WHITE PAPER

Analyzing the Performance of STAR-CCM+® Solvers using SGI® ICE™ X and Intel® Xeon® E5-2600 v3

Authors

Dr. Ayad Jassim

SGI Applications Engineering

Abstract

Developed by CD-adapco, STAR-CCM+ is a comprehensive engineering simulation integrated package. It is an entire engineering process for solving problems involving flow (of fluids or solids), heat transfer and stress. In its CFD core, STAR-CCM+ contains (among other solvers) two main solvers, the Segregated Solver and the Coupled Solver. Previously A. Jassim and M. Kremenetsky [5] examined the parallel performance of a number of industrial CFD software codes with respect to Intel® Hyper-Threading Technology and Turbo Boost Technology based on the Intel® Xeon® Processor 5500 Series, (code named Nehalem), as the compute base unit. In this paper we will extend this work to the more recent Intel® Xeon® E5-2960 v3 Processor Family, specifically the Xeon E5-2690 v3 and Xeon E5-2698 v3 processors. The analysis will examine the parallel performance of the STAR-CCM+ Segregated and Coupled solvers with respect to the features of Hyper-Threading Technology and Turbo Boost along with the effect of InfiniBand Single versus Dual-rail in the SGI ICE X architecture. The objective is to provide STAR-CCM+ users with usage guidelines on how each of the two solvers relate to this type of compute hardware.

TABLE OF CONTENTS

1.0 Introduction

Computational Fluid Dynamics (CFD) is a powerful tool for solving a wide variety of industrial and manufacturing problems where commercial general-purpose codes have been developed to solve a very broad spectrum of fluids flow problems. Development work on solver algorithms, meshing, and user interface generation are ongoing, with the objectives of improving accuracy and reducing solution time. CFD is routinely used today in a variety of industries, including aerospace, automotive, power generation, chemical manufacturing, polymer processing, petroleum exploration, medical research, meteorology, and astrophysics. The use of CFD in process industries has led to reductions in the cost of product and process development and optimization activities (by reducing down time), reduced the need for physical experimentation, shortened time to market and improved design reliability.

In this paper we will analyze compute experiments that will help manufacturers using CFD industrial applications maximize their production based on executing CFD simulations on recent compute hardware technologies. The CFD application considered is STAR-CCM+, an industrial CFD process integrated software environment from CD-adapco designed to simplify the analysis of a wide range of flow regimes. It is equipped with a comprehensive selection of physics models along with other features such as built-in surface-wrapping, advanced automated meshing and the ability to “copy and paste” components between models. STAR-CCM+ provides two main solvers (among other solvers), the Segregated Solver and the Coupled Solver for flow simulations. Previously, A. Jassim and M. Kremenetsky [5] examined the parallel performance of a number of industrial CFD software codes with respect to Intel Hyper-Threading Technology [6], where a processor is treated by the operating system as two virtual processors, and Turbo Boost Technology, where a processor monitors the activities of its cores and accordingly boosts the clock speed of some cores when the remaining cores are idle or not fully utilized, based on the Intel® Xeon® Processor 5500 Series as the compute base unit.

Thus, we will extend this work to the SGI® ICE™ X system with the more recent Intel® Xeon® Processors E5-2600 v3 product family. The analysis will examine the parallel performance of the STAR-CCM+ Segregated and Coupled solvers with respect to the features of Hyper-Threading Technology and Turbo Boost along with the effect of InfiniBand Single versus Dual rail in the SGI ICE X architecture. The objective is to provide STAR-CCM+ users with usage guidelines on how each of the two solvers relate to this type of compute hardware. Note that the SGI® ICE™ X system is a compute cluster from Silicon Graphics International Corporation (SGI).

Section 2, presents a brief hardware background for the Intel E5-2600 v3 processor product family in addition to a brief description of the SGI® ICE™ X compute cluster. Section 3 presents the concepts of Intel® Hyper-Threading Technology and Turbo Boost Technology along with definitions and notations that enables proper interpretations of these two features. Section 4 presents a brief introduction to the STAR-CCM+ Segregated and Coupled solvers. Section 5 presents experiment results for tests involving four input models. These tests are based on running the experiments in a standard, hyper-threading and combined hyper-threading with turbo boost modes of execution using Single and Dual rail InfiniBand® configuration of the SGI ICE X cluster. Section 6 presents overall comparisons and conclusions based on the experiments results of Section 5. A summary is a presented towards the end of this paper.

2.0 Hardware Background

2.1 Overview of the Intel® Xeon® Processor E5-2600 v3 Product Family

The Intel® Xeon® processor E5-2600 v3 product family, is a 2-socket platform based on Intel's most recent microarchitecture. Figure 1 provides an overview of this processor product family microarchitecture. Processors in this family have up to 18 cores. They also have additional cache where the top-bin SKU, the Intel® Xeon® E5-2699 v3 has 45 MB. Some of the new features that come with the Intel Xeon processor E5-2600 v3 product family include:

1. Intel® Advanced Vector Extensions 2 (Intel® AVX2) instructions
2. Haswell New Instructions (HNI)
3. Support for DDR4 memory
4. Power Management feature improvements

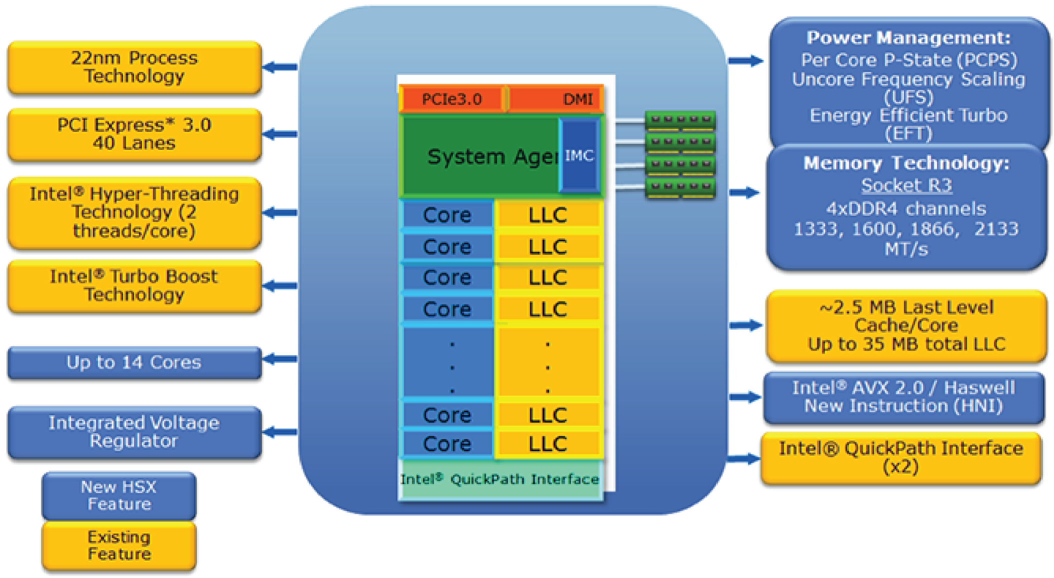


Figure 1: Intel Xeon E5-2600 v3 Processor. © 2015 Intel Corporation

Note that this processor is a follow on to its predecessor the Intel® Xeon® E5-2600 v2 processor [2]. The following listing is a comparison of the Intel® Xeon® processor E5–2600 v3 product family to the Intel® Xeon® processor E5–2600 v2 product family

Feature	Intel® Xeon® E5-2600 v2	Intel® Xeon® E5-2600 v3
QPI Speed (GT/s)	8.0, 7.2 and 6.4	6.4, 8.0, 9.6
Cores	Up to 12	Up to 18
Threads Per Socket	Up to 24 threads	Up to 28 threads
Last-level Cache (LLC)	Up to 30mb	Up to 35mb
Intel® Turbo Boost Technology[1]	Yes	Yes
Memory Population	4 channels of up to 3 RDIMMs, 3 LRDIMMs or 2 UDIMMs	4 channels up to 3 DIMMs per channel and 24 DIMM slots (4 x DDR4)
Max Memory Population	Up to 1866	Up to 2133
PCIe* Lanes / Controllers/Speed (GT/s)	40 / 10 (PCIe* 3.0 at 8 GT/s)	40 / 10 (PCIe* 3.0 at 10 GT/s)
TDP (W)	150 (Workstation only), 130, 115, 95, 80, 70, 60, 50	160, 145, 130 (Workstation only), 135, 120, 105, 85, 80, 70
Idle Power Targets (W)	10.5W or higher, 7.5W for LV SKUs	9W or higher, 7.5W for LV SKUs

The experiments in the paper are based on the E5-2690 v3 and the E5-2698 variants of this processor. For more details on these product families may be found in [1] and [2].

2.2 The SGI® ICE™ X System Overview

The SGI Integrated Compute Environment (ICE) X systems is an integrated compute node (blade) environment that can include thousands of compute nodes. The system can be configured with compute nodes comprised of Intel® Xeon® processor E5-2600 v3 series exclusively or with compute nodes comprising of both Intel® Xeon® processor E5-2600 v3 series and Intel® Xeon Phi™ coprocessors. The system can operate with SUSE® Linux® Enterprise Server and Red Hat® Enterprise Linux operating systems. In the ICE X each 42U SGI rack holds one or two 21U-high (blade enclosure pairs). An enclosure pair consists of two 18-blade enclosures. The enclosures are separated by two power “shelves” each holds three power supplies. Each enclosure also has an internal InfiniBand communication backplane, Figure 2a.

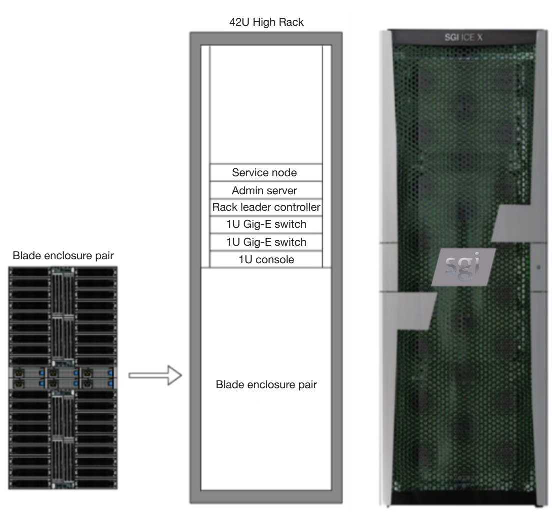


Figure 2a: SGI ICE X IRU and Rack diagram.

Figure 2b presents an overall diagram of the SGI ICE X nodes components. A single service node can provide both login and batch services. A leader node is connected to blades in its rack via the GigE VLAN. It is connected to all blades and service nodes via a InfiniBand fabric. Users access compute nodes from the service nodes and jobs are started on compute nodes using commands on the service node.

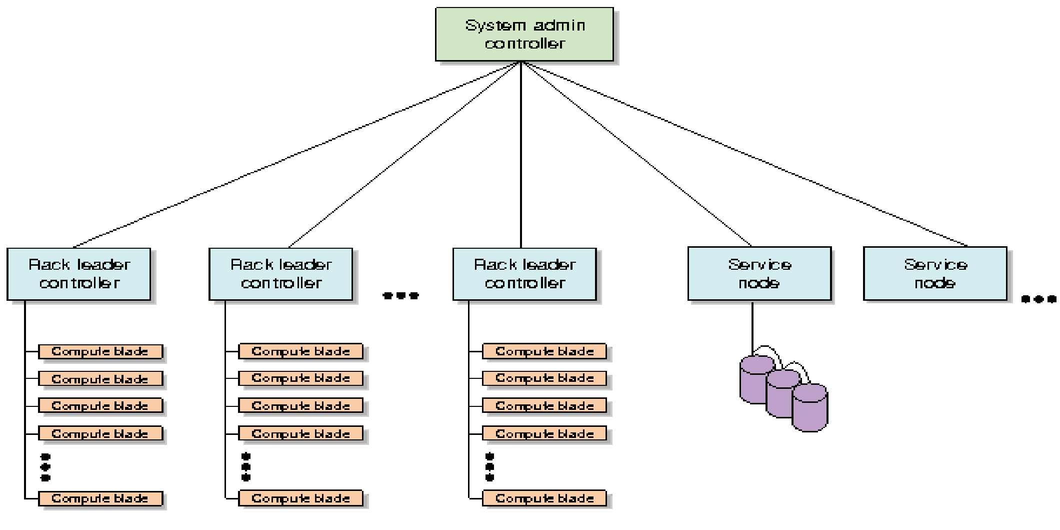


Figure 2b: An overall diagram of the SGI ICE X nodes components

The ICE X configuration considered in this paper comprises a cluster with Intel® Xeon® 12-core 2.6 GHz E5-2690 v3 and 16-core 2.3 GHZ E5-2698 v3 processors, an integrated InfiniBand FDR interconnect Hypercube/Fat Tree, 4 GB of memory per core with memory speed of 2133 MT/s and SUSE® Linux Enterprise Server 11 SP3 operating system. For more details on the SGI ICE X hardware architecture refer to SGI technical manuals [3] and [4].

3.0 Intel® Hyper-Threading Technology

Hyper-Threading Technology (HTT) is an Intel patented technique for simultaneous multithreading (SMT) [6] in which some execution elements are duplicated, specifically elements that store the executional state while elements that actually do the execution are not duplicated. This means that only one processor is physically present but the operating system sees two virtual processors, and shares the workload between them. Units such as L1 and L2 cache, and the execution engine itself are shared between the two competing threads. A processor may stall due to a cache miss, branch misprediction, data dependency or the completion of an I/O operation. This allows another thread to execute concurrently on the same core taking advantage of such idle periods. Hyper-threading requires both operating system and CPU support. Thus a compute node with, say, 2x12 cores per (that is a total of 24 physical cores) can scale to a total of 48 virtual cores using Hyper-Threading Technology.

Experiments have shown that some CFD applications have benefitted as much as 14% from HTT providing an approach in gaining extra performance. However different software can have diverse execution profiles, thus one will expect that HTT to have a variable effect on different applications. For CFD software, HTT benefits will not only depend on the fact that software has different coding techniques, but also different CFD capabilities and also the different options and features of various CFD input models. Generally we may classify a software implementation of HTT in the following two approaches,

1. Single job in an HTT environment: This implies that a job's threads execute on all virtual HTT cores of the compute nodes it requires.
2. Throughput jobs in an HTT environment: In this case more than one job may share the HTT virtual cores of each compute node. This may also apply on different applications running simultaneously by sharing the HTT virtual cores.

In this paper we will focus on experiments using the first approach only by focusing on STAR-CCM+ as the underlying application. In either of the above two approaches, a performance criteria may be devised as follows:

Consider a homogenous cluster in which submitted jobs are assumed to spawn equal number of executing threads per compute node, thus

let *'n'* and *'m'* be two integers, where *'n'* represents the number of physical cores in a compute node or blade and *'m'* being the total number of virtual cores in a compute node/blade. That is *'m'* is a multiple of *'n'* such that this multiple represents the number times a physical core is hyper-threaded. Thus let

'S_n' denote the elapsed time for the *n*-thread per node job in a standard non HTT and none Turbo Boost mode of operation.

And let

'H_m' denote the elapsed time for an *m*-thread per node job under HTT mode of operation.

A performance criteria for an application to benefit from HTT must be such that

$H_m < S_n$; *for n >0, m a multiple of n.*

A typical sequence of *'m'* and *'n'* is where *'m=2n'* and *'n'* is the number of physical cores in a compute node.

In this paper the experiments will be based on compute hardware with a value of *'m =2n'* that is each physical core is hyper-threaded twice, thus the total number of hyper-threaded cores per compute node is *'2n'*.

One further note on Hyper-Threading technology is that one will expect this technique to be limited by a parallelism threshold beyond which a CFD model's parallel scalability will no longer scale. For example, given a CFD model and assuming that this model will normally (i.e. without HTT) scale up to 64 parallel threads where beyond this number will result in either the same or slower elapsed times. Thus, the parallel scalability for this model using 64 HTT parallel threads will be also limited by this observation.

4.0 Turbo Boost and Combined Turbo Boost with Hyper-Threading Technology

In turbo mode, the processor monitors the activities of its cores and accordingly boosts the clock speed of some cores when the remaining cores are idle or less than fully utilized. In order to properly address performance data based on turbo mode and turbo mode combined with hyper-threading technology, we need to define the following performance measurement variables. Thus let

T_n denote the elapsed time for an *n*-thread per node job under Turbo boost mode of operation.

And let C_m denote the elapsed time for an *m*-thread job under the combined Turbo boost and HTT modes of operations.

As defined in section 3, *'m'* and *'n'* are two integers where $n > 0$ and *'m'* a multiple of *'n'* such that this multiple represents the number of hyper-threads per physical core.

Thus, C_m is the job elapsed time under using Turbo boost mode and hyper-thread features combined together. Hence an overall performance criteria for an application to benefit from HTT, Turbo boost and the combined two features must be such that

$$C_m < \{T_n , H_m\} < S_n \quad ; \text{for } n > 0, \ m \text{ a multiple of } n.$$

Recall that H_m is the job elapsed time based only on HTT feature as defined in section 3. Note also that in the above criteria the relation between 'H_m' and 'T_n' generally cannot be determined beforehand. This is because the individual effect of Turbo boost mode compared to HTT will be application dependent. For example, an application may benefit from Turbo boost mode more than from HTT while this relation may be reversed for another application. Again, a typical sequence of *'m'* and *'n'* is where *'m=2n'* and *'n'* being a the number of physical cores in a compute node. In order to establish an approximate relationship between the metrics S_n, H_{2n}, T_n and C_{2n}, let ΔH_{2n} and ΔT_n be the actual absolute gains of H_{2n} and T_n respect to S_n respectively. That is

$$\Delta T_n \ , = \ S_n, - T_n \quad \text{and} \quad \Delta H_{2n} = S_n - H_{2n}$$

If accepting that C_{2n} is the absolute simultaneous gain of both H_{2n} and T_n with respect to S_n given by the following relationship

$$C_{2n} = S_n - \Delta T_n - \Delta H_{2n}$$

Then,

$$C_{2n} = T_n + H_{2n} - S_n$$

The relationship may help to act as an approximation in the case of actual experiments which we will observe in the experiments described and analysed in sections 5 and 6.

5.0 STAR-CCM+ Segregated and Coupled Solvers

STAR-CCM+ is a process oriented CAE software to solve for multi-disciplinary problems within a single integrated environment. A flexible approach to parallel processing with client-server architecture. The software is deployed as a client that handles the user interface and visualization, and a server which performs the compute operations. At the heart of the STAR-CCM+ software are the flow solvers of which, among other solvers are, the Segregated and Coupled solvers.

The Segregated Flow model solves the flow equations (one for each component of velocity, and one for pressure) in a segregated, or uncoupled manner. The linkage between the momentum and continuity equations is achieved with a predictor-corrector approach. The Segregated Flow solver controls the solution update for the Segregated Flow model according to the SIMPLE algorithm. It controls two additional solvers:

- **Velocity solver:** The velocity solver controls the under-relaxation factor and algebraic multigrid parameters for the momentum equations. More specifically, it solves the discretized momentum equation to obtain the intermediate velocity field.
- **Pressure solver:** The pressure solver controls the under-relaxation factor and algebraic multigrid parameters for the pressure correction equation. More specifically, it solves the discrete equation for pressure correction, and updates the pressure field.

The Coupled Flow model solves the conservation equations for mass, momentum, and energy simultaneously using a pseudo-time-marching approach. One advantage of this formulation is its robustness for solving flows with dominant source terms, such as rotation. Another advantage of the coupled solver is that CPU time scales linearly with cell count; in other words, the convergence rate does not deteriorate as the mesh is refined. In addition to the Coupled Flow model, the Coupled Energy model is an extension of the Coupled Flow model, where together they solve the conservation equations for mass, momentum, and energy simultaneously using a time- (or pseudo-time-) marching approach. This formulation is robust for solving compressible flows and flows with dominant source terms, such as buoyancy. Note that The Coupled Energy model is available for selection only after the Coupled Flow model has been selected.

The Coupled Implicit solver controls the solution update for the Coupled Flow model. If the Coupled Energy Model and the Coupled Species Model are activated, the Coupled Implicit solver controls these models also. The solver is used for implicit spatial integration in both steady and unsteady analyses, using a coupled algebraic multi-grid method.

The Coupled Explicit solver controls the solution update for the Coupled Flow model and Coupled Energy model. It is used for explicit integration using a Runge-Kutta multi-stage scheme. The Coupled Explicit node represents this solver in the Solvers node.

In general, when choosing between the Segregated Flow model and the Coupled Flow model, one should consider the following points:

- The segregated algorithm uses less memory than the coupled.
- The number of iterations that the coupled algorithm requires to solve a given flow problem is independent of mesh size. However, the number of iterations that the segregated algorithm requires increases with mesh size.
- The coupled algorithm yields more robust and accurate solutions in compressible flow.
- The coupled algorithm is more robust for High-Rayleigh number natural convection.
- The Segregated Flow model may be suitable for incompressible or mildly compressible flows.

In this paper we will present and analyze a different view on the choice between these two solvers using the Intel E5-2600 v3 architectural features of Hyper-Threading Technology and Turbo Boost in addition to the InfiniBand single and dual rail effects of the SGI® ICE™ X clusters. Refer to the STAR-CCM+ user guide [7] for more information on the Segregated and Coupled solvers and further details on STAR-CCM+.

6.0 STAR-CCM+ Parallel Performance

In this section we will examine the effects of Hyper Threading and the combined Hyper-Threading with Turbo Boost features both separately and then applied simultaneously within the same test. In order to facilitate this analysis, four models will be considered, namely the Mercedes A-Class 6M Model, the LeMansCar 17M Model, the LeMansCar 94M Model and a large Racing Car model. All four models are steady state incompressible external airflow simulations around a car body.

Table 1 shows the model sizes and number of iterations along with the STAR-CCM+ release used for these models. Furthermore, each model will be examined with respect to the STAR-CCM+ solvers, namely the Segregated the Coupled flow solvers. These models have been run on the SGI ICE X compute cluster using STAR-CCM+ 9.04.011 and 9.06.011 releases. Tests were made in two groups based on the two solvers above. For each number of compute nodes tests were carried out as Standard (S_n), Hyper-threaded (H_{2n}) and combined Hyper-threaded –Turbo Boost (C_{2n}) modes of operation. All tests used SGI MPI message passing toolkit. Note that in order to reduce the amount of presented data, the effect of Turbo boost (T_n) will not be presented as a separate entity but rather will be observed in the result values the combined Hyper-threaded – Turbo Boost (C_{2n}) experiment.

Model Name	Number Cells	Number Iterations	STAR-CCM+ Release
Mercedes A-Class 6M	6 Million	100	9.06.011
The LeMansCar 17M	17 Million	500	9.04.011
LeMansCar 94M	94 Million	500	9.06.011
Racing Car	220 Million	10	9.04.011

Table 1: Example models

6.1 Mercedes A-Class 6M Model

An external air flow simulation around the Mercedes Benz A-Class car. The number of cells is 5,914,426. The fluid flow is turbulent incompressible High Reynolds K-Epsilon model. Figure 3.

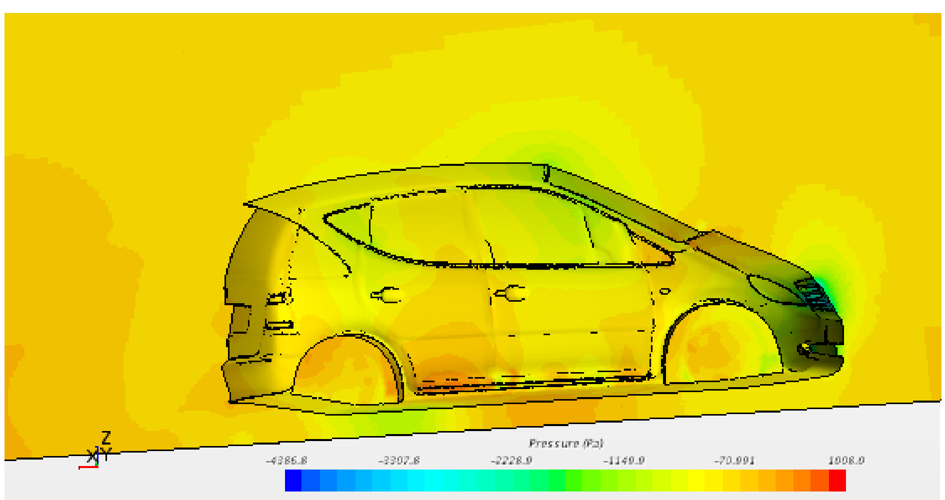


Figure 3: The Mercedes A-Class 6M Car model mesh and post-processing

Table 2 presents InfiniBand Single and Dual rail average solver elapsed time per iteration, S_n, using Segregated solver for tests performed on the SGI ICE X cluster with 2x16 core Intel E5-2698 v3 @ 2.3GHz sockets along with percentage gains values $\%H_{2n}$ and $\%C_{2n}$ (n=32) for single and dual InfiniBand respectively. Similarly Table 3 presents the equivalent results for the Coupled solver. All tests used STAR-CCM+ release 9.06.011 with SGI MPI message passing library.

Single Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	0.4628	0.4299	0.4151	7.11	10.31
18	0.2518	0.2490	0.2241	1.11	4.37
36	0.1513	0.1757	0.1714	-16.13	-13.28
72	0.1082	0.1602	0.1555	-48.06	-43.71
144	0.1033	0.1042	0.1050	-0.87	-1.65

Dual Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	0.4655	0.4311	0.4166	7.39	10.51
18	0.2474	0.2432	0.2323	1.70	6.10
36	0.1464	0.1608	0.1556	-9.84	-6.28
72	0.0958	0.1340	0.1286	-39.87	-34.24
144	0.0861	0.0668	0.6610	22.42	23.23

Table 2: Mercedes A-Class 6m Model InfiniBand Single & Dual Rail Segregated solver average elapsed time/Iteration S_n, H_{2n}, T_n and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2698 v3 @ 2.3GHz.

Single Rail

#nodes n=32	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	0.4994	0.4609	0.4483	7.71	10.23
18	0.2713	0.2643	0.2553	2.58	5.90
36	0.1639	0.1842	0.1795	-12.39	-9.52
72	0.1145	0.1659	0.1609	-44.89	-40.52
144	0.1070	0.1087	0.1084	-1.59	-1.31

Dual Rail

#nodes n=32	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	0.4977	0.4615	0.4441	7.27	10.77
18	0.2645	0.2594	0.2473	1.93	6.50
36	0.1559	0.1695	0.1637	-8.72	-5.00
72	0.1037	0.1398	0.1327	-34.81	-27.96
144	0.0933	0.0737	0.0718	21.01	23.04

Table 3: Mercedes A-Class 6m Model InfiniBand Single & Dual Rail Coupled solver average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2698 v3 @ 2.3GHz.

6.2 The LeMansCar 17M Model

The LeMansCar 17M model is a steady state external airflow model with a mesh size of approximately 17 million cells, figure 4.

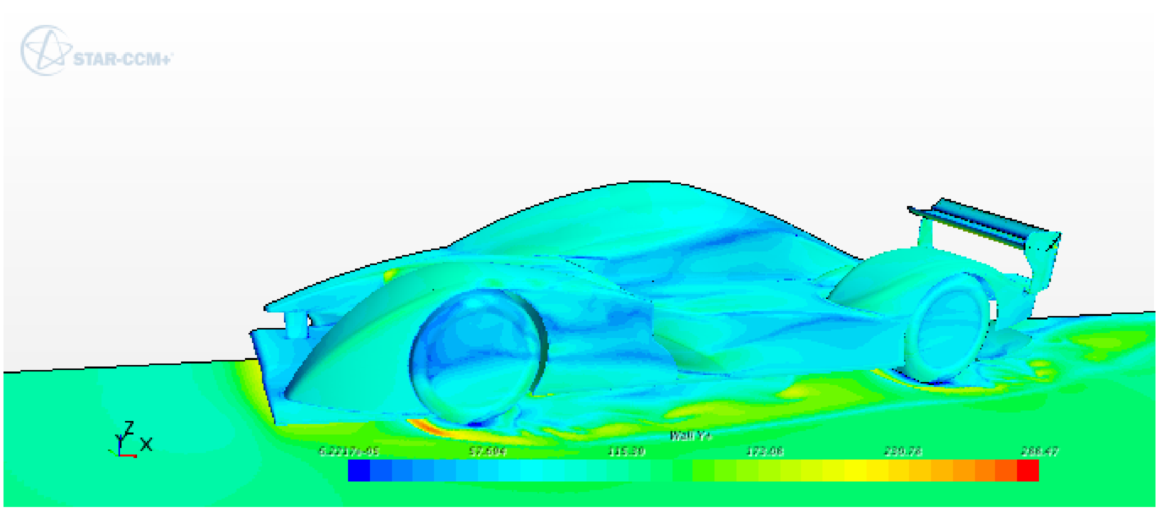


Figure 4: The LeMansCar Model mesh and post-processing

Table 4 presents InfiniBand Single and Dual rail average solver elapsed time per iteration, S_n, using Segregated solver for tests performed on the SGI ICE X cluster with 2x12 core Intel E5-2690 v3 @ 2.6GHz sockets along with percentage gains values $\%H_{2n}$ and $\%C_{2n}$ (n=24) for single and dual InfiniBand respectively. Similarly, Table 5 presents the equivalent results for the Coupled solver. All tests used STAR-CCM+ release 9.04.011 with SGI MPI message passing library.

Single Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	1.3861	1.2355	1.1562	10.86	16.59
18	0.7302	0.6672	0.6227	8.63	14.72
36	0.3917	0.3781	0.3582	3.47	8.55
72	0.2338	0.2561	0.2303	-9.54	1.50
144	0.1506	0.1760	0.1673	-16.87	-11.09
216	0.1241	0.1565	0.2012	-26.11	-62.13

Dual Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	1.3690	1.2288	1.1525	10.24	15.81
18	0.7162	0.6513	0.6174	9.06	13.79
36	0.3859	0.3580	0.3399	7.23	11.92
72	0.2186	0.2170	0.2059	0.73	5.81
144	0.1370	0.1517	0.1417	-10.73	-3.43
216	0.1139	0.1314	0.1868	-15.36	-64.00

Table 4: LeMansCar 17M Model InfiniBand Single & Dual Rail Segregated solver Average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2690 v3 @ 2.6GHz.

Single Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	1.8931	1.5809	1.5008	16.49	20.72
18	0.9517				
36	0.4990	0.4141	0.3870	17.01	22.44
72	0.2581	0.2233	0.2086	13.48	19.18
144	0.1464	0.1325	0.1248	9.49	14.75
216	0.1085	0.1029	0.0977	5.16	9.95

Dual Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	1.8741	1.5817	1.5048	15.60	19.71
18	0.9498	0.8034	0.7478	15.41	21.27
36	0.4843	0.4143	0.3842	14.45	20.67
72	0.2618	0.2189	0.2031	16.39	22.42
144	0.1441	0.1249	0.1157	13.32	19.71
216	0.1057	0.0948	0.0893	10.31	15.52

Table 5: LeMansCar 17M Model InfiniBand Single & Dual Rail Coupled solver average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2690v3@2.6GHz.

6.3 The LeMansCar 94M Model

The LeMansCar 94M Model example is a further mesh refinement of the LeMansCar 94M Model of figure 4 with a mesh size of approximately 94 million cells. Table 6 presents InfiniBand Single and Dual rail average solver elapsed time per iteration, S_n, using Segregated solver for tests performed on the SGI ICE X cluster with 2x16 core Intel E5-2698 v3 @ 2.3GHz sockets along with percentage gains values $\%H_{2n}$ and $\%C_{2n}$ (n=32) for single and dual InfiniBand respectively. Similarly, Table 7 presents the equivalent results for the Coupled solver. All tests used STAR-CCM+ release 9.06.011 with SGI MPI message passing library.

Single Rail

#nodes n=32	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	3.9269	3.4746	3.3979	11.52	13.47
18	1.8501	1.7564	1.7178	5.06	7.15
36	1.0156	1.0199	1.0047	-0.42	1.44
72	0.5500	0.5334	0.5227	3.02	4.96
144	0.3162	0.3348	0.3277	-5.88	-3.64

Dual Rail

#nodes n=32	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	3.9190	3.4743	3.4045	11.35	13.13
18	1.8511	1.7495	1.7048	5.49	7.90
36	0.9954	0.9713	0.9472	2.42	4.84
72	0.5282	0.4830	0.4702	8.56	10.98
144	0.2857	0.2855	0.2764	0.07	3.25

Table 6: LeMansCar 94M Model InfiniBand Single & Dual Rail Segregated solver average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2698 v3 @ 2.3GHz.

Single Rail

#nodes n=32	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	4.7125	4.1931	4.1000	11.02	13.00
18	2.3576	2.0924	2.0605	11.25	12.60
36	1.1924	1.0486	1.0165	12.06	14.75
72	0.5954	0.5269	0.5118	11.50	14.04
144	0.3166	0.2796	0.2707	11.69	14.50

Dual Rail

#nodes n=32	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	4.7175	4.1962	4.0933	11.05	13.23
18	2.3678	2.0941	2.0422	11.56	13.75
36	1.1827	1.0474	1.0187	11.44	16.40
72	0.5901	0.5210	0.5063	11.71	14.20
144	0.3140	0.2733	0.2646	12.96	15.73

Table 7b: LeMansCar 94M Model InfiniBand Single & Dual Rail Coupled solver average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2698 v3 @ 2.3GHz.

6.4 A Racing Car Model

To further observe the effects of Hyper-Threading and Turbo boost features, a Racing Car model has been selected as the fourth example in this paper. Assume that there is a steady state of external airflow around the car model. The size of the model is approximately 220 million cells, figure 5.

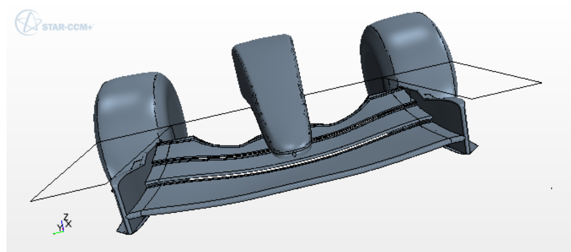


Figure 5: A Racing Car Model

Table 8 presents Single and Dual InfiniBand rail average solver elapsed time per iteration, S_n, using Segregated solver for tests performed on the SGI ICE X cluster with 2x12 core Intel E5-2690 v3 @ 2.6GHz along with percentage gains values $\%H_{2n}$ and $\%C_{2n}$ (n=24) for single and dual InfiniBand respectively. Similarly, table 9 presents the equivalent results for the Coupled solver. All tests used STAR-CCM+ release 9.04.011 with SGI MPI message passing library.

Single Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	9.7211	9.4182	8.9637	3.12	7.79
18	5.2556	5.0282	4.8476	4.33	7.76
36	2.8350	2.7437	2.6477	3.22	6.61
72	1.5859	1.6279	1.5618	-2.65	1.52
144	0.9224	1.0855	1.0449	-17.68	-13.28
288	0.6491	-	-	-	-

Dual Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	9.6540	9.3740	9.0171	2.90	6.60
18	5.2487	4.9202	4.6802	6.26	10.83
36	2.7987	2.6210	2.4836	6.35	11.26
72	1.5056	1.4958	1.4043	0.65	6.73
144	0.8517	0.9242	0.8860	-8.51	-4.03
288	0.5385	-	-	-	-

Table 8: A Racing Car InfiniBand Single & Dual Rail Segregated solver average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2690 v3 @ 2.6GHz.

Single Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	14.9632	13.2559	12.6875	11.41	15.21
18	7.8941	6.8102	6.4116	13.73	18.78
36	4.0780	3.5991	3.5983	11.74	11.76
72	2.2366	2.0107	1.8926	10.10	15.38
144	1.2880	1.1922	1.1354	7.44	11.85
288	0.8077	0.7908	-	2.09	-

Dual Rail

#nodes n=24	S_n	H_{2n}	C_{2n}	$\%H_{2n}$	$\%C_{2n}$
9	14.7652	13.2112	12.7189	10.52	13.86
18	7.8790	6.7899	6.4640	13.71	17.85
36	4.0367	3.5701	3.4046	11.56	15.66
72	2.2319	1.9822	1.8631	11.19	16.52
144	1.3095	1.1707	1.1154	10.60	14.82
288	0.8076	0.7746	0.7403	4.09	8.33

Table 9: A Racing Car InfiniBand Single & Dual Rail Coupled solver average elapsed time/Iteration S_n, H_{2n} and C_{2n} (seconds) with percentage gains $\%H_{2n}$ and $\%C_{2n}$ on SGI ICE X cluster with Intel's E5-2690 v3 @ 2.6GHz.

7.0 Comparisons and Conclusions

In Section 5, we presented experiment results for four input models. For each model we produced the measurements for the metrics S_n, H_{2n}, C_{2n}, $\%H_{2n}$ and $\%C_{2n}$ where the latter percentages represent the actual performance percentage gains made in the corresponding measurements H_{2n} and C_{2n}. These experiments were made using Single and Dual rail InfiniBand configuration of the SGI ICE X cluster. Figures 6a, 6b, 6c and 6d present plots of Hyper-Threading gains, $\%H_{2n}$, for the four input models using the Segregated solver Single rail, Segregated solver Dual rail, Coupled solver Single rail and Coupled solver Dual rail respectively. These figures describe the trends of Hyper-Threading gains for the four models. Comparing the trends for the four input models it seems that Hyper-Threading gains are relatively the lowest for the Mercedes A-Class 6m Model which is the smallest size model (approximately 6M cells). However for the other three models, namely the LeMansCar 17M, LeMansCar 94M and the Racing Car, the corresponding gains appear to be of similar trends despite the significant differences in the sizes of the three models. This indicates that Hyper-Threading gains are limited by the compute node's resources such as memory bandwidth and cache sizes.

Figure 6a: Segregated Solver Single Rail %H2n gains on ICE X for the four models

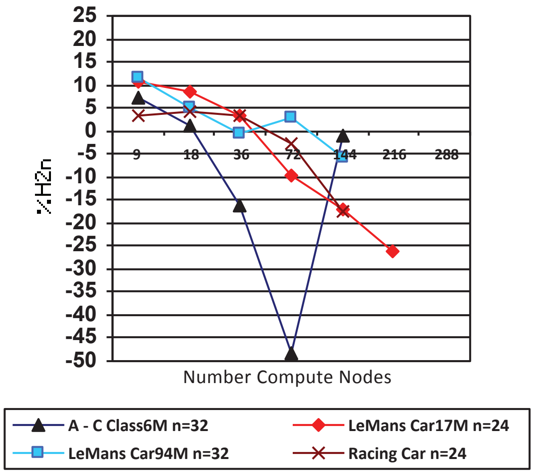


Figure 6b: Segregated Solver Dual Rail %H2n gains on ICE X for the four models

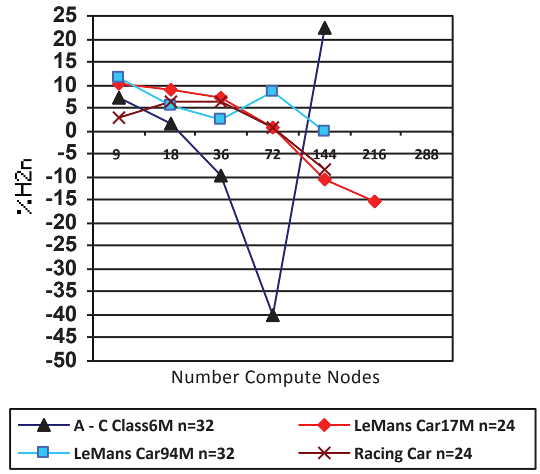


Figure 6c: Coupled Solver Single Rail %H2n gains on ICE X for the four models

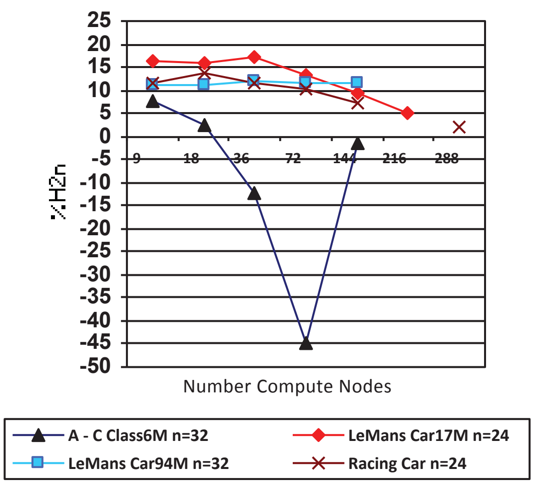
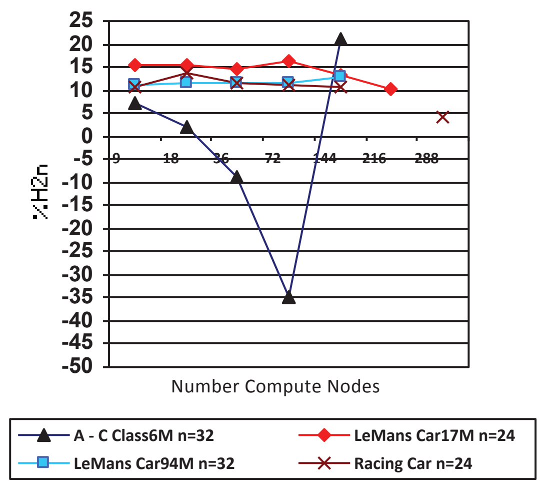


Figure 6d: Coupled Solver Dual Rail %H2n gains on ICE X for the four models



In fact, the Racing Car model gains appear to be slightly below those of the two LeMansCar models. This observation indicates that extremely large size models can incur heavier demands on the hyper-threading resources of the compute node. Thus, hyper-threading gains have a limit beyond which there are no further gains irrespective of how large the model size may be. Another observation from figures 6 is that Hyper-Threading gains appear to be higher in the case of the Coupled than the Segregated solver. Furthermore, comparing figures 6a and 6c with 6b and 6d shows the effect of Dual rail versus Single rail InfiniBand where all four models appear to show similar effects. Overall, the results show that the effect of Hyper-Threading Technology is best at lower node counts with the gains depreciating to negative values as node counts increase further. However, it is worth noting that in the case of Mercedes A-Class 6m Model model the upward spike in these values at 144 node counts possibly caused by the increased compute node bandwidth in the case of very large compute node counts (144 in this case) resulting in better caching effects and memory bandwidth. Figures 7a, 7b, 7c and 7d present plots of Hyper-Threading combined with Turbo boost gains, $\%C_{2n}$, values for the four input models using the Segregated solver Single rail, Segregated solver Dual rail, Coupled solver Single rail and Coupled solver Dual rail respectively. As noted in figures 6, figures 7 show similar observation; however, figures 7 also demonstrate the added effect of the Turbo boost feature.

Figure 7a: Segregated Solver Single Rail %C2n gains on ICE X for the four models

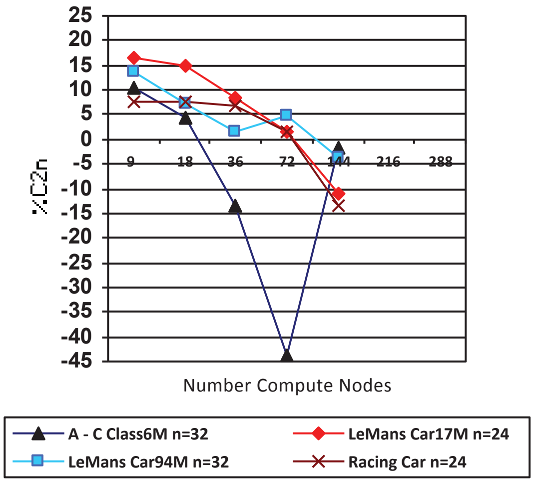


Figure 7b: Segregated Solver Dual Rail %C2n gains on ICE X for the four models

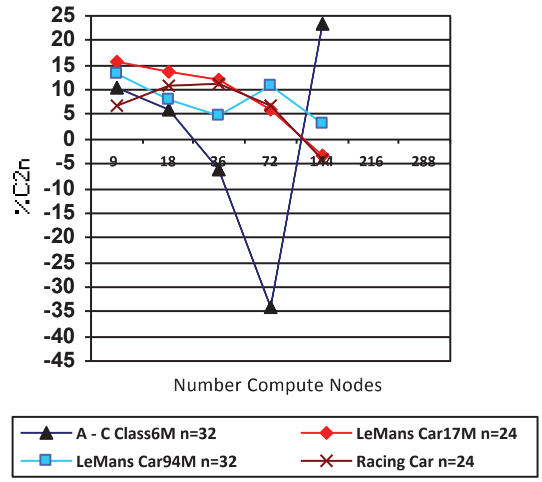


Figure 7c: Coupled Solver Single Rail %C2n gains on ICE X for the four models

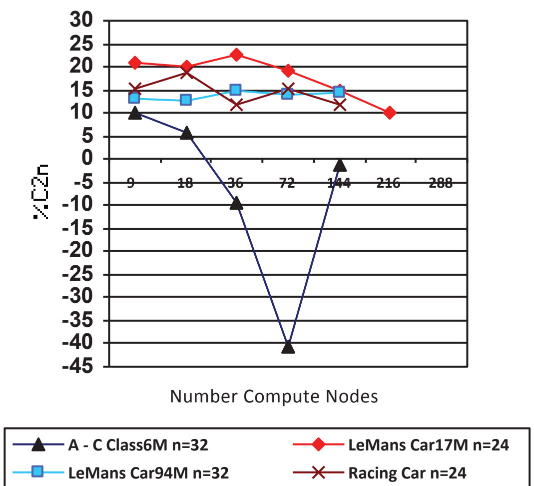
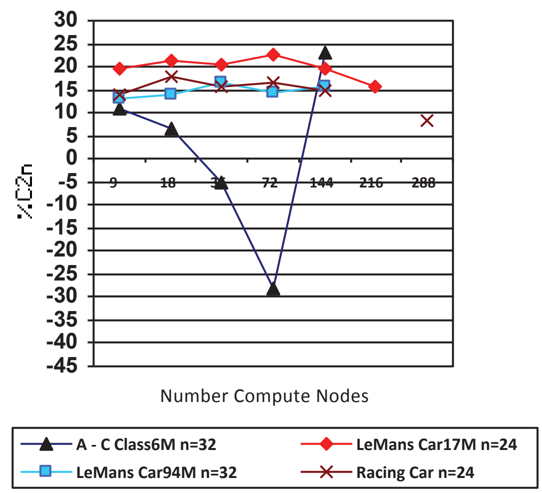


Figure 7d: Coupled Solver Dual Rail %C2n gains on ICE X for the four models



Figures 8, 9, 10 and 11 present in more detail the trends of the $\%H_{2n}$ and $\%C_{2n}$ gains for the four models respectively. Thus, figures 8a, 8b, 8c and 8d present the experiments data for the Mercedes A-Class 6m Model. Note that figures 8a and 8b are plots for Single versus Dual rail of $\%H_{2n}$ and $\%C_{2n}$ gains for the Segregated and Coupled solvers respectively, thus showing the amount benefit the Mercedes A-Class 6m Model may gain from Dual rail. While figures 8c and 8d are plots for the Segregated versus Coupled solvers corresponding $\%H_{2n}$ and $\%C_{2n}$ values with respect to Single and Dual rail respectively, thus showing how do the two solvers compare against each other in terms of Hyper-Threading and Turbo boost.

Figure 8a: AClass6M Segregated Solver %H2n and %C2n Single versus Dual rail, n=32

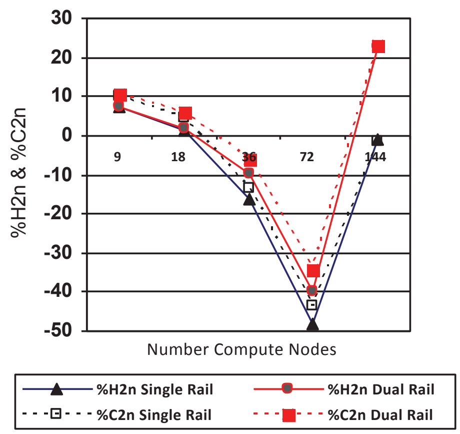


Figure 8b: AClass6M Coupled Solver %H2n and %C2n Single versus Dual rail, n=32

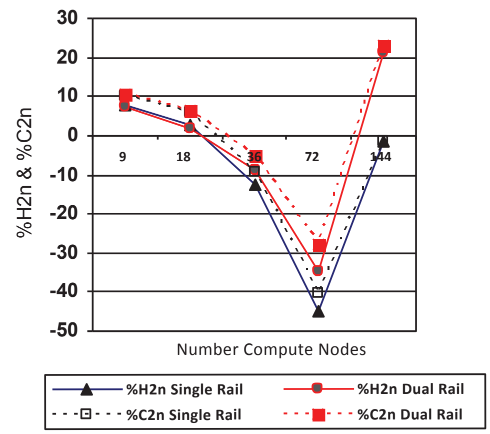


Figure 8c: AClass6M %H2n and %C2n Single rail Segregated versus Coupled Solver, n=32

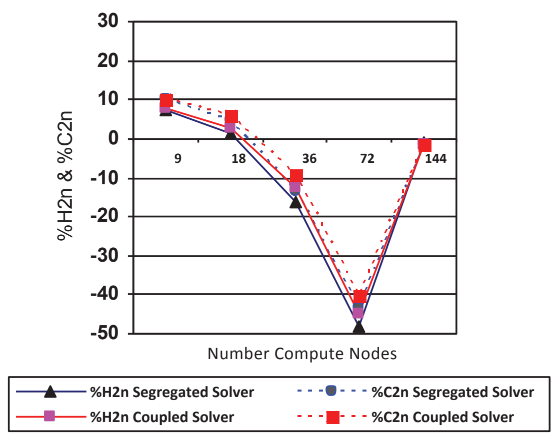
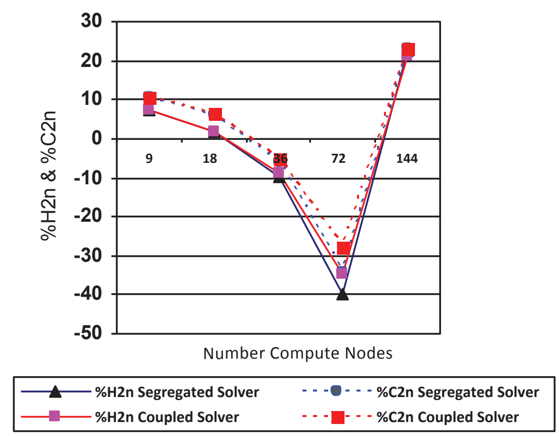


Figure 8d: AClass6M %H2n and %C2n Dual rail Segregated versus Coupled Solver, n=32



Figures 9a, 9b, 9c and 9d are the same as the corresponding figures 8 but for the LeMansCar 17M Model.

Figure 9a: LeMansCar17M Segregated Solver %H2n and %C2n Single versus Dual rail, n=24

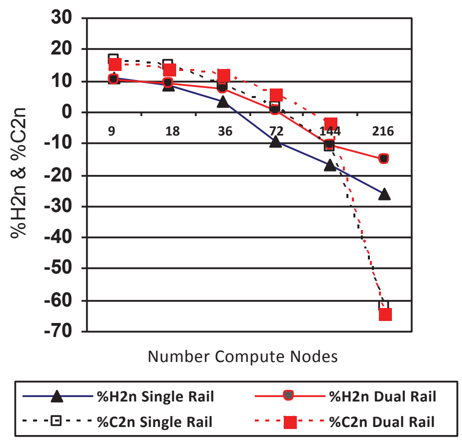


Figure 9b: LeMansCar17M Coupled Solver %H2n and %C2n Single versus Dual rail, n=24

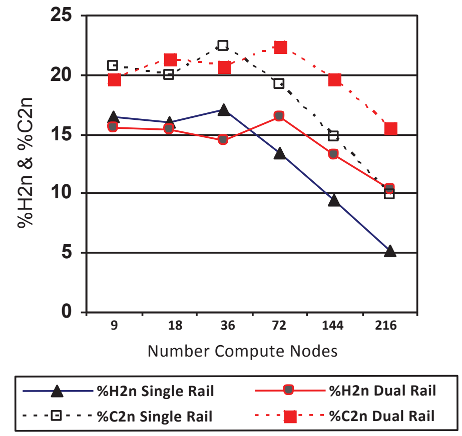


Figure 9c: LeMansCar17M %H2n and %C2n Single rail Segregated versus Coupled Solver, n=24

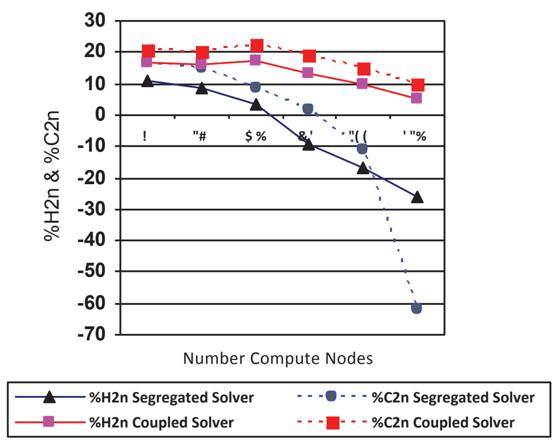
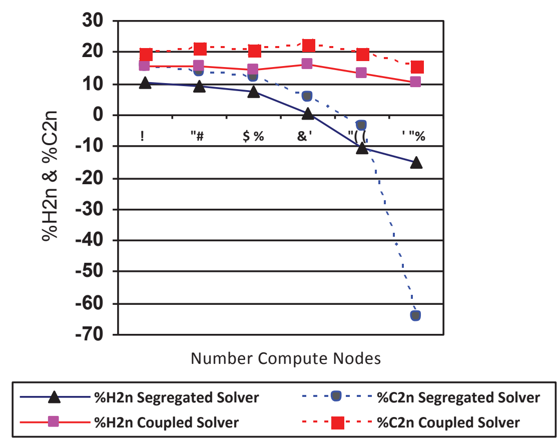


Figure 9d: LeMansCar17M %H2n and %C2n Dual rail Segregated versus Coupled Solver, n=24



Figures 10a, 10b, 10c and 10d are the same as the corresponding figures 8 but for the LeMansCar 94M Model.

Figure 10a: LeMansCar94M Segregated Solver %H2n and %C2n Single versus Dual rail, n=32

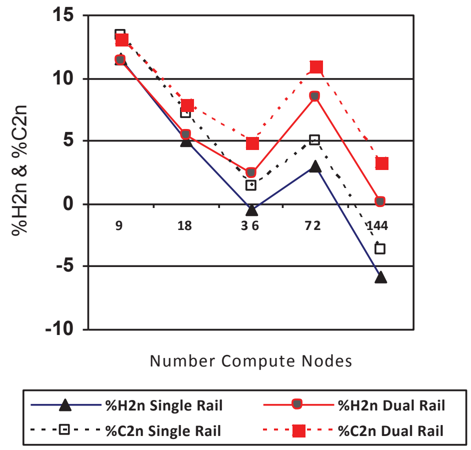


Figure 10b: LeMansCar94M Coupled Solver %H2n and %C2n Single versus Dual rail, n=32

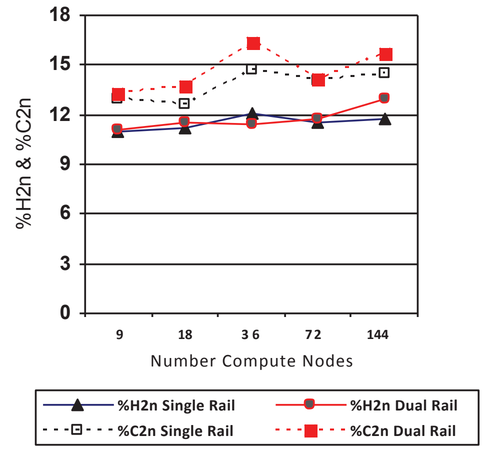


Figure 10c: LeMansCar94M %H2n and %C2n Single rail Segregated versus Coupled Solver, n=32

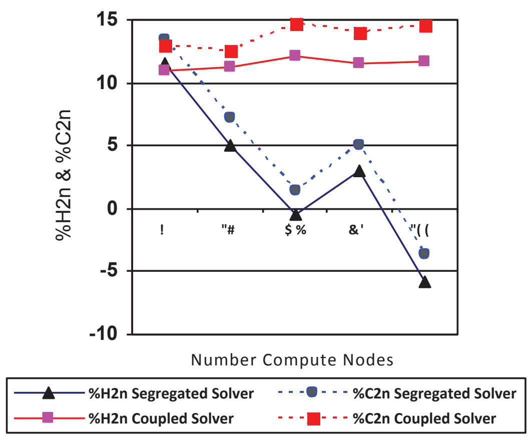
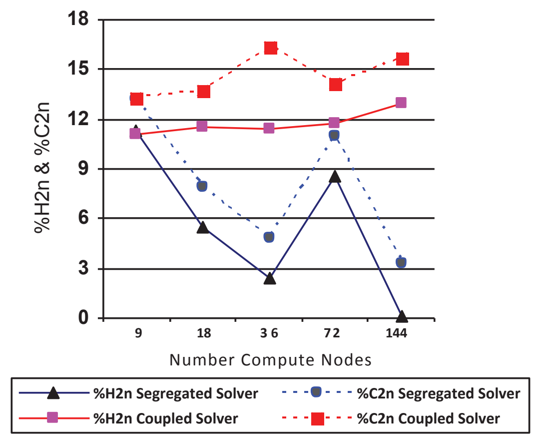


Figure 10d: LeMansCar94M %H2n and %C2n Dual rail Segregated versus Coupled Solver, n=32



Figures 11a, 11b, 11c and 11d are the same as the four corresponding figures 8 but for the Racing Car model.

Figure 11a: Racing Car Segregated Solver
%H2n and %C2n Single versus Dual rail, n=24

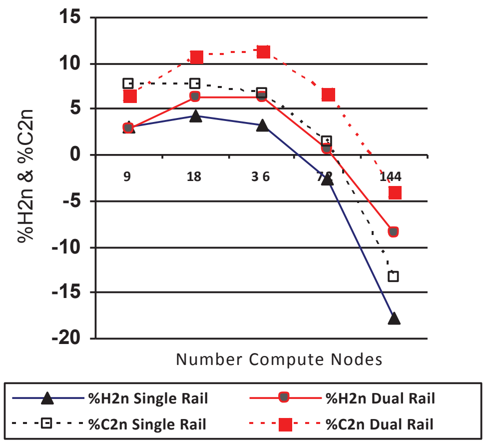


Figure 11b: Racing Car Coupled Solver
%H2n and %C2n Single versus Dual rail, n=24

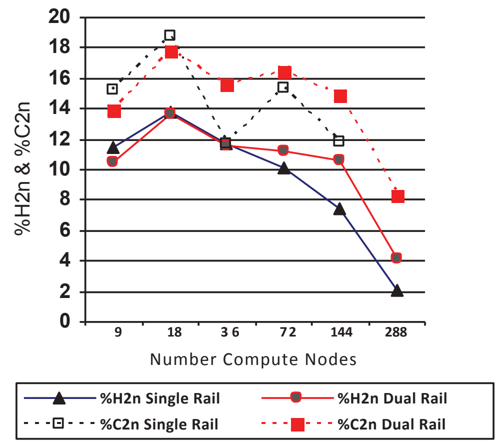


Figure 11c: Racing Car %H2n and %C2n Single rail
Segregated versus Coupled Solver, n=24

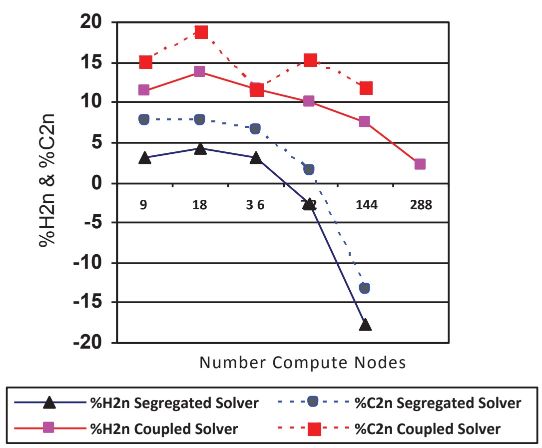
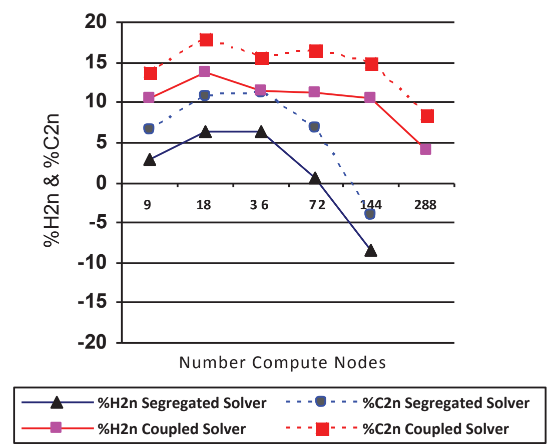
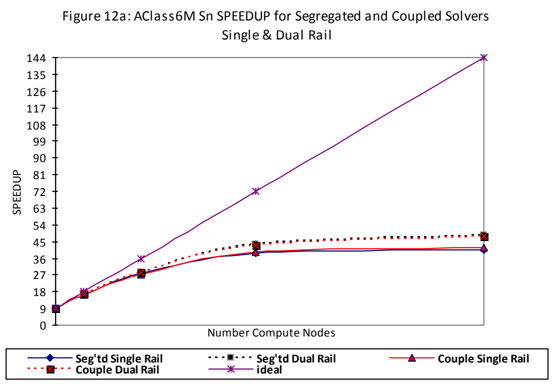


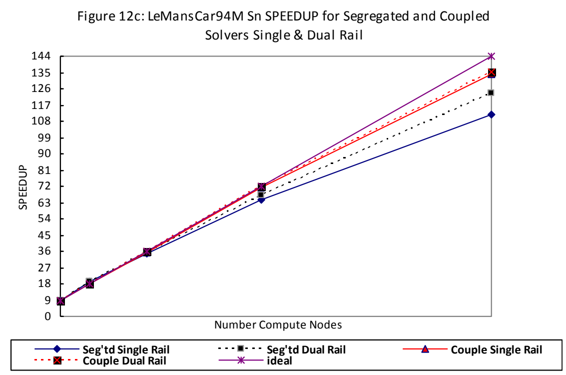
Figure 11d: Racing Car %H2n and %C2n Dual rail
Segregated versus Coupled Solver, n=24

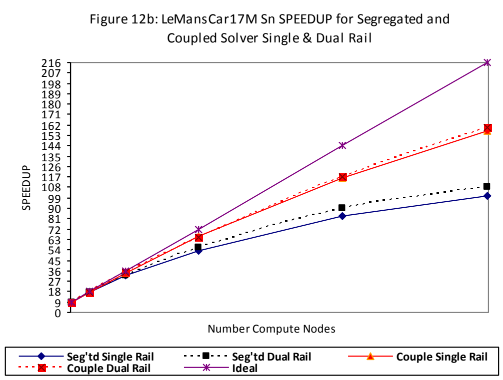


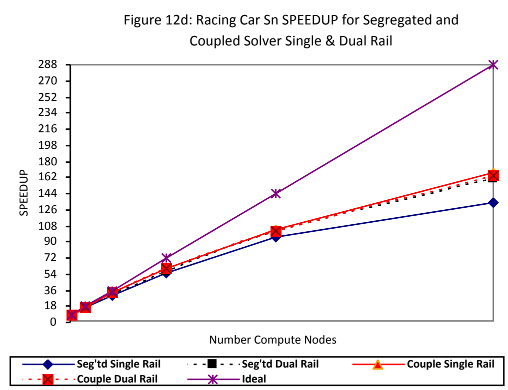
By considering each of the four models separately in figures 8, 9, 10 and 11, we can observe that the four models show similar patterns in the way the two solvers compare with each other and benefit from Hyper-Threading and Turbo boost as well as the way they compare with respect to Single versus Dual rail.

Figures 12a, 12b, 12c and 12d present plots of the S_n Parallel scalability factors measured with respect to the S_n value recorded for the tests using 9 compute nodes for each of the four models respectively.









The plots in figures 12 show:

- That parallel scalability is better for Coupled than Segregated Solver for these models
- Except for the Mercedes A-Class 6m Model, Dual rail is more effective for Segregated than Coupled solver, where the latter seem to show little benefit from Dual rail.

Examining the trends in tables of Section 5 and the figures in this section may help to provide a STAR-CCM+ user with guidance on what and how to make the best of the features of Hyper-Threading Technology and Turbo Boost in order to achieve best possible production levels. By numerically examining the actual values of S_n, H_{2n} and C_{2n} in the previous tables and figures, these metrics appear to reasonably satisfy the relationship given at the end of section 4 for which we will list here as an approximation.

Hence,

$C_{2n} \sim= H_{2n} + T_n - S_n$

Note that the reason that the above relationship is made as an approximation is because practical experiments normally are subject to noise in the execution environment thus, exact values for these metrics in practical experiments are unlikely to be observed.

A useful approach for gaining the maximum benefit of Hyper-Threading and Turbo Boost features may be achieved by implementing the following algorithm. Given an input model and based on a relatively small number of iterations/time-steps, execute the model using the following steps:

1. Start with an N number of compute nodes with n cores per node.

2. Run the model to obtain S_n and H_{2n}, values.
3. If $\Delta H_{2n} > 0$ then increase N and repeat step 2, where $\Delta H_{2n} = S_n - H_{2n}$
4. Else if $\Delta H_{2n} < 0$ then stop iterating and use the previous iteration values the N nodes and associated 'n' value as the resulting number of compute nodes that will provide the optimal value of H_{2n} and thus the corresponding C_{2n} value. Consequently, run the full simulation as a combined Hyper-Threading and Turbo Boost mode of operation using the resulting N as the optimal number of compute nodes.

That is the above algorithm enables users to approximately determine the optimal number of cores and nodes corresponding to the resulting best value of $\%C_{2n}$. Accordingly, users can run the full simulation using the resulting number of compute nodes/cores in a combined Hyper-Threading and Turbo Boost mode of operation for an optimal parallel performance.

The experiments of section 5.2 have shown that given a number of iterations, the Segregated solver out performs the Coupled solver at relatively lower core/node counts; however, at higher core/node counts both solvers have a reasonable parallel scalability scaling to 288 nodes depending on the model. Note that in this paper we ran the two solvers using equal number of iterations and accordingly recorded the final average elapsed time per iteration.

8.0 Summary

In this paper an analysis of the parallel performance of the commercial CFD application STAR-CCM+ from CD-adapco has been presented. The analysis has been based on experiments performed on the SGI ICE X compute cluster based on the Intel® Xeon® E5-2600v3. The analysis is primarily based on two features of this family of processors, namely the Intel® Hyper-Threading Technology and Turbo Boost. For these two features we defined the performance metrics 'S_n', 'H_{2n}','T_n' and 'C_{2n}' to correspond to the Solver's average elapsed time in the case of a Standard, Hyper-threaded, Turbo Boost and the Combined Hyper-threaded with Turbo Boost execution modes respectively. On the other hand, on the application side, four input models were considered, the Mercedes A-Class 6M, the LeManCar 17M, the LeManCar 94M and a Racing Car models. Experiments were performed using two STAR-CCM+ solvers, the Segregated flow solver and the Coupled flow solver. These experiments have shown that STAR-CCM+ can variably benefit from Hyper-Threading and Turbo Boost. Results have shown that Hyper-Threading gains appear to be larger in the case of the Coupled solver as opposed to the Segregated solver. Hyper-Threading feature percentage gains tends to drop gradually to a threshold value relative to an increase in the number of compute nodes beyond which gains drop significantly to negative values. This drop depends also on the number of cores per socket where larger number of cores per socket result in further acceleration in the drop of Hyper-Threading percentage gains. Overall, Hyper-Threading may be impaired by the following:

- Hyper-Threading will no longer gain performance if the executing model parallel scalability had reached its threshold for that number of domain decompositions.
- Hyper-Threading will not gain further performance if the data access requirement of the hyper-threaded threads exceeds the total bandwidth per compute node resulting in possible performance degradation.

Experiments have also shown that the application may benefit from the combined use of features of Hyper-Threading and Turbo Boost for a range of a number of compute nodes involved in the test. For the four input models, experiments have shown that the combined two features can provide significant performance gains for up to 18 compute nodes in the case of the segregated solver and up to 36 nodes in the case of the Coupled solver per test. Note that a combined gain may be considered as significant when the corresponding Hyper-Threading gain yields a positive performance gain for a test. More importantly, the two features combined gain can be quantified by the following approximation

$$C_{2n} \sim= H_{2n} + T_n - S_n$$

An algorithm to iteratively determine the optimal number of compute nodes and cores per node to achieve the optimal value of C_{2n} for a parallel experiment has been presented. Accordingly, STAR-CCM+ users can execute the full CFD simulation in production using the resulting number of compute nodes and thus achieving the best possible parallel performance. Finally, on the concept of InfiniBand Single versus Dual rail of the SGI ICE X architecture, results have shown that the Segregated solver appears to benefit more from Dual rail than the Coupled Solver.

9.0 References

[1] http://software.intel.com/en-us/articles/intel-xeon-processor-e5-2600-v2-product-family-technical-overview

[2] https://software.intel.com/en-us/articles/intel-xeon-processor-e5-2600-v3-product-family-technical-overview

[3] SGI ICE X System Hardware User Guide (document number: 007-5806-001, published 2012-03-28).

[4] SGI ICE X Installation and Configuration Guide (document number: 007-5917-002, published: 2013-11-20).

[5] "Performance Analysis of Industrial CFD software Using Parallel Measurements produced on Silicon Graphics x86-64 Compute Hardware". Ayad Jassim and Mark Kremenetsky. Silicon Graphics Int. Fremont, California USA. 2009. A VKI paper, Brussels, Belgium.

[6] White Paper: Intel Hyper-Threading. Hard Hat Area, February 2002, Vol 2, Issue 2.

[7] STAR-CCM+ standard installation PDF file: For example see latest STAR-CCM+ 10.02.010 release directory CCM+10.02.010/doc/UserGuide_10.02.pdf

10.0 About SGI

SGI is a global leader in high performance solutions for compute, data analytics and data management that enable customers to accelerate time to discovery, innovation, and profitability. Visit sgi.com for more information.

Global Sales and Support: sgi.com/global