MAIN()
- XEON®
- RESULTS

- MAIN()
- XEON®
- RESULTS

- MAIN()
- XEON®
- XEON PHI™
- RESULTS

- MAIN()
- XEON®
- XEON PHI™
- RESULTS

- MAIN()
- XEON®
- RESULTS

- MAIN()
- XEON®
- RESULTS

- Multicore Only
- Multicore Hosted with Many-Core Offload
- Symmetric
- Many-Core Only
Flexible Execution Models
Optimized Performance for different Usage Models

- **NATIVE ONLY**
- **OFFLOAD**
- **CO-WORKER**
- **SYMMETRIC**

*Other brands and names are the property of their respective owners*