Performance Evaluation of an Intel Haswell- and Ivy Bridge-Based Supercomputer Using Scientific and Engineering Applications

[1]Subhash Saini, [2]Robert Hood, [2]Johnny Chang and [3]John Baron
[1]NASA Ames Research Center, [2]CSRA Inc., [3]SGI Inc.
{subhash.saini, robert.hood, johnny.chang}@nasa.gov, jbaron@sgi.com

Abstract — We present a performance evaluation conducted on a production supercomputer of the Intel Xeon Processor E5-2680v3, a twelve-core implementation of the fourth-generation Haswell architecture, and compare it with Intel Xeon Processor E5-2680v2, an Ivy Bridge implementation of the third-generation Sandy Bridge architecture. Several new architectural features have been incorporated in Haswell including improvements in all levels of the memory hierarchy as well as improvements to vector instructions and power management. We critically evaluate these new features of Haswell and compare with Ivy Bridge using several low-level benchmarks including subset of HPCC, HPCG and four full-scale scientific and engineering applications. We also present a model to predict the performance of HPCG and Cart3D within 5%, and Overflow within 10% accuracy.

Keywords: Intel Haswell processor, performance evaluation, benchmarking.

I. INTRODUCTION

Intel's adoption of a "tick-tock" model for design and fabrication of Xeon chips has provided a consistent pattern in the introduction of new hardware for high performance computers [1]. The tocks occur when a new microarchitecture is introduced; the ticks occur when that microarchitecture is then shrunk with a new manufacturing technology. Haswell chips are based on a fourth-generation tock and are implemented in the same 22-nanometer process that was used with Ivy Bridge chips. The Ivy Bridges are the result of a tick, a shrink of the third generation Sandy Bridge "tock" [2,3].

Twelve computing systems out of the top 20 on the November 2015 TOP500 list are based on either Ivy Bridge or Haswell processors, e.g. Ivy Bridge: 1, 15, 16, 17, 19 and 20; Haswell: 6, 8, 9, 13, 14, and 18 [6]. In September 2015, NERSC installed a Cray XC system with 1,630 compute nodes, each with 16-core Haswell processors with a total of 52,160 cores [4]. In October 2015, LANL and SNL installed a Cray XC40 with 301,056 cores based on Haswell [5]. The computing system used in this study has 54,000 cores of Ivy Bridge and 25,056 cores of Haswell [7].

Our survey of the literature yielded no detailed study that evaluates the performance of high performance computing systems based on Ivy Bridge and Haswell processors using real-world and production-quality applications. To the best of our knowledge, this is the first paper to conduct a critical and extensive performance evaluation and characterization of an SGI ICE X cluster based on the Intel Xeon E5-2680v3, hereafter called "Haswell," and Intel Xeon E5-2680v2, hereafter called "Ivy Bridge," using subset of the High Performance Computing Challenge (HPCC) suite, memory latency and bandwidth benchmarks, the High Performance Conjugate Gradient (HPCG) benchmark and four real-world production-quality scientific and engineering applications (Overflow, MITgcm, USM3D, and Cart3D) used for aerospace simulations and climate modeling [8-13]. Specifically, the paper presents:

a. Measurement of the latency and memory load bandwidth of L1 cache, L2 cache, L3 cache, and main memory for Haswell and Ivy Bridge.
b. A performance study of the subset of HPCC benchmark suite on Haswell and Ivy Bridge.
c. An evaluation of the performance of the HPCG benchmark and a model to predict its performance.
d. A performance evaluation using four full-scale applications of AVX2 for Haswell including dynamical changes in AVX and turbo frequency and its comparison with AVX.
e. A performance evaluation of hyper-threading (HT) for Haswell and Ivy Bridge using full-scale applications.
f. Performance models for two real-world production quality applications − Overflow and Cart3D.

The remainder of the paper is organized as follows: Section II provides details of the Ivy Bridge and Haswell systems used in the study; in Section III we briefly describe the benchmarks and applications used; in Section IV we present our results comparing the performance of the two systems; and in Section V we present our conclusions.

II. COMPUTING PLATFORMS

The supercomputer we used is an SGI ICE X system consisting of more than 10,000 nodes interconnected with an InfiniBand (IB) network in a dual-rail hypercube topology [7]. The nodes are based on four different Xeon processors from Intel: Westmere, Sandy Bridge, Ivy Bridge and Haswell. In this study, we used only the Ivy Bridge- and Haswell-based nodes.

A. *Ivy Bridge*

As shown in Figure 1, the Ivy Bridge-based node used in this study has two Xeon E5-2680v2 processors, each with 10 cores. Each processor is clocked at 2.8 GHz, with a peak double-precision floating-point performance of 224 Gflop/s. The total peak performance of the node is therefore 448 Gflop/s. Each core has 64 KB of L1 cache (32 KB data and 32 KB instruction) and 256 KB of L2 cache. All ten cores share 25 MB of last level cache (LLC), also called L3 cache. The on-chip memory controller supports four DDR3 channels running at 1866 MHz, with a peak memory bandwidth per processor of 59.7 GB/s (and twice that per node). Each processor has two QPI links to connect with the

.

other processor in the node to form a non-uniform-memory access (NUMA) architecture. The QPI link runs at 4 GHz or 8 GT/s ("T" for transfer), at which rate 2 bytes can be transferred in each direction, an aggregate of 32 GB/s [7].

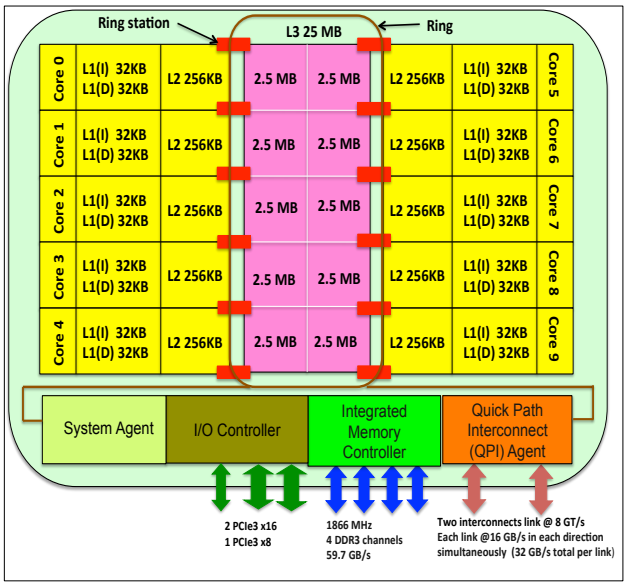

Figure 1. Schematic diagram of a 10-Core Ivy Bridge processor.

B. *Haswell*

As shown in Figure 2, the Haswell-based node used in this study has two Xeon E5-2680v3 processors, each with 12 cores. Each processor is clocked at 2.5 GHz, with a peak performance of 480 Gflop/s. The total peak performance of the node is therefore 960 Gflop/s. Each core has 64 KB of L1 cache (32 KB data and 32 KB instruction) and 256 KB of L2 cache. All 12 cores share 30 MB of L3 cache. The on-chip memory controller supports four DDR4 channels running at 2133 MHz, with a peak memory bandwidth per processor of 68.3 GB/s (and twice that per node). As in the Ivy Bridge node, each processor has two QPI links. In Haswell each of the links runs at 4.8 GHz or 9.6 GT/s, at which rate 2 bytes can be transferred in each direction, for an aggregate of 38.4 GB/s. Each link runs at 19.2 GB/s in each direction simultaneously. As shown in Figure 2 there are two rings with one memory controller (IMC) each. The QPI and Peripheral Component Interconnect Express (PCIe) links are connected to the first ring. The rings are connected with bi-directional queues.

The Intel Haswell microarchitecture is the successor to the Sandy Bridge microarchitecture used in Ivy Bridge processors. The Haswell instruction set architecture (ISA) has been extended to support AVX2 and FMA instructions, which are discussed in detail below. Out-of-order execution is enhanced significantly in the Haswell due to more scheduler and reorder buffer entries, larger register files, and more load/store buffers in order to extract more instruction level parallelism (see Table I).

Haswell's 12-core dies use a partitioned design as shown in Figure 2. Eight cores, six L3 slices of 2.5 MB each, one memory controller, the QPI interface, and the PCIe controller are connected to one bi-directional ring. The remaining four cores, six L3 slices, and the second memory controller are connected to another bi-directional ring. When Cluster-on-Die feature is enabled in the BIOS, the socket is presented by BIOS to the OS as two NUMA nodes with 6 cores each (colored as Cluster 1 and Cluster 2 on the diagram), but it is important to note that cores in Cluster 2 are from two rings. Both rings are connected via two bi-directional queues. When configured in the default snoop mode setting, the ring topology is hidden from the operating system, which exposes all cores and resources in a single NUMA domain.

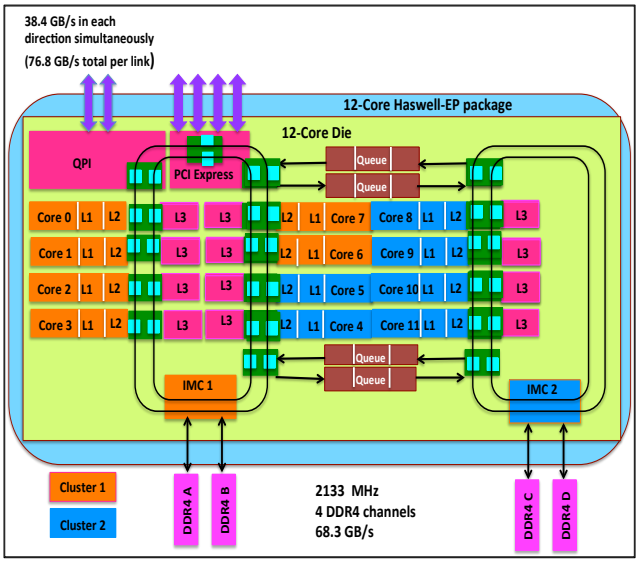

Figure 2. Schematic diagram of a 12-Core Haswell processor.

The following are the new and extended architectural features of Haswell.

AVX2: Ivy Bridge has Advanced Vector Extensions (AVX), a new set of x86 instruction-set extensions of SSE4.2 [18]. It increases the width of the registers from 128 bits to 256 bits; each register can hold eight single-precision floating-point values or four double-precision floating-point values that can be operated on in parallel using SIMD (single-instruction, multiple-data) instructions. The Haswell processor has AVX2 instructions, which enable the newly added, fused floating-point multiply and add (FMA) 256-bit wide SIMD unit and thus can do 16 double-precision floating points in each cycle. Note that when 256-bit instructions are detected, the base frequency may be reduced to the AVX base frequency and the available turbo frequencies are restricted. The degree of downclocking depends on thermal design power (TDP), number of cores, AVX base frequency, workload, etc. The uncore frequency is adapted to the workload dynamically by the hardware. (The "uncore" part of the microprocessor includes functions not in the core, such as the QPI controllers, the L3 cache, the snoop agent pipeline and the on-die memory controller.)

QPI 2.0: In Ivy Bridge, two QPIs connect the two processors/sockets of the node. One is used to form a non-

uniform-memory access (NUMA) architecture to do point-to-point communication; the other connects to the IO hub [19]. For 2 QPI links, the total inter-processor bandwidth is 64 GB/s. For Haswell, the links runs at 9.6 GT/s with total inter-processor bandwidth of 76.8 GB/s to balance out the faster main memory and increased core count.

Memory Subsystem: In Ivy Bridge the memory unit can service three memory requests per cycle, two 16-byte loads and a 16-byte store, for a total of 48 bytes per cycle. The memory unit of Haswell can service two 32-byte loads and a 32-byte store, for a total of 96 bytes per cycle.

Power Management and Turbo Boost: The Ivy Bridge has the same P-states for all the cores whereas Haswell has per core P-states. In addition, in Ivy Bridge the frequency of core and uncore is the same whereas Haswell has independent core and uncore frequencies. That means applications that are bound by memory and cache latency can drive the uncore (the L3 cache and the interconnect rings) faster without boosting the cores; applications that are compute bound can boost their core clocks without having to raise the uncore regions and waste power [17].

Hyper-Threading 2.0: Hyper-Threading (HT) enables two hardware threads to execute simultaneously, filling unused stages in the functional unit pipelines. When one thread stalls, a second thread is allowed to proceed. The advantage of HT is its ability to better utilize processor resources and hide memory latency [15, 16]. Both Ivy Bridge and Haswell support two threads per core, presenting an abstraction of two independent logical cpus. The physical core contains a mixture of resources, some of which are shared between threads.

Memory: Ivy Bridge uses 4 DDR3 channels and Haswell uses 4 DDR4 channels. Memory speed is increased from DDR3 1866 MHz in Ivy Bridge to DDR4 2133 MHz, an increase of 14.3%. The primary advantages of DDR4 over its predecessor include higher module density and lower voltage requirements, coupled with higher data transfer rates. Specifically, DDR4 operates at a voltage of 1.2V compared to 1.5V for DDR3, reducing the power demand. DDR4 has more banks (16 vs. 8) and bank groups (4 vs. none) providing faster burst accesses. The chip densities are 512Mb–8Gb for DDR3 and 4–16 Gb for DDR4 in Haswell. However, latency for DDR4 is higher than DDR3.

C. Network Interconnect (FDR InfiniBand)

The Ivy Bridge and Haswell nodes used in this study are connected to the two fabrics of the machine's InfiniBand (IB) network via the dual-port, four-link fourteen data rate (4x FDR) IB. The Ivy Bridge nodes used in this study have one dual-port HCA, while the Haswell nodes have two single-port HCAs, which are attached to separate PCIeGen3 x8 buses. There is an option to use either single rail or dual rails. The MPI and I/O traffic are separated when MPI uses a single rail, but in dual rails MPI can also send the traffic across both the rails. In this study, MPI was configured to use only one rail. Table I presents the characteristics of Ivy Bridge and Haswell.

TABLE I. CHARACTERISTICS OF IVY BRIDGE AND HASWELL.

Characteristic	Ivy Bridge Node	Haswell Node
Intel Xeon model name	E5-2680v2	E5-2680v3
SIMD ISA	AVX	AVX2
FPU SIMD width	2× 256 bit(1 add, 1 mul)	2× 256 bit FMA
Hyper-Threading (HT)	ON	ON
Turbo Boost	ON	ON
Core clock (GHz)	2.8	2.5
Floating/clock/core	8	16
# Cores per socket	10	12
# Cores per node	20	24
# Processors per node	2	2
Peak perf. (Gflop/s) / core	22.4	40
Peak perf. (Gflop/s) / node	448	960
# Hyper-Threads per core	2	2
# Hyper-Threads per node	40	48
Inter-socket QPI links	2	2
Execute (micro-ops/cycle)	6	8
Retire ((micro-ops/cycle)	4	4
Scheduler entries	54	60
ROB entries	168	192
INT/FP registers	160/144	168/168
Load/store buffers	64/36	72/42
Memory		
L1 cache size	32 KB (I)+32 KB (D)	32 KB (I) + 32 KB (D)
L1 cache associativity	8	8
L1 cache line (byte)	64	64
L1(D) accesses per cycle	2× 16 byte load + 1× 16 byte store	2× 32 byte load + 1× 32 byte store
L2 cache size	256 KB/core	256 KB/core
L2 cache associativity	8	8
L2 cache line (byte)	64	64
L2 bytes/cycle	32	64
L3 cache size (MB)	25 – shared by 10 cores; 10 L3 slices	30 – shared by 12 cores; 12 L3 slices
L3 cache associativity	20	20
L2 cache line (byte)	64	64
L1-L2 bandwidth	32 byte per cycle	64 byte per cycle
L2-L3 bandwidth	32 byte per cycle	32 byte per cycle
L3-MEM peak BW (GB/s)	59.7	68.3
Memory speed (MHz)	4 DDR3 1866	4 DDR4 2133
Quick Path Interconnect	8.0 GT/s or 32 GB/s	9.6 GT/s or 38.4 GB/s
	Throughput	
ADD	1 per cycle	1 per cycle
MUL	1 per cycle	2 per cycle
FMA	Unsupported	2 per cycle
Scalar	2 LD or 1 LD & 1 ST	2 LD & 1 ST
AVX /AVX2	1 LD & ½ ST	2 LD & 1 ST
Inter-node Network		
IB Device on node	Dual-port 4x FDR IB Mezzanine card (with 1 dual-port HCA); 56 Gbits/s	Dual single-port 4x FDR IB Mezzanine card (with 2 single-port HCAs); 56 Gbits/s
IB Switches between nodes	4x FDR; 56 Gbits/s	4x FDR; 56 Gbits/s
Peak performance (Gbits/s)	56	56
Network topology	Hypercube	Hypercube
Network fabric	InfiniBand (IB)	InfiniBand (IB)
System Software		
Compiler	Intel 2015.3.187	Intel 2015.3.187
MPI library	MPT 2.12r26	MPT 2.12r26
Math library	Intel MKL 11.2.3	Intel MKL 11.2.3
Operating system	SLES11sp3	SLES11sp3

.

III. BENCHMARKS AND APPLICATIONS

In this section we present a brief description of the low-level benchmarks and production quality, real-world applications used in this study.

A. Memory Subsystem Latency and Bandwidth

A deep understanding of the performance of the design and operating principles of the microprocessor memory hierarchy of Ivy Bridge and Haswell is crucial to optimize application performance. Toward this end we measured the latency and bandwidth for L1, L2, and L3 caches and main memory for both Ivy Bridge and Haswell using LMbench benchmark suite [14].

B. HPC Challenge Benchmarks (HPCC)

The HPCC benchmarks are intended to test a variety of attributes that can provide insight into the performance of high-end computing systems [8]. These benchmarks examine not only processor characteristics but also the memory subsystem and system interconnects.

C. High Performance Conjugate Gradient

The relevance of High Performance LINPACK (HPL) for the real world is almost negligible as it is basically a matrix-matrix multiplication subroutine called DGEMM which the major HPC vendors like Cray, IBM, and Intel have optimized to as high as 99% floating-point efficiency. In contrast, almost all real-world applications run at less than 10% efficiency. To overcome this drawback and to better align with real-world application performance, there is benchmark called High Performance Conjugate Gradient (HPCG). It is based on an iterative sparse-matrix Conjugate Gradient (CG) kernel where the pre-conditioner is a three-level hierarchical multi-grid method (MG) with Gauss-Seidel relaxation [9]. The CG benchmark from the NAS Parallel Benchmark (NPB) suite shares several attributes with HPCG but CG is non-physical and uses no preconditioning.

D. Science and Engineering Applications

For this study, we used four production-quality, full applications used for aerospace simulations and climate modeling.

1) OVERFLOW-2 is a general-purpose Navier-Stokes solver for CFD problems [10]. The code uses finite differences in space with implicit time stepping. It uses overset-structured grids to accommodate arbitrarily complex moving geometries. The dataset used is a wing-body-nacelle-pylon geometry (DLRF6) with 23 zones and 36 million grid points. The input dataset is 1.6 GB in size, and the solution file is 2 GB.

2) Cart3D is a high-fidelity, inviscid CFD application that solves the Euler equations of fluid dynamics [11]. It includes a solver called Flowcart, which uses a second-order, cell-centered, finite volume upwind spatial discretization scheme, in conjunction with a multi-grid accelerated Runge-Kutta method for steady-state cases. We used the geometry of the Space Shuttle Launch Vehicle (SSLV) for the simulations. The SSLV uses 24 million cells for computation, and the input dataset is 1.8 GB. The application requires 16 GB of memory to run.

3) USM3D is a 3-D unstructured tetrahedral, cell-centered, finite volume Euler and Navier-Stokes flow solver [12]. Spatial discretization is accomplished using an analytical reconstruction process for computing solution gradients within tetrahedral cells. The solution is advanced in time to a steady-state condition by an implicit Euler time-stepping scheme. The test case uses 10 million tetrahedral meshes, requiring about 16 GB of memory and 10 GB of disk space.

4) MITgcm (MIT General Circulation Model) is a global ocean simulation model for solving the equations of fluid motion using the hydrostatic approximation [13]. The test case uses 50 million grid points and requires 32 GB of system memory and 20 GB of disk to run. It writes 8 GB of data using Fortran I/O. The test case is a ¼-degree global ocean simulation with a simulated elapsed time of two days.

IV. RESULTS

In this section we present our benchmarking results starting with an analysis of memory subsystem latency and bandwidth. All experiments were run with Turbo Boost enabled and used only a single IB rail. Hyper-Threading (HT) was also enabled but we used only single thread (ST) in our study except in Section IV-G Performance Impact on Hyper-Threading where we used both ST and HT. The AVX compiler options used were `–xCORE-AVX2` and `–xAVX` on Haswell and Ivy Bridge respectively. All of the benchmarks were run twenty times and average performance is reported. The performance variation from run to run was less than 0.3%.

A. Memory Latency and Bandwidth

Figure 3 shows the memory latency of the Ivy Bridge- and Haswell-based systems. All the performance numbers presented are for a single thread. They show a step function pattern with four steps, corresponding to L1 cache, L2 cache, L3 cache, and main memory. The latency for Haswell is higher than that of Ivy Bridge for all four steps.

a. L1 cache latency is 1.3 ns for Ivy Bridge and 1.4 ns for Haswell.
b. L2 cache latency is 3.7 ns and 3.9 ns for Ivy Bridge and Haswell respectively.
c. L3 cache latency is 12.8 ns for Ivy Bridge and 16.1 ns for Haswell.
d. Main memory latency is 56.5 ns and 88.6 ns for Ivy Bridge and Haswell respectively; i.e. the latency of Haswell is 57% higher than Ivy Bridge. The reason for this is that Haswell uses DDR4, which has high Column Address Strobe (CAS) latency (CL)—the time it takes between the processor asking memory

.

for data and memory returning it. In summary, higher latency is intrinsic to the DDR4 used in the system.

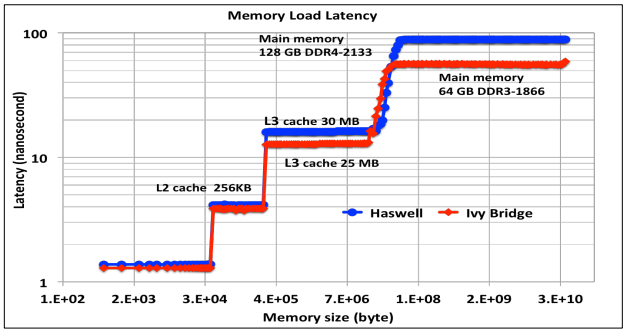

Figure 3. Memory latency of Ivy Bridge and Haswell.

Figure 4 shows the memory load bandwidth of L1, L2, and L3 caches, and main memory for the two systems.

L1 (Data): For L1 (Data) cache, read and write bandwidth is higher on Haswell than on Ivy Bridge by 2.7% to 3.6%.

- Read: Ivy Bridge 14.9 GB/s; Haswell: 15.3 GB/s
- Write: Ivy Bridge 11.2 GB/s; Haswell: 11.6 GB/s

The reason for this is that in Haswell the L1 (D) accesses per cycle are two times that of Ivy Bridge; i.e. Ivy Bridge: 2×16-byte load + 1×16-byte store and Haswell: 2×32-byte load + 1×32-byte store.

L2 cache: For L2 cache read and write bandwidth is higher on Haswell than on Ivy Bridge by 2.7% and 3.8%.

- Read: Ivy Bridge 14.6 GB/s; Haswell: 15.0 GB/s
- Write: Ivy Bridge 10.5 GB/s; Haswell: 10.9 GB/s

The reason for this is that L2 bytes accessed per cycle for Haswell are two times that of Ivy Bridge (32 vs. 64 bytes).

L3 cache: For L3 cache, read bandwidth is 3.5% better on Haswell and write bandwidth is 6.4% better on Ivy Bridge.

- Read: Ivy Bridge 14.2 GB/s; Haswell 14.7 GB/s
- Write: Ivy Bridge 10.8 GB/s; Haswell 10.1 GB/s

Main memory: Read and write bandwidth from main memory is higher on Ivy Bridge than on Haswell by 27.2% and 11.3%, respectively. The higher read bandwidth on Ivy Bridge is consistent with the lower measured memory latency. For a single thread, the memory bandwidth is roughly the number of outstanding memory references times the cache line size divided by the memory latency.

- Read: Ivy Bridge 12.5 GB/s; Haswell 9.1 GB/s
- Write: Ivy Bridge 8.0 GB/s; Haswell 7.1 GB/s

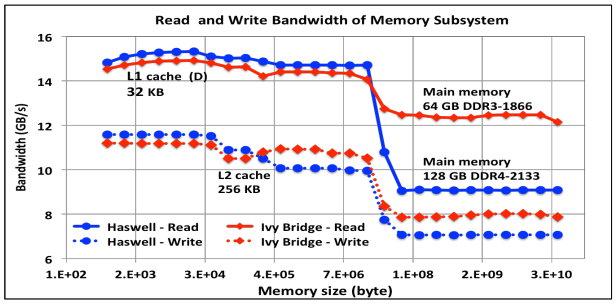

Figure 4. Memory load bandwidth of Ivy Bridge and Haswell.

B. *HPC Challenge Benchmarks (HPCC)*

In this section we present results for the subset of HPCC Version 1.4.1 benchmarks [11] for the two systems. Figure 5 shows performance of the compute-intensive embarrassingly parallel (EP) DGEMM for the two systems. The theoretical one-core peak for Ivy Bridge is 22.4 Gflop/s and for Haswell is 40 Gflop/s. For both Ivy Bridge and Haswell we give the efficiency for their base frequencies of 2.8 GHz and 2.5 GHz, respectively. The efficiency of Haswell was computed using Thermal Design Power (non-AVX) base frequency of 2.5 GHz. The efficiency of Ivy Bridge is higher than that of Haswell—Ivy Bridge 93.2–99.99% and Haswell 75.7–79.4%. Performance of single DGEMM on Haswell is 41.5 to 42.7 Gflop/s. It is clear that Turbo Boost has enhanced the performance by 6.8% as the peak performance on a Haswell core is 40 Gflop/s. Performance on Ivy Bridge ranges from 22.6 to 23.2 Gflop/s, so Turbo Boost increased the performance by 3.6%. The performance gain by Haswell in terms of floating-point operations is 42% to 50% better than Ivy Bridge due to AVX2 instructions, which enable the FMA operations on two ports (256-bit each) as opposed to one port (256-bit) for multiply and one for add (no fused FMA) on Ivy Bridge. As a result, Haswell can perform 16 double-precision floating-point operations per cycle compared to 8 on Ivy Bridge.

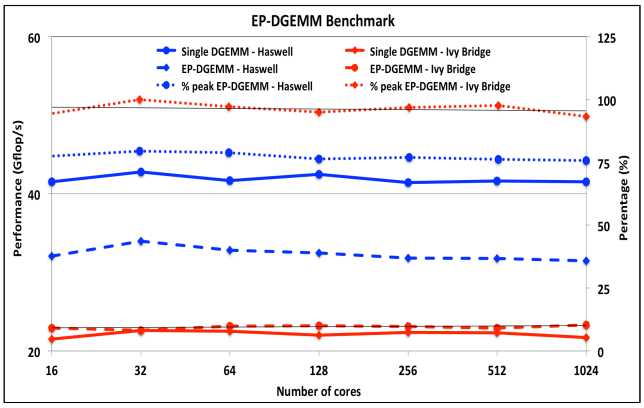

Figure 5. Performance of EP-DGEMM on Ivy Bridge and Haswell.

In Figure 6 we plot performance of the compute-intensive global high-performance LINPACK (G-HPL) benchmark. For both Ivy Bridge and Haswell we give the efficiency relative to their base frequencies of 2.8 GHz and 2.5 GHz, respectively. The efficiency of Ivy Bridge is higher than that of Haswell for the entire range of number of cores – Ivy Bridge: 85–98% and Haswell: 71–76%. The performance gain by Haswell in terms of floating-point operations is 31–46% better than that on Ivy Bridge due to AVX2 instructions and the FMA unit as described in the EP-DGEMM section. As for DGEMM here also we used Thermal Design Power (non-AVX) base frequency of 2.5 GHz to compute the efficiency for Haswell. However, if we use AVX frequency for Haswell then efficiency of LINPACK is 86–92%. Ivy Bridge does not have AVX frequency.

.

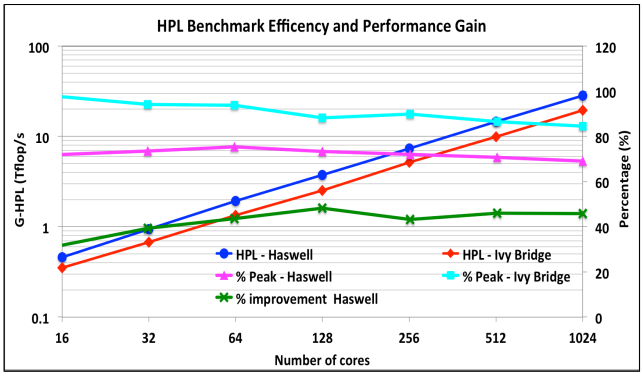

Figure 6. Performance of G-HPL on Ivy Bridge and Haswell.

In Figure 7 we show memory bandwidth for each system using the EP-Stream Triad benchmark. For a single core, the measured bandwidths were 9.9 GB/s and 18.7 GB/s for Ivy Bridge and Haswell respectively, i.e., 88.9 % higher for Haswell due to faster memory speed (DDR3 is 1866 vs. DDR4's 2133 MHz). For 16 cores, these values decreased to 7.4 GB/s and 6.5 GB/s for Ivy Bridge and Haswell due to memory contention. The aggregate node level bandwidth for Ivy Bridge in fully subscribed mode is 5.3 GB/s × 20 cores = 106 GB/s, which translates into 88.8% of the peak-memory bandwidth of 119.4 GB/s. The aggregate node-level bandwidth for Haswell in fully subscribed mode is 5.2 GB/s × 24 cores = 124.8 GB/s, which translates into 91.4% of peak-memory bandwidth of 136.5 GB/s. It should be noted that average EP-Stream bandwidths are 5.3 GB.s and 5.2 GB/s for Ivy Bridge and Haswell, respectively; one may conclude that the performance of memory-intensive applications at a fixed core count would be about the same on both systems.

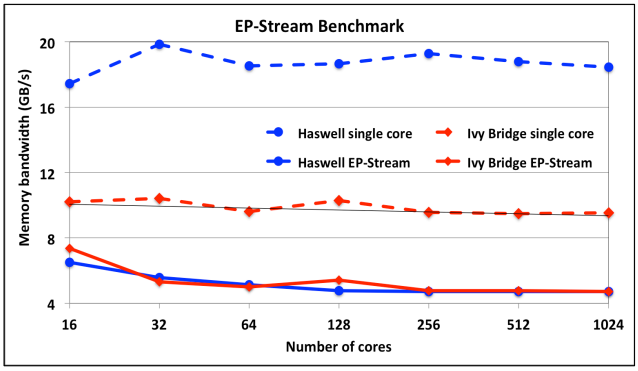

Figure 7. Performance of EP-STREAM on Ivy Bridge and Haswell.

C. *HPCG:*

Figure 8 plots the performance of HPCG benchmark for numbers of cores ranging from 16 to 1024. The performance of HPCG is higher on Ivy Bridge than on Haswell by 6.1–8.4%. The reason for this is that reference implementation of HPCG uses a sparse matrix and cannot reuse the data from caches so it is memory bound. The optimized implementations do try to make more efficient use of the memory subsystem. While STREAM-triad memory bandwidth per core is about the same for both systems, memory latency on Haswell is higher than Ivy Bridge by 57%. In addition, sustained peak performance on Ivy Bridge is two times that of Haswell (2% vs. 1%) because HPCG can't make use of the second FMA unit (i.e. it can't sustain 16 floating-point-operations per cycle).

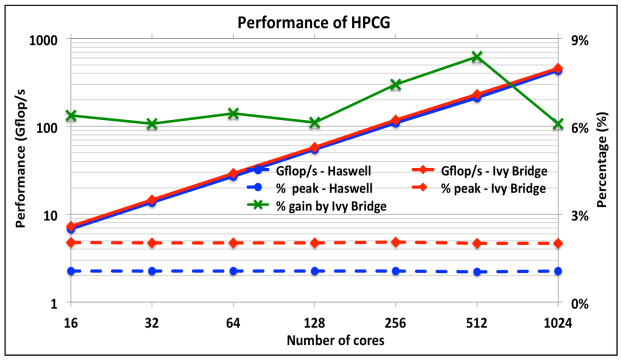

Figure 8. Performance of HPCG on Ivy Bridge and Haswell.

The performance of the HPCG benchmark mainly depends on two parameters: sustained main memory bandwidth and highest occurring interconnect latency. HPCG uses *MPI_Allreduce* with message size of 8 bytes so interconnect bandwidth is irrelevant. As can be seen in the figure the effect of interconnect latency is very small in the range of cores we studied, therefore to a first approximation it can be neglected.

Figure 9 shows the performance of HPCG versus the sustained system memory bandwidth and we see that it is almost a straight line. This figure also shows a linear curve fitting to predict the performance of HPCG by the linear equation $y = 0.0898\ x - 1.9016$, where y is the performance in Gflop/s and x is the memory bandwidth in GB/s, with the coefficient of determination, $R^2 = 0.99981$. This memory-based performance model has prediction accuracy of ±5%.

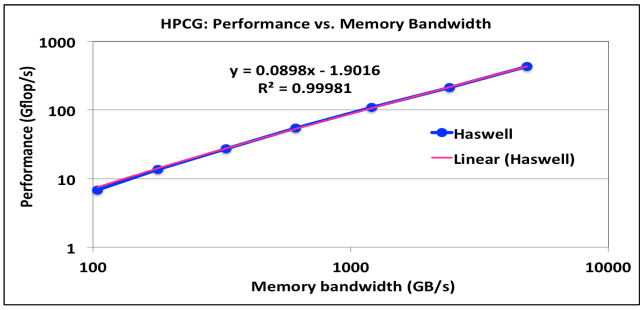

Figure 9. Memory bandwidth model for HPCG on Haswell.

The HPCG benchmark has a 27:1 read:write ratio, which is different from STREAM Triad's. If performance ratios are fixed, it is valid to use STREAM to extrapolate HPCG performance, as it is just a multiplicative constant. We looked at systems from the latest HPCG list, estimated the aggregate STREAM Triad for Xeon-only systems (Sandy Bridge/Ivy Bridge/Haswell), and computed the ratio of HPCG performance to Triad performance. We found that despite differences in HPCG implementation, interconnect

.

fabric, etc., the aggregate STREAM Triad bandwidth can estimate HPCG performance on Xeons within +/- 30% (and in most cases within +/- 13%).

Figure 10 shows the sustained floating-point efficiency of HPL and HPCG for Haswell and Ivy Bridge. For both HPL and HPCG the peak percentage is higher on Ivy Bridge than on Haswell—HPL: 85–98% on Ivy Bridge and 71-76% on Haswell; HPCG: 2.0% on Ivy Bridge and 1.0% on Haswell. We find that efficiency of Ivy Bridge is much better than that of Haswell. It is clear that HPL is not a very useful benchmark for real world applications whose sustained peak percentage performance is in the single digits.

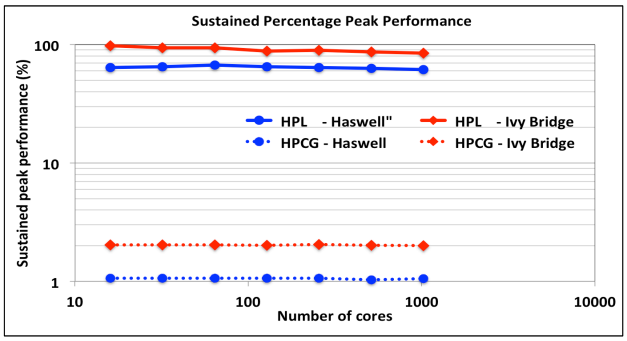

Figure 10. Performance of HPL and HPCG on Ivy Bridge and Haswell the % numbers for HPL are already in Fig 6.

D. *Scientific and Engieering Applications*

In this section we focus on the comparative performance of four full applications (Overflow, Cart3D, USM3D, and MITgcm) [14-17] on the two systems. The reported time for all four applications is for the main loop, i.e., compute and communication time, and does not include I/O time.

Figure 11 shows a summary of the performance gain by Haswell over Ivy Bridge for the four applications. We found that low vectorization of USM3D (20%) and Cart3D (1%) means they can't take advantage of the 256-bit long vector FMA pipe on Haswell. The other two applications have higher vectorization rates with Overflow at 64% and MITgcm at 50%. However, they are memory bandwidth bound. In addition, the length of vector loops in Overflow is short because of 5x5 matrices (pentadiagonal). MITgcm for a large number of cores is interconnect latency bound as it uses *MPI_Allreduce* for 8-byte messages.

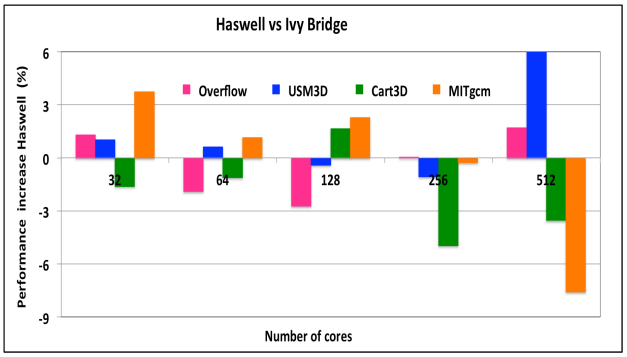

Figure 11. Applications performance on Ivy Bridge and Haswell.

Figure 12 shows the performance models of Cart3D and Overflow. The performance prediction of Cart3D (within 5%) is much better than that of Overflow (within 10%). It clearly shows that the performance of these two applications strongly depend on the memory bandwidth of the computing system. In the model $y = a \times x^b$, y is the run time in second for the application and x is the sustained system main memory bandwidth in GB/s as measured by the EP-Stream benchmark. In both models, the coefficient of determination, R^2, is greater than 99%, indicating that the model fits the performance data very well.

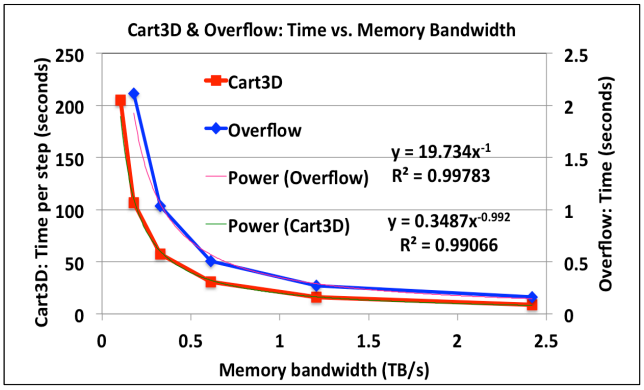

Figure 12. Performance model for Cart3D on Haswell.

E. *Performance Impact of AVX2*

Figure 13 shows the performance gain of AVX2 vs. AVX for the four applications used in this study running on Haswell. It ranges from –3% to +2%. This is because these applications are memory bound and not compute bound so they don't benefit from the AVX2 instructions.

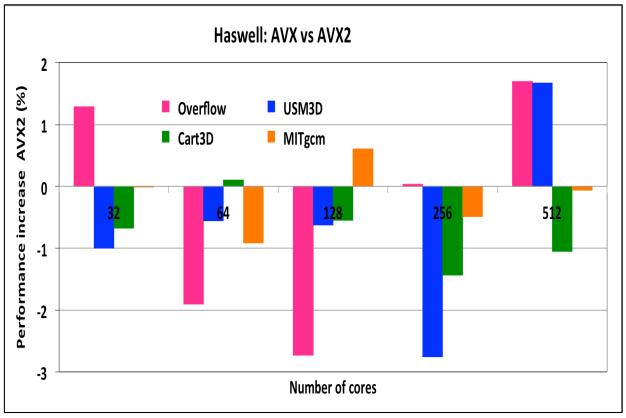

Figure 13. Applications performance on Haswell.

F. *Performance Impact of Hyper-Threading*

In Figure 14 we show the performance gain from Hyper-Threading (HT) by Overflow, Cart3D, USM3D, and MITgcm on Haswell. With HT, the node can handle twice as many processes (48) as without HT (24). With more processes per node, there is greater communication overhead. In other words, more processes compete for the same host channel adapters (HCA) on the node. On the other hand, additional processes (or threads) can steal

.

cycles in cases of communications imbalance or memory access stalls. The result is better overall performance for memory latency-bound applications For example, USM3D, where 70% of the data comes from main memory because of indirect addressing, can't reuse the L2/L3 cache and thus gets an opportunity to hide the memory latency. Cart3D also benefits from HT as the code is 99% scalar and has more opportunities to schedule the instructions in the pipeline. Overflow and MITgcm are 64% and 51% vectorized, respectively, so these two applications do not benefit from HT as there is saturation of floating point units. Another reason why Overflow does not benefit from HT is because it is very cache sensitive. Running in HT mode reduces the amount of L3 cache available to each process, so data has to be fetched from main memory instead of from L3 cache, causing HT performance to suffer. The other reason for reduced HT performance is that in order to realize a gain at e.g. 512 cores, the application has to scale nearly perfectly from 512 MPI processes to 1024 MPI processes; otherwise the HT gains are negated by the lack of scaling.

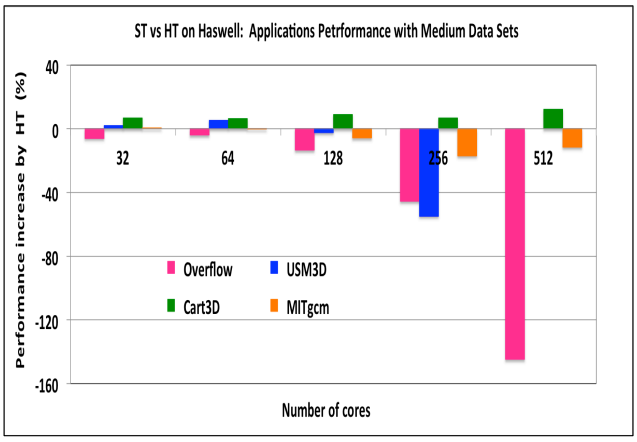

Figure 14. Applications performance gain from HT on Haswell.

V. CONCLUSIONS

In this paper we conducted a comprehensive performance evaluation and analysis of a major Ivy Bridge- and Haswell-based supercomputing platform, using low-level benchmarks and four production applications. Our key findings are as follows:

- The advantage of AVX2 over AVX instructions is insignificant—ranging from –3 to +2% on the four applications.
- The performance improvement due to Hyper-Threading technology on Ivy Bridge and Haswell is minimal on the 4 applications tested.
- The sustained memory bandwidth per core of Ivy Bridge is almost same as Haswell.
- The memory latency of the memory subsystem (main memory, L1/L2/L3 cache) is higher on Haswell than on Ivy Bridge.
- The efficiency of the reference implementation of HPCG is 2% on Ivy Bridge and 1% on Haswell.
- We provided a performance model for the newly introduced HPCG benchmark predicting the performance within 5% on the systems we used.
- We provided a performance model for Overflow and Cart3D, two of most widely used CFD codes used in aerospace industry, predicting the performance within 10% and 5% respectively.
- Our results show that per-core performance of Haswell and Ivy Bridge are very similar. There are architectural differences between the two that might benefit one over the other.

REFERENCES

[1] Intel Tick_Tock Model, www.intel.com/content/www/.../intel-tick-tock-model-general.html

[2] Ivy Bridge: Intel Xeon Processor E5-2600 v2 Product Family - http://www.intel.com/content/www/us/en/intelligent-systems/romley/xeon-e5-v2-c604-c602-j-chipset.html

[3] Haswell: Intel Xeon Processor E5-2600 v3, Product Family: http://www.intel.com/content/www/us/en/processors/xeon/xeon-e5-brief.html

[4] Cori Phase 1 system based on Intel Haswell processor: https://www.nersc.gov/users/computational-systems/cori/cori-phase-i/ , 2015

[5] Trinty Phase 1 system based on Haswell processors: http://www.lanl.gov/projects/trinity/

[6] TOP500 List November 2015; http://www.top500.org/list/2015/11/

[7] NASA Pleiades: http://www.nas.nasa.gov/hecc/resources/pleiades.html

[8] HPC Challenge Benchmarks (HPCC), http://icl.cs.utk.edu/hpcc/

[9] HPCG: *hpcg-benchmark.org/*

[10] Overflow, http://aaac.larc.nasa.gov/~buning/

[11] D. J. Mavriplis, M. J. Aftosmis, and M. Berger. High Resolution Aerospace Applications using the NASA Columbia Supercomputer, in: Proc. ACM/IEEE SC05, Seattle, WA, 2005.

[12] USM3D: http://tetruss.larc.nasa.gov/usm3d/

[13] M.I.T General Circulation Model (MITgcm), http://mitgcm.org/

[14] lmbench: Portable Tools for Performance Analysis, Proceedings of the USENIX 1996 Annual Technical Conference, San Diego, California, January 1996, https://www.usenix.org/legacy/publications/library/proceedings/sd96/full_papers/mcvoy.pdf

[15] Intel Hyper-Threading Technology (Intel HT Technology), http://www.intel.com/technology/platform-technology/hyper-threading/ .

[16] D. Marr, et al., "Hyper-Threading Technology Architecture and Microarchitecture," Intel Technology Journal, Volume 06, Issue 01 February 14, 2002

[17] Intel Turbo Boost Technology 2.0, http://www.intel.com/content/www/us/en/architecture-and-technology/turbo-boost/turbo-boost-technology.html

[18] Overview: Intrinsics for Intel® Advanced Vector Extensions 2 (Intel® AVX2) Instructions, https://software.intel.com/en-us/node/513926.

[19] An Introduction to the Intel , QuickPath Interconnect, January 2009, http://www.intel.com/content/dam/doc/white-paper/quick-path-interconnect-introduction-paper.pdf

.