CHALLENGE™ L and XL & ONYX™ Performance Report # 4

by SGI Performance Engineering

Silicon Graphics Inc.

Oct 15, 1993

SUMMARY: This report presents the results of several standard benchmarks on the 100 & 150 Mhz CHALLENGE L and XL servers and the ONYX graphics supercomputers

List of Contributors

We wish to thank the following people who have made contributions to this report:

Rick Bahr
David Baumgartner
Rosario Caltabiano
Mimi Celis
Wei-Chau Chang
David Ciemiewicz
Mike Galles
Jon Habets
Dick Hessel
Mike Humphrey
Cheng Liao
Mike Marcell
Viggy Mokkarala
Mat Myszewski
Jean-Pierre Panziera
Jack Perry
Jim Winget

Introduction

The fourth issue of the Performance Report focuses on benchmark suites for both uniprocessor and multiprocessor configurations for the 150 MHz CHALLENGE and Onyx computers running the latest version of the IRIX operating system - IRIX 5.1.1. This report is a superset of the Performance Report #3 that was published in July, 1993.

There are several benchmarks that are still missing from the list. Topics already on our list and presently in process are:

- AIM
- LADDIS

In addition, we are preparing a number of tutorials, with particular attention given to parallel processing and tuning codes for maximum performance.

If you have questions or comments, please address them to either Dick Hessel or Viggy Mokkarala in CSG Marketing.

Table of Contents

Benchmark Page

Section 1 Livermore FORTRAN Kernels

The Livermore Fortran Kernels (LFK) are a collection of 24 FORTRAN code fragments extracted from application programs at Lawrence Livermore National Laboratory (LLNL). The benchmark was developed by Frank McMahon in the early 1970's when LLNL acquired the CDC 7600's. The initial purpose was to measure the efficiency of CPU code generated by their LRLTRAN (LLNL FORTRAN) compiler and to compare results with the CDC FORTRAN compilers, RUN(66) and FTN(77). In the late 1970's all the CDC software was ported to the Cray1, and the original 12 loops were augmented to 24 in 1985. McMahon first published vendor results in 1986.

The LFK produces a MFLOP rating for each of the 24 kernels, at each of three vector lengths, for a total of 72 MFLOP ratings. Both double and single precision results are produced.

The three average vector lengths are called short, medium and long (19, 90, and 470). A fourth set of results are provided for the overall average length of 167. These are composite statistics drawn from the other three sets of samples. Valid comparisons can be made only between results for the same vector length. In the test results, the average vector length is identified as DO Span.

The LFK produces a set of summary statistics, three of which (Harmonic mean, Geometric Mean, and Arithmetic Mean) are included in this report. These composite MFLOP ratings provide information on the sensitivity of the test machine's performance to the percentage of vectorization of the code being run.

The Harmonic Mean rating roughly corresponds to performance on code that is 40% vectorizable. The Geometric Mean roughly corresponds to performance on code that is 70% vectorizable, and the arithmetic mean roughly corresponds to performance on code that is 90% vectorizable.

Eleven of the 24 kernels can be effectively parallelized. Fifteen can be effectively vectorized. This mix makes LFK results sensitive to the effects of both parallelization and vectorization.

LFK provides an excellent characterization of a machine's FORTRAN floating point performance. The kernels are drawn from production programs at LLNL, chiefly hydrodynamics applications. Some of the kernels are synthetic, e.g.. #7 - a sequence of multiply/adds (or "linked triads" in vector parlance), and give an accurate measure of specific features in vector architecture and accompanying software.

Performance enchancements from parallelization and vectorization make the kernels an interesting and effective tool to compare RISC, cache-based machines with vector machines. One of LFK's strengths is the realistic diversity of programming styles, which can have a strong effect on the benefits of parallelization and vectorization.

100 Mhz CHALLENGE L (1 cpu) RESULTS - DP - 167 DO SPAN

Memory size: 256 MB

Interleaving: 2-way interleaving

Compiler Flags: f77 -O2 -mips2 -sopt -r=3

Date of run: March 16, 1993

MFLOPS RANGE:	REPORT ALL RANGE STATISTICS:		
Maximum Rate	=	35.1132	Mega-Flops
Average Rate	=	15.7940	Mega-Flops
Geometric Mean	**=**	**14.1369**	**Mega-Flops**
Harmonic Mean	=	12.6808	Mega-Flops
Minimum Rate	=	6.3119	Mega-Flops
Standard Dev.	=	7.479	Mega-Flops

150 Mhz CHALLENGE L (1 cpu) RESULTS - DP - 167 DO SPAN

Memory size: 256 MB

Interleaving: 2-way interleaving

Compiler Flags: f77 -O3 -mips2 -sopt,-r=3,-chs=16,-ur=8,-ur2=770000,-so=3
-K -Wo,-loopunroll,8

IRIX: IRIX 5.0.1

Date of run: July 21, 1993

43.9560

MFLOPS RANGE:	REPORT ALL RANGE STATISTICS:		
Maximum Rate	=	43.9560	Mega-Flops
Average Rate	=	21.7302	Mega-Flops
Geometric Mean	**=**	**19.7797**	**Mega-Flops**
Harmonic Mean	=	17.9027	Mega-Flops
Minimum Rate	=	8.9600	Mega-Flops
Standard Dev.	=	9.1550	Mega-Flops

Section 2 NAS Kernels Benchmark

The “NAS Kernel Benchmark Program” contains seven tests assembled in 1984 by NASA-Ames to represent their CFD computational requirements. The tests are Fortran subroutines dominated by 64-bit floating point arithmetic, and contain nested loops operating on multidimensional arrays.

Each of the tests performs an error check and a MFLOPS calculation based on execution time and operation count. A total MFLOPS calculation is also computed, based on the aggregate time and operation count of all seven tests.

These NAS Kernels are not to be confused with the “NAS Parallel Kernels”, a different set of tests released more recently for testing massively parallel systems.

Although developed on Crays and suitable for vector and parallel compiler optimizations, the kernels contain several undesirable characteristics for supercomputer architectures, such as non-stride-1 inner loops, power-of-2 array dimensions and strides, and loops with small iteration counts. This makes them an interesting set of tests for investigating the performance characteristics of processors, cache/memory systems, parallel architectures, and optimizing compilers.

The names and descriptions of the kernels are:

- MXM Matrix Multiply
- CFFT2D Complex 2D FFT
- CHOLSKY Cholesky Decomposition/Solution of Banded Systems
- BTRIX Block Tridiagonal Solver
- GMTRY Generate Solid-Related Matrix, Gaussian Eliminate
- EMIT Emit Vortices, Pressure, Forces
- VPENTA Vectorized Inversion of 3 Pentadiagonals

The “original version” results shown in Table 1 were achieved using automatic parallelization optimizations performed by the Power Fortran Accelerator (PFA) through the -pfa compiler switch alone.

The “tuned version” consists entirely of compiled Fortran, with no use of hand-coding or optimized math libraries. In some cases, DOACROSS directives were inserted to bypass PFA and parallelize specific loops, and in other cases code was reorganized so that PFA would perform a more effective optimization.

For the 150 MHz benchmark runs, some of the apparent anomalies have good explanations. For example, BTRIX levels off at 16 processors but takes a jump at 32 because the parallel loop has 30 iterations. GMTRY has superlinear speedup because the matrix is 2 MB and so there are more cache misses on one processor than when the matrix is split up among processors.

Tables of SGI 100 Mhz CHALLENGE Results for NAS Kernels

The following tables contain the results achieved on an SGI CHALLENGE System with 100 MHz R4400 CPUs. These results were obtained with IRIX 5.0.

Memory size: 1024 MB

Interleaving: 8-way interleaving

Compiler Flags: Untuned version = -align64 -r8 -o2 -mips2 -non_shared -pfa

Tuned versions = different flags for different subroutines

Date of run: March 27, 1993

Table 1 Original Version, using Automatic PFA Parallelization

	MFLOPS Rates						Parallel Speedups				
	Number of Processors:										
	1	4	8	12	16	20	4	8	12	16	20
MXM	18.5	69.3	140.8	187.3	263.8	275.9	3.7	7.6	10.1	14.3	14.9
CFFT2D	11.0	18.7	20.6	19.2	16.9	15.4	1.7	1.9	1.7	1.5	1.4
CHOLSKY	3.9	5.4	5.7	5.6	5.5	5.4	1.4	1.5	1.5	1.4	1.4
BTRIX	11.1	13.1	15.9	16.8	17.6	16.7	1.2	1.4	1.5	1.6	1.5
GMTRY	8.1	31.8	62.7	89.2	110.0	131.7	3.9	7.7	11.0	13.5	16.2
EMIT	15.9	55.5	85.6	111.9	122.2	127.7	3.5	5.4	7.0	7.7	8.0
VPENTA	3.9	13.4	23.9	29.1	32.2	32.1	3.5	6.1	7.5	8.3	8.2
TOTAL	8.3	16.9	21.0	21.7	21.4	20.6	2.0	2.5	2.6	2.6	2.5

Table 2 Tuned Version, using PFA Parallelization plus Directives and Source Changes

	MFLOPS Rates						Parallel Speedups				
	Number of Processors:										
	1	4	8	12	16	20	4	8	12	16	20
MXM	31.1	118.2	224.3	317.8	403.3	476.6	3.8	7.2	10.2	13.0	15.3
CFFT2D	8.3	32.6	63.6	89.7	115.8	65.5	3.9	7.7	10.8	14.0	7.9
CHOLSKY	5.4	19.8	40.7	61.4	76.0	94.9	3.7	7.5	11.4	14.1	7.6
BTRIX	6.6	27.1	51.1	68.1	98.8	100.0	4.1	7.7	10.3	15.0	15.2
GMTRY	8.1	31.3	61.6	88.1	110.5	132.5	3.9	7.6	10.9	13.6	16.4
EMIT	15.0	53.2	89.0	109.7	122.9	130.7	3.5	5.9	7.3	8.2	8.7
VPENTA	10.2	33.5	63.0	89.8	116.3	135.8	3.3	6.2	8.8	11.4	13.3
TOTAL	9.4	35.6	68.4	95.6	123.2	112.0	3.8	7.3	10.2	13.1	11.9

The following tables contain the results achieved on an SGI CHALLENGE System with 150 MHz R4400 CPUs.

Memory size: 768 MB

Interleaving: 2-way interleaving

Compiler Flags: Untuned version = -align64 -r8 -o2 -mips2 -non_shared -pfa

Tuned versions = different flags for different subroutines

Date of run: Oct 13, 1993

Table 3 Original Version, using Automatic PFA Parallelization

MFLOPS Rates

		Number of Processors:					
	1	4	8	12	16	24	32
MXM	26	102	197	263	385	496	671
Speedup		4.0	7.7	10	15	19	26
CFFT2D	17	34	36	35	32	29	23
Speedup		1.9	2.1	2.0	1.9	1.6	1.3
CHOLSKY	6	8	8	8	8	8	8
Speedup		1.3	1.4	1.4	1.4	1.4	1.4
BTRIX	15	18	24	25	27	25	24
Speedup		1.2	1.5	1.6	1.7	1.6	1.5
GMTRY	12	54	100	140	172	228	259
Speedup		4.5	8.4	12	14	19	22
EMIT	23	84	145	184	210	230	239
Speedup		3.7	6.3	8.0	9.2	10	10.4
VPENTA	5	17	31	36	37	38	41
Speedup		3.6	6.5	7.4	7.7	8.0	8.6
TOTAL (MFLOPS)	12	25	32	33	33	32	31
TOTAL (Speedup)		2.1	2.7	2.9	2.9	2.8	2.7

Table 4 Tuned Version, using PFA Parallelization plus DIrectives and Source Changes

MFLOPS Rates

	Number of Processors:						
	1	4	8	12	16	24	32
MXM	42	163	301	447	560	748	895
Speedup		3.9	7.1	11	13	18	21
CFFT2D	13	49	90	126	155	159	176
Speedup		3.7	6.7	9.4	12	12	13
CHOLSKY	10	39	90	126	155	159	176
Speedup		4.1	7.6	12	15	19	25
BTRIX	11	40	77	105	150	148	267
Speedup		3.7	7.2	9.8	14	14	25
GMTRY	12	53	99	138	170	223	262
Speedup		4.5	8.4	12	15	19	22
EMIT	23	83	144	179	213	234	237
Speedup		3.7	6.4	7.9	9.4	10	11
VPENTA	14	48	91	129	164	192	211
Speedup		3.4	6.5	9.2	12	14	15
TOTAL (MFLOPS)	15	56	104	147	187	209	256
TOTAL (Speedup)		3.8	7.1	10	13	14	17

Section 3 Linpack Benchmark

The Linpack Benchmark is widely accepted in both the computer industry and the user community as the initial measure of floating point performance of a compute server. While it is accepted that other application derived benchmarks and end user specific benchmarks more accurately depict the expected performance in daily use, Linpack mflops are still the most common performance measure. Procurements for large system or numbers of systems will invariably have a minimum mflops criterion.

Jack Dongarra of Argonne National Laboratories developed this floating point performance benchmark [Dongarra, 91]. The code is 736 lines of Fortran that compile into 8KB of instructions and 315 KB of static data. Linpack benchmark results are reported in MFLOPS for the matrix sizes 100 and 1000 in 32-bit and 64-bit floating point precision.

The Linpack 100x100 measures the Fortran compiler as much as the machine architecture, since the "run rules" state that the code must be run without human intervention. The small size of the matrix makes it difficult for machines with fast CPUs, large memory and high throughput capacity to approach a sizable fraction of their "sustainable" mflop rate.

For the 1000x1000 matrix size test, the vendor is allowed to completely rewrite the solver to optimize performance on their architecture. Only results for the same matrix size can be meaningfully compared. The Double Precision, 100 x 100 result is the most commonly quoted Linpack result. If a quoted Linpack result is not fully identified, it is assumed to be this one.

For the 100x100 matrix size benchmark, only the Silicon Graphics Power Fortran Acceleration (PFA) and the compiler are used to inline, unroll and parallelize the code. For the 1000x1000 matrix size benchmark, the solver was rewritten to perform calls to the optimized BLAS library bundled with IRIX. This library is optimized for the register, and cache architecture of the SGI line of products and provides coarse grain parallel execution when multiple processors are available.

Memory size: 256 MB

Interleaving: 2-way interleaving

Compiler Flags: -mips2 -o -non_shared -noisam -sopt,-r=3 (for 1000x1000)

different flags for different runs of the 100x100 benchmark

Date of run: Apr 12, 1993

Table 5 Linpack Benchmark results for 100 Mhz CHALLENGE L

	Number of Processors							
	1	2	4	6	8	12	16	20
1000 x 1000								
linpackd (tuned)								
MFLOPS	34.2	64.4	119.6	165.9	204.5	263.3	299.9	331
100 x 100								
linpackd								
MFLOPS	17.5	24.9	39.0	45.65	48.7			

Memory size: 512 MB

Interleaving: 8-way interleaving

1000x1000 parallel:

IRIX: 5.1

Compiler Flags: -mips2 -o -non_shared -noisam -sopt,-r=3

100x100 parallel:

IRIX: 5.1.1

-pfa -keep -WK,-ipa=daxpy:saxpy,-cachesize=16,-roundoff=3,
-unroll=1,-minconcurrent=1000 -O2 -mips2 -Wo,-loopunroll,16 -Olimit 2000
-Wf,-dcacheopt -jmpopt -non_shared

Date of run: Oct 13, 1993

Table 6 Linpack Benchmark results for 150 Mhz CHALLENGE L

	Number of Processors							
	1	2	4	8	12	16	24	32
1000 x 1000 linpackd (tuned) MFLOPS	48.4	93.5	178.3	311	405.3	467.6	552.6	608
100 x 100 linpackd MFLOPS	26.7	37.8	59.1	76.1				

Section 4 Homogeneous Capacity (SPECrate_fp92, SPECrate_int92)

The CINT92 and CFP92 Homogeneous Capacity (HC) suites succeed the Spec89 Thruput A suite, which used the Spec89 suite to measure throughput. The two HC suites measure the processing capacity of a machine by measuring the number of INT92 and FP92 tests, respectively, that complete within a specified time period. This is not a system-level test: SPEC SDM is intended to serve as a system test. The HC suites are aimed at assessing the raw CPU throughput horsepower of a machine.

The HC method uses precisely the same benchmarks in the INT92 and FP92 suites. The executables are identical, but the tools for execution and evaluation of results have changed.

There are two metrics, SPECrate_int92 and SPECrate_fp92, for HC of INT92 and FP92, respectively. The definition for each is:

SPECrate_(x) = #CopiesRun * ReferenceFactor* UnitTime / ElapsedExecutionTime

- **#CopiesRun** - Number of copies for each test, e.g., Users are free to choose an "optimal" number for each test in INT92 & FP92.
- **Reference** - The duration of the longest test which is 25500 seconds. This is intended to be a normalization factor.
- **UnitTime** - One week in seconds = 604800. This is intended to normalize SPECrate throughput by one weeks worth of throughput on a VAX 11-780.

Memory size: 256 MB

Interleaving: 2-way interleaving

Compiler Flags: different for each program

Date of run: Apr 5, 1993 (5.0)

July 20, 1993 (5.0.1)

Table 7 INT92 Homogeneous Capacity (SPECrate_int92). Results are for 100 Mhz CHALLENGE running IRIX 5.0 and 5.0.1.

# CPUs	1	4	8	12
SPECrate_int92 (5.0)	1478	5562	10177	13406
SPECrate_int92 (5.0.1)	1414	5524	10272	15506

Table 8 FP92 Homogeneous Capacity (SPECrate_fp92). Results are for 100 MHz CHALLENGE running IRIX5.0 and 5.0.1.

# CPUs	1	4	8	12	16	20
SPECrate_fp92 (5.0)	1508	6131	12020	17370	22291	27459
SPECrate_fp92 (5.0.1)	1580	6058	11893	17628	23062	26359

Memory size: 768 MB

Interleaving: 2-way interleaving

Compiler Flags: different for each program

Date of run: Oct 13, 1993

{PUT IN SALES STUFF HERE]

Table 9 Int92 Homogeneous Capacity (SPECrate_int92). Results are for 150 MHz CHALLENGE running IRIX5.1.1.

# CPUs	1	2	4	8	12	16	20
SPECrate_int92	2221	4408	8679	16849	23696	27242	31073
% drop	0%	.76%	2.3%	5.1%	11.1%	23.4%	30%

Table 10 FP92 Homogeneous Capacity (SPECrate_fp92). Results are for 150 MHz CHALLENGE running IRIX5.01.

# CPUs	1	2	4	8	12	16	20
SPECrate_fp92	2306	4550	9079	17584	25171	33956	40013
% drop	0%	1.3%	1.6%	4.7%	9.1%	8%	13.3%

#CPUs	24	28	32
SPECrate_fp92	45776	53796	56840
% drop	17.3%	16.7%	23%

Section 5 SPEC 1.0 Benchmark Suite Release 1.0

SPEC (Systems Performance Evaluation Cooperative) began in 1988 and produced Release 1.0 of the SPEC Benchmark Suite in early 1989. The charter of SPEC is to develop, maintain and endorse a set of application based benchmarks which can effectively measure performance of the newest generation of high performance computers. The overall goal is to provide accurate and comprehensive performance measurement tools, sufficient to make meaningful comparisons among machines. One of the more specific goals was to provide a means of measurement and comparison which would simplify and transcend prior mips comparisons where the basic notion of "mips" was itself nonstandard.

Release 1.0 of SPEC consists of 10 benchmarks drawn from real-world applications, including CASE, MCAE, CFD, Quantum Chemistry, and others. Also included are the GNU C compiler and a LISP interpeter. All tests All tests are CPU intensive. Four of these ten tests are essentially integer, and the remaining six are essentially floating point. The ratio of integer/floating point instructions, by count, varies widely over the set of ten tests.

The ratio, (elapsed time) / (VAX 11-780 elapsed time) is designated as the SPECmark on a given machine for each of the 10 tests. The overall SPECmark is the Geometric Mean of the ten individual Specmarks. SPEC joins the rest of the benchmarking community in attempting to reduce performance to a single number, for easy comparisons.

More detailed comparisons can be made by separating the tests into the integer and floating point subsets, and this idea formed the basis for the eventual development of the SPEC92 suites, FP92 and INT92.

Memory size: 256 MB

Interleaving: 2-way interleaving

Compiler Flags: different for each program

Date of run: Apr 5, 1993

Table 11 SPEC*89 Results for 100 Mhz CHALLENGE L

Test	VAX Reference	SGI	SPEC
001.gcc1.35	1481.5	33.10	44.8
008.espresso	2266.0	43.49	52.1
013.spice2g6	23951.4	524.86	45.6
015.doduc	1863.0	33.13	56.2
020.nasa7	20093.1	13.95	93.9
022.li	6206.2	85.17	72.9
023.eqntott	1100.8	14.17	77.7
030.matrix300	4525.1	16.78	270.0
042.fpppp	3038.4	50.65	60.0
047.tomcatv	2648.6	34.86	76.0
SPECmark			**72.3**
SPECint			**60.3**
SPECfp			**81.6**

Section 6 SPECin92 and SPECfp92 Benchmarks.

SPEC release 2.0 appeared in Jan'92. SPEC92 consists of two distinct suites, FP92 and INT92. These suites augment the original SPEC89 (Release 1.0) and separate the 6 integer and 14 floating point tests into two suites.

The basic philosophy remains, and the new tests measure real-world applications not represented in SPEC89. INT92 added text compression and graphics (with a spreadsheet emulation) for a total of 6 tests. FP92 includes robotics, weather, Fluid Dynamics, and optics, for a total of 14 tests. Additionally, the problem sizes were increased for several of the test retained from SPEC89. All test except MATRIX300 were retained from SPEC89.

The highly vectorizable tests can be used to compare the newer cache-based risc architectures with the older vector machines.

Memory size: 256 MB (100 MHz run); 768 MB (150 MHzrun)

Interleaving: 2-way interleaving for both 100 & 150 MHz

Compiler Flags: different for each program

Date of run: Apr 5, 1993 (100 MHz run); Oct 13, 1993 (150 MHz run)

Table 12 SPEC92 FP Individual Test Data for 150 MHz CHALLENGE L/XL

TEST	TIME	SPEC
013.spice2g6	365.2	65.7
015.doduc	21.64	86
034.mdljdp2	52.8	134
039.wave5	48.46	76.4
047.tomcatv*	25.18	105
048.ora	66.56	111
052.alvinn	67.33	114
056.ear	116.21	219
077.mdljsp2	50.94	65.8
078.swm256	254.45	49.9
089.su2cor	115.99	111
090.hydro2d	117.37	117
093.nasa7	153.66	109
094.fpppp	110.95	82.9

SPECfp92		97.1

Table 13 SPEC92 FP Individual Test Data for 100 Mhz CHALLENGE L.

TEST	TIME	SPEC
013.spice2g6	516.6	46.5
015.doduc	34.5	53.9
034.mdljdp2	78.9	89.9
039.wave5	70.6	52.4
047.tomcatv*	34.65	76.5
048.ora	100.2	74.1

052.alvinn	100.7	76.4
056.ear	267.3	140.0
077.mdljsp2	75.6	44.3
078.swm256	357.4	35.5
089.su2cor	155.75	82.8
090.hydro2d	163.4	83.8
093.nasa7	223.7	75.1
094.fpppp	64.6	55.9

SPECfp92		66.5

Table 14 SPEC92 INT Individual Test Data for 150 Mhz CHALLENGE L.

TEST	TIME	SPEC
008.espresso	27.52	82.5
022.li	58.02	107
023.eqntott	9.33	118
026.compress	31.82	87.1
072.sc	47.04	96.3
085.gcc	70.59	77.3

SPECint92		93.7

Table 15 SPEC92 INT Individual Test Data for 100 Mhz CHALLENGE L.

TEST	TIME	SPEC
008.espresso	42.2	54.3
022.li	84.6	73.4
023.eqntott	14.0	79.7
026.compress	50.3	49.0
072.sc	78.9	68.2
085.gcc	103.4	55.3

SPECint92		62.4

Section 7 FFT Timings on the 100 Mhz CHALLENGE

The FFT timings in these tables were computed with SGI's LIBFFT version 2.3. The binary version of this library can be obtained on SGI's FTP site "sgi.com" accessable through the usual anonymous login protocol.

A compressed tar file will be found in directory sgi/libfft.

C-C : complex ---> complex Single Precision

Z-Z : complex ---> complex Double Precision

R-C : real ---> complex Single Precision

D-Z : real ---> complex Double Precision

The 150 MHz measurements were made under the following conditions:

Memory size:	768 MB, 2-way interleaved
IRIX:	IRIX 5.1.1
Date of run:	Oct 14, 1993

Table 16 1D FFTs for 100 MHz systems - Timings in Seconds. Mflops are shown as (NN). The Mflops are computed according to a count of 5.N.Log(N) for an FFT of size N (complex-complex).

SIZE	C_C	Z-Z	R-C	D-Z
64	7.4e-5 (26)	7.1e-5 (27)	4.9e-5 (16)	5.2e-5 (14)
256	2.9e-4 (35)	2.8e-4 (37)	1.6e-4 (28)	1.7e-4 (23)
1024	1.8e-3 (28)	2.6e-3 (20)	7.2e-4 (32)	1.3e-3 (18)
4096	1.1e-2 (22)	1.4e-2 (18)	4.5e-3 (25)	6.7e-3 (17)
16384	5.7e-2 (20)	7.6e-2 (15)	3.1e-2 (17)	3.5e-2 (15)
65536	3.2e-1 (16)	6.3e-1 (8)	1.4e-1 (17)	2.5e-1 (10)

Table 17 1D FFTs for 150 MHz systems - Timings in Seconds. Mflops are computed according to a count of 5.N.Log(N) for an FFT of size N (complex-complex).

SIZE	C_C	Z-Z	R-C	D-Z
64	4.8e-5	4.3e-5	3.3e-5	3.2e-5
256	2.0e-4	1.8e-4	1.1e-4	1.2e-4
1024	1.2e-3	1.7e-3	4.7e-4	8.2e-4
4096	7.4e-3	9.2e-3	3.0e-3	4.5e-3
16384	3.8e-2	5.2e-2	2.0e-2	2.4e-2
65536	2.3e-1	4.8e-1	9.5e-2	1.8e-1

Table 18 2D FFTs for 100 MHz systems - Complex-Complex Single Precision (C-C). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
64	9.7e-3 (25)	7.9e-3 (x1.2)	4.4e-3 (x2.2)	3.4e-3 (x2.8)
256	1.8e-1 (29)	1.3e-1 (x1.4)	7.2e-2 (x2.5)	4.3e-2 (x4.2)
1024	4.7e+0 (22)	2.6e+0 (x1.8)	1.3e+0 (x3.6)	7.2e-1 (x6.5)
2048	2.2e+1 (20)	1.2e+1 (x1.8)	5.9e+0 (x3.7)	3.1e+0 (x7.1)

Table 19 2D FFTs for 150 MHz systems - Complex-Complex Single Precision (C-C). Timings in Seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
64	6.4e-3	6.3e-3	3.5e-3	2.3e-3	2.0e-3	2.3e-3
256	1.1e-1	8.4e-2	4.8e-2	2.8e-2	1.7e-2	2.8e-3
1024	3.2e+0	1.7e+0	9.3e-1	5.1e-1	3.2e-1	2.4e-1
2048	1.5e+1	8.1e+0	4.2e+0	2.2e+0	1.3e+0	1.1e+0

Table 20 2D FFTs for 100 MHz systems - Complex-Complex Double Precision (Z-Z). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
64	1.1e-2 (22)	1.0e-2 (x1.1)	5.4e-3 (x2.0)	5.1e-3 (x2.2)
256	2.3e-1 (23)	1.8e-1 (x1.3)	1.0e-1 (x2.3)	6.5e-2 (x3.5)
1024	7.3e+0 (15)	3.8e+0 (x1.9)	2.0e+0 (x3.6)	1.2e+0 (x6.1)
2048	3.6e+1 (13)	1.8e+1 (x2.0)	9.6e+0 (x3.7)	5.1e+0 (x7.1)

Table 21 2D FFTs for 150 MHz systems - Complex-Complex Double Precision (Z-Z). Timings in seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
64	7.3e-3	7.3e-3	4.0e-3	3.5e-3	3.2e-3	3.3e-3
256	1.5e-1	1.2e-1	7.8e-2	4.3e-2	2.6e-2	3.6e-2
1024	5.0e+0	2.7e+0	1.5e+0	7.8e-1	4.7e-1	3.9e-1
2048	2.5e+1	1.3e+1	6.7e+0	3.7e+0	2.3e+0	1.8e+0

Table 22 2D FFTs for 100 MHz systems - Real-Complex Single Precision (R-C). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
64	5.5e-3 (20)	5.1e-3 (x1.1)	3.7e-3 (x1.5)	3.0e-3 (x1.8)
256	9.9e-2 (25)	8.4e-1 (x1.2)	9.1e-2 (x1.1)	1.2e-2 (x.82)
1024	2.6e+0 (19)	1.5e+0 (x1.7)	1.1e+0 (x2.4)	1.2e-1 (x2.2)
2048	1.2e+1 (19)	6.6e+1 x1.8)	3.8e+0 (x3.2)	2.8e+0 (x4.3)

Table 23 2D FFTs for 150 MHz systems - Real-Complex Single Precision (R-C). Timings in seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
64	3.6e-3	2.9e-3	2.3e-3	2.0e-3	1.5e-3	2.1e-3
256	7.0e-2	5.4e-2	5.3e-2	8.0e-2	1.2e-1	1.2e-1
1024	1.8e+0	1.0e+0	7.2e-1	7.2e-1	1.0e+0	1.7e+0
2048	7.9e+0	4.3e+0	2.5e+0	1.7e+0	1.8e+0	2.5e+0

Table 24 2D FFTs for 100 MHz systems - Real-Complex Double Precision (R-Z). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
64	7.1e-3 (16)	5.9e-3 (x1.2)	4.1e-3 (x1.7)	3.9e-3 (x1.8)
256	1.4e-1 (18)	1.3e-1 (x1.1)	1.6e-2 (x.87)	2.2e-1 (x.63)
1024	3.8e+0 (13)	2.4e+0 (x1.6)	1.5e+0 (x2.5)	1.4e+0 (x2.7)
2048	1.7e+1 (13)	9.8e+0 (x1.7)	5.6e+0 (x3.0)	3.3e+0 (x5.2)

Table 25 2D FFTs for 150 MHz systems - Real-Complex Double Precision (R-Z). Timings in seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
64	4.7e-3	4.6e-3	3.0e-3	2.6e-3	2.6e-3	2.7e-3
256	9.1e-2	9.0e-2	9.3e-2	1.3e-1	2.5e-1	2.6e-1
1024	2.7e+0	1.7e+0	1.0e+0	9.3e-1	1.1e+0	1.9e+0
2048	1.2e+1	6.9e+0	3.7e_0	2.2e+0	1.8e+0	2.0e+0

Table 26 3 D FFTs for 100 MHz systems - Complex-Complex Single Precision (C-C). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
32	1.2e-1 (21)	8.4e-2 (x1.4)	4.8e-2 (x2.5)	2.6e-2 (x4.6)
64	1.0e+0 (23)	6.0e-1 (x1.7)	3.3e-1 (x3.3)	1.8e-1 (x5.5)
128	8.9e+0 (25)	4.8e+0 (x1.9)	2.9e+0 (x3.1)	1.3e+0 (x6.8)

Table 27 3D FFTs for 150 MHz systems - Complex-Complex Single Precision (C-C). Timings in Seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
32	8.0e-2	6.4e-2	4.7e-2	2.4e-2	1.4e-2	2.0e-2
64	7.0e-1	4.6e-1	2.7e-1	1.4e-1	7.7e-2	5.2e-2
128	6.0e+0	3.3e+0	1.8e+0	9.2e-1	5.1e-1	4.1e-1

Table 28 3 D FFTs for 100 MHz systems - Complex-Complex Double Precision (Z-Z). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
32	1.3e-1 (19)	9.3e-2 (x1.4)	5.5e-2 (x2.4)	3.1e-2 (x4.2)
64	1.3e+0 (18)	7.6e-1 (x1.7)	4.1e-1 (x3.2)	2.2e-1 (x5.9)
128	1.2e+1 (19)	6.6e+0 (x1.8)	3.4e+0 (x3.5)	1.7e+0 (x7.0)

Table 29 3D FFTs for 150 MHz systems - Complex-Complex Double Precision (Z-Z). Timings in Seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
32	8.9e-2	7.2e-2	4.4e-2	2.8e-2	1.7e-2	1.8e-2
64	8.8e-1	5.0e-1	2.9e-1	1.6e-1	1.0e-1	7.4e-2
128	8.0e+0	4.5e+0	2.3e+0	1.2e+0	7.8e-1	5.6e-1

Table 30 3 D FFTs for 100 MHz systems - Real-Complex Single Precision (R-Z). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
32	1.1e-1 (16)	4.0e-2 (x2.7)	2.2e-2 (x5.0)	1.4e-2 (x7.8)
64	6.0e-1 (20)	3.1e-1 (x1.9)	1.6e-1 (x3.7)	1.0e-1 (x6.0)
128	4.8e+0 (22)	2.6e+0 (x1.8)	1.3e+0 (x3.7)	1.1e+0 (x4.4)

Table 31 3D FFTs for 150 MHz systems - Real-Complex Single Precision (R-Z). Timings in Seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
32	4.8e-2	2.7e-2	1.6e-2	8.3e-3	7.3e-3	7.7e-3
64	3.7e-1	2.0e-1	1.3e-1	6.0e-2	5.0e-2	5.0e-2
128	3.3e+0	1.7e+0	9.4e-1	4.9e-1	3.2e-1	3.7e-1

Table 32 3 D FFTs for 100 MHz systems - Real-Complex Double Precision (D-Z). Timings in Seconds. Mflops are shown as (NN) and Parallel Speedups as (x N.N).

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs
32	8.5e-2 (13)	5.0e-2 (x1.7)	2.8e-2 (x3.0)	1.9e-2 (x4.5)
64	7.7e-1 (14)	4.2e-1 (x1.8)	2.3e-1 (x3.3)	1.4e-1 (x5.5)
128	6.4e+0 (16)	3.5e+0 (x1.8)	1.9e+0 (x3.4)	1.1e+0 (x5.8)

Table 33 3D FFTs for 150 MHz systems - Real-Complex Double Precision (D-Z). Timings in Seconds.

SIZE	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs	32 CPUs
32	5.5e-2	3.1e-2	1.9e-2	1.3e-2	7.3e-3	1.1e-2
64	5.2e-1	2.7e-1	1.5e-1	9.1e-2	7.3e-2	9.2e-2
128	4.4e+0	2.3e+0	1.2e+0	7.1e-1	5.2e-1	5.1e-1

Section 8 Dhrystone Benchmark.

The Dhrystone benchmark is a CPU intensive, synthetic benchmark intended to test compiler and processor efficiency at handling integer instructions. In this context, ‘synthetic’ means that the benchmark code performs no meaningful task; it is a collection of instructions selected to exercise the compiler and processor. The Dhrystone 1.1 benchmark consists of 100 statements written in C. No floating point data or operations are used. The Dhrystone benchmark includes mixes of: various assignment statements, various data types and data locality, various control statements, procedure calls and parameter passing, integer arithmetic and logical operations.

Many optimizing compilers are able to eliminate a portion of the Dhrystone 1.1 benchmark code through a process known as ‘dead code removal’. Dhrystone 2.1 was developed in an attempt to reduce the amount of ‘dead’ code, and thus reduce the dependency of a machine’s Dhrystone performance on the capabilities of its compiler. Dhrystone 2.1 results are typically 10% - 15% below Dhrystone 1.1 results for the same machine.

Performance results are reported in Dhrystones per second. Higher numbers indicate higher performance. It is common practice to convert Dhrystone results to VAX MIPS by dividing test machine results by the Dhrystone performance of a VAX 11/780. The VAX MIPS results reported here assume 1 VAX MIP = 1757 Version 1.1 Dhrystones (VAX 11/780, VAX/VMS 4.2).

Dhrystone output includes results for register and non-register variations, as well as for four possible levels of optimization. The results reported here are for level 3 optimization, with registers. Valid comparisons can be made only with the same results from other machines.

Table 34 Dhrystone 1.1 on the 100 Mhz CHALLENGE

148920 Dhrystones/sec	=	84.76 Dhrystone MIPS.

Table 35 Dhrystone 2.1 on the 100 Mhz CHALLENGE

125000.0 Dhrystones/sec

Table 36 Dhrystone 2.1 on the 150 Mhz CHALLENGE

199134.7 Dhrystones/sec

Section 9 Matrix Multiply Benchmark

The matrix multiply benchmark is a multiplication of two large double-precision floating-point matrices. This operation is indicative of the kinds of computations found in large floating-point-intensive applications such as image processing and CFD.

The benchmark was run with an internal verson of BLAS dgemm matrix multiply code which has been blocked for 16 KB caches. This version of BLAS is available in Irix 5.0.1 or later. The driver for this benchmark run was LAPACK. The near-linear parallel speedups were achieved by manually selecting the block sizes in dgemm based on the matrix sizes and the number of processors that were used, to ensure a good load balance. Other technicalities include using an odd leading dimension for the matrices (leading dimension = 973 for a 960x960 matrix multiply) and decreasing the stacksize (by the limit built-in of the C shell) for runs engaging large numbers of cpus.

The benchmark scales very well through 30 CPUs which deliver in excess of 1 GFLOPs in the 100 MHz servers and 1.6 GFLOPs in the 150 MHz servers.

Table 37 Matrix Multiply Benchmark on 100 MHz Challenge

# CPUs	**1 CPU**	**2 CPUs**	**4 CPUs**	**6 CPUs**	**8 CPUs**
Peak (MFLOPs)	50	100	200	300	400
Actual (MFLOPs)	37.1	72.7	146.5	216.1	294.4.2
Speedup		1.96	3.91	5.79	7.69
# CPUs	**12 CPU**	**16 CPUs**	**20 CPUs**	**24 CPUs**	**28 CPUs**
Peak (MFLOPs)	600	800	1000	1200	1400
Actual (MFLOPs)	427.4	546.1	699.4	842.6	959[1]
Speedup	11.59	14.88	19.06	22.96	26.13
# CPUs1 CPU	**30CPUs**				
Peak (MFLOPs)	1500				
Actual (MFLOPs)	1020[2]				
Speedup	27.79				

1. Dimension of matrix = 1120, leading dimension = 1133
2. Dimension of matrix = 1080, leading dimension = 1097

The 150 MHz measurements were made under the following conditions:

Memory size: 768 MB, 2-way interleaved

IRIX: IRIX 5.1.1

Date of run: Oct 14, 1993

Table 38 Matrix Multiply Benchmark on 150 MHz Challenge

# CPUs	**1 CPU**	**2 CPUs**	**4 CPUs**	**8 CPUs**	**12 CPUs**
Peak (MFLOPs)	75	150	300	600	900
Actual (MFLOPs)	54.2	107.2	215.8	432.6	645.8
Speedup		1.98	3.98	7.98	11.92

# CPUs	**16 CPUs**	**24 CPUs**	**32 CPUs**
Peak (MFLOPs)	1200	1800	2400
Actual (MFLOPs)	823	1263.9	1638.4
Speedup	15.18	23.32	30.23