GCC UPC 4.2.3 BENCHMARKS

Author: Nenad Vukicevic <nenad@intrepid.com>
Intrepid Technology, Inc.
2155 Park Blvd
Palo Alto, CA 94306
Date: May 30, 2008

Background

Ongoing GCC UPC development work continues to provide improvements in its general optimization passes and low-level architecture specific optimizations. However, with a multitude of optimization options to choose from, it isn't at all clear which combinations provide the most benefit. In addition, with recent improvements in the UPC specific compilation aspects of shared pointer representations and run-time operations, it is difficult to determine the benefit and magnitude of gains without first establishing some standard set of benchmarks, and baselines.

For this purpose, the GCC UPC 4.2.3 compiler was benchmarked in two areas:

1. **Best compiler optimization options**: The open-source project ACOVEA was used to find the best compiler options for a certain number of code samples.
2. **Optimal UPC compiler and run-time options**: A set of micro benchmarks were integrated into a test suite that compares execution time for various compiler/run-time configuration.

Best Compiler Optimization Options

ACOVEA Overview

The open-source project ACOVEA can be found at http://www.coyotegulch.com/products/acovea/. The project summary states:

> ACOVEA (Analysis of Compiler Options via Evolutionary Algorithm) implements a genetic algorithm to find the "best" options for compiling programs with the GNU Compiler Collection (GCC) C and C++ compilers. "Best", in this context, is defined as those options that produce the fastest executable program from a given source code. ACOVEA is a C++ framework that can be extended to test other programming languages and non-GCC compilers.

The GCC UPC compiler is based on the GNU C compiler (GCC), which has hundreds of options and compile time switches. Many of the settings are influenced by various machine architectures and it is difficult to determine which of the options will make the generated code smaller and faster. By default, many developers use the *-O1*, *-O2*, or *-O3* optimization levels without any understanding of the optimizations that are enabled on each level. For example, *-O3* turns on more optimizations then *-O2*, including more aggressive function inlining and in cases where this is not desired an additional option -*-fno-inline-functions* must be provided.

Measuring the results of all possible permutations and combinations of optimization options is not practical and ACOVEA attempts to find the best compile options for a program, or code sample, by using a genetic algorithm that searches for the best solution among the set of given options. It is an iterative process that starts from a random number of specified compiler options, measures the “fitness” of the result, and creates a new set of options based on the fitness of previous results. ACOVEA counts the number of times individual options are enabled: a neutral option should occur an average number of times and a detrimental option will appear very few times, if any.

ACOVEA Test Case Description

Test cases for the ACOVEA benchmarking effort were selected among the publicly available UPC benchmarks. The table below summarizes the selected benchmarks.

CG	NAS Parallel Benchmarks 2.4 UPC implementations – CG The UPC implementations were developed by HPCL-GWU and are derived from the OpenMP version (developed by RWCP) and from the MPI version (developed by NAS).	Berkeley upc test suite NPB2.4/CG/cg.c
Edmiston	Compute Edit Distance Matrix using the Edmiston's Algorithm on MultiProcessors. This program will further be used as a sub-routine to implement Heirschberg's Divide and Conquer Algorithm. i.e a hybrid of Heirschberg and Edmiston's Algorithm.	Berkeley upc test suite benchmarks/Edmiston.upc
lu	Parallel dense blocked LU factorization (no pivoting)	Berkeley upc test suite splash2/lu/non_contiguous_blocks/lu.upc
matmult	Various Matrix example programs.	Berkeley upc test suite benchmarks/matmult/*

For the purpose of ACOVEA testing, the following changes were made to each benchmark:

1. All informational and diagnostic messages that don't signal test failure were removed.
2. All benchmark timing code was removed.
3. Special logic required by ACOVEA was added:
    a. Start the timing loop:
    ```
    struct timespec start, stop;
    clock_gettime(CLOCK_REALTIME,&start);
    ```
    b. End the timing loop:
    ```
    clock_gettime(CLOCK_REALTIME,&stop);
    double run_time = (stop.tv_sec - start.tv_sec) + (double)(stop.tv_nsec -
    start.tv_nsec) / 1000000000.0;
    fprintf(stdout,"%f",run_time);
    ```

In all of the benchmarks, only the most critical computational code sequence was measured, thus eliminating the overhead of the setup/initialization.

ACOVEA Configuration for GCC UPC 4.2.3

ACOVEA is driven by a special XML based configuration file that specifies various commands and options used by ACOVEA. Some sample configurations for previous versions of GCC are already available. In developing a configuration file for GCC UPC 4.2.3 the following factors were considered:

1. *The -O2* optimization switch was selected as the default optimization.
2. All optimization options that can be taken away or added to *–O2* are listed as ACOVEA's options for finding the optimal solution.
3. *The -O3* optimization is specified as an additional ACOVEA run for reference only.

The following file is a sample GCC UPC 4.2.3 ACOVEA configuration file, for an AMD Opteron based system:

```
<?xml version="1.0"?>
<acovea_config>
    <acovea version="5.0.0" />
    <description value="upc Opteron (AMD64/x86_64)" />
    <get_version value="upc --version" />

    <quoted_options value="false" />

    <prime command="upc"
           flags='-O2 -fupc-threads-4 ACOVEA_OPTIONS -o ACOVEA_OUTPUT ACOVEA_INPUT -
lrt'/>

    <baseline description="-O3"
              command="upc"
              flags='-O3 -fupc-threads-4 ACOVEA_OPTIONS -o ACOVEA_OUTPUT ACOVEA_INPUT -
lrt'/>

    <!-- A list of flags that will be "evolved" by ACOVEA -->
    <flags>
        <!-- O2 options -->
        <flag type="simple" value="-fno-thread-jumps" />
        <flag type="simple" value="-fno-crossjumping" />
        <flag type="simple" value="-fno-optimize-sibling-calls" />
        <flag type="simple" value="-fno-cse-follow-jumps" />
        <flag type="simple" value="-fno-cse-skip-blocks" />
        <flag type="simple" value="-fno-gcse" />
        <flag type="simple" value="-fno-gcse-lm" />
        <flag type="simple" value="-fno-expensive-optimizations" />
        <flag type="simple" value="-fno-rerun-cse-after-loop" />
        <flag type="simple" value="-fno-caller-saves" />
        <flag type="simple" value="-fno-peephole2" />
        <flag type="simple" value="-fno-schedule-insns" />
        <flag type="simple" value="-fno-schedule-insns2" />
        <flag type="simple" value="-fno-sched-interblock" />
        <flag type="simple" value="-fno-sched-spec" />
        <flag type="simple" value="-fno-regmove" />
        <flag type="simple" value="-fno-strict-aliasing" />
        <flag type="simple" value="-fno-strict-overflow" />
        <flag type="simple" value="-fno-delete-null-pointer-checks" />
        <flag type="simple" value="-fno-reorder-blocks" />
        <flag type="simple" value="-fno-reorder-functions" />
        <flag type="simple" value="-fno-align-functions"/>
        <flag type="simple" value="-fno-align-jumps" />
        <flag type="simple" value="-fno-align-loops" />
        <flag type="simple" value="-fno-align-labels" />
        <flag type="simple" value="-fno-tree-pre" />
        <flag type="simple" value="-fno-tree-vrp" />

        <!-- O3 options -->
        <flag type="simple" value="-finline-functions" />
        <flag type="simple" value="-funswitch-loops" />
        <flag type="simple" value="-fgcse-after-reload" />

        <!-- Additional options -->
        <flag type="simple" value="-ffloat-store" />
        <flag type="simple" value="-fprefetch-loop-arrays" />
        <flag type="simple" value="-fno-inline" />
        <flag type="simple" value="-fpeel-loops" />
        <flag type="simple" value="-ftracer" />
        <flag type="simple" value="-funswitch-loops" />
        <flag type="enum"   value="-funroll-loops|-funroll-all-loops" />
        <flag type="enum"   value="-fbranch-target-load-optimize|-fbranch-target-load-
optimize2" />
        <flag type="simple" value="-fmodulo-sched" />
        <flag type="simple" value="-fno-function-cse" />
        <flag type="simple" value="-fgcse-sm" />
        <flag type="simple" value="-fgcse-las" />
        <flag type="simple" value="-freschedule-modulo-scheduled-loops" />
        <flag type="simple" value="-ftree-loop-im" />
        <flag type="simple" value="-ftree-loop-ivcanon" />
        <flag type="simple" value="-fivopts" />
        <flag type="simple" value="-ftree-vectorize" />
        <flag type="simple" value="-fvariable-expansion-in-unroller" />
        <flag type="simple" value="-fforce-addr" />
        <flag type="simple" value="-fivopts" />
        <flag type="simple" value="-frerun-loop-opt" />
        <flag type="simple" value="-freorder-blocks" />
        <flag type="simple" value="-funit-at-a-time" />

        <!-- Options specific to "fast math" -->
        <flag type="simple" value="-fno-math-errno" />
        <flag type="simple" value="-funsafe-math-optimizations" />
        <flag type="simple" value="-fno-trapping-math" />
        <flag type="simple" value="-ffinite-math-only" />
        <flag type="simple" value="-fno-signaling-nans" />
        <flag type="simple" value="-fcx-limited-range" />

        <!-- Tuning options that have a numeric value -->
        <flag type="tuning" value="-finline-limit" default="600" min="100"
                       max="10000" step="100" separator="=" />
    </flags>

</acovea_config>
```

The above configuration can be used for UPC programs that are contained in one source file. To run ACOVEA, the following command is used:

```
runacovea –config sample.config –input lu.upc
```

For programs that have code located in multiple source files, a simple Makefile must be created. (The CG benchmark required a Makefile.)

ACOVEA Results

The ACOVEA tests were run on an Opteron based system with two dual core 2.2GHz processors and 8 Gb of memory. For each of the benchmarks, the best switch combination was found. The benchmark results presented in this study are affected by the following factors in our testing environment:

1. The number of ACOVEA generations was lowered from a default of 20 to a value of 10 in order to shorten the overall execution time.
2. Each test was tuned to insure that it executed in between .5 and 1 second.
3. The benchmark system does not have capabilities to isolate CPU’s specifically for benchmarking only. Various normal background system functions were also executed while the benchmarks were run.

With the above constraints, we were able to limit the overall ACOVEA execution time to less then ten (10) hours for all four benchmarks.

The following ACOVEA options were used:

```
  # of populations: 5
   population size: 40
     survival rate: 10% (4)
    migration rate: 5% (2)
     mutation rate: 1%
    crossover rate: 100%
   fitness scaling: sigma
generations to run: 10
random number seed: 937318890
      testing mode: speed
```

For example, the CG benchmark produced the following best combination of switches:

> *-O2 -fno-thread-jumps -fno-crossjumping -fno-cse-skip-blocks -fno-peephole2 -fno-sched-interblock -fno-sched-spec -fno-delete-null-pointer-checks -fno-align-functions -fno-align-loops -fno-align-labels -fno-tree-pre -finline-functions -funswitch-loops -fgcse-after-reload -fno-inline -ftracer -funroll-loops -fmodulo-sched -fno-function-cse -fgcse-las -freschedule-modulo-scheduled-loops -ftree-loop-ivcanon -ftree-vectorize -fvariable-expansion-in-unroller -fivopts -fno-math-errno -funsafe-math-optimizations -fno-trapping-math -ffinite-math-only -fno-signaling-nans -fcx-limited-range -finline-limit=500*

The options that showed the most improvement are:

> *-fno-align-loops (1.567)*
> *-ftracer (2.031)*
> *-fvariable-expansion-in-unroller (2.124)*
> *-fno-math-errno (1.567)*
> *-funsafe-math-optimizations (1.938)*

The most pessimistic options are:

> *-fno-schedule-insns2 (-2.145)*
> *-fno-strict-overflow (-2.145)*
> *-ffloat-store (-2.145)*
> *-fprefetch-loop-arrays (-2.145)*
> *-fbranch-target-load-optimize2 (-1.588)*

A relative improvement graph for the CG benchmark was also produced:

```
  Acovea's Best: ************************************     (0.223329)
Acovea's Common: **************************************   (0.227566)
            -O3: **************************************** (0.246804)
```

Detailed ACOVEA benchmark results can be found in Appendix A.

Conclusions

1. The *–O3* optimization switch consistently produced worse results then *–O2* in combination with other switches on all four benchmarks.
2. The *–O2* switch on its own is not enough to produce an optimal execution time. Many optimization switches interact with each other; therefore, the best combination can be found only through testing.
3. The ACOVEA best solutions for all four benchmarks were compared and it was found that some of the switches appear in all four of the benchmark best cases:

   -freschedule-modulo-scheduled-loops
   -fno-tree-pre
   -fivopts

   And others appear in three (3) out of four of the best benchmark cases:

   -funsafe-math-optimizations
   -funroll-loops
   -ftree-vectorize
   -ftree-loop-im
   -fno-strict-aliasing
   -fno-signaling-nans
   -fno-rerun-cse-after-loop
   -fno-math-errno
   -fno-inline
   -fno-function-cse
   -fno-delete-null-pointer-checks
   -fno-cse-skip-blocks
   -fno-align-loops
   -fno-align-functions
   -fmodulo-sched
   -fgcse-las

It is obvious from the above that the *-O2* optimization level is not enough on its own to produce the fastest benchmark times. Some optimization options must be disabled and some additional option must be enabled.

Micro Benchmarks

As GCC UPC evolves, many compilation options and run-time optimizations are implemented to provide for both ease-of-use and runtime efficiencies. As new capabilities are added, it is important to measure progress in terms of the execution time of the generated code.

The latest GCC UPC 4.2.3 release offers some new features that directly affect both code size and execution time:

1. In general, inlining the shared space access routines improves the execution time of a program compiled by GCC UPC.
2. Implementation of a packed shared pointer representation also improves the program's execution time as the size of a shared pointer is halved.
3. CPU scheduling affinity and memory affinity generally improves UPC program execution times on computing platforms that support those features.

Various members of the UPC community have developed benchmarks to measure the low level latencies for accessing remote and local data. Some of the benchmarks have unique UPC implementations, and others were converted from well established parallel execution environments (MPI, OpenMP, etc.).

Micro Benchmarks: Description

For the purpose of establishing UPC relative performance, the following tests were selected:

1. A collection of micro benchmarks developed by Zhang Zhang at MTU. The following tests were used:

baseline	This benchmark measures the baseline performance for shared reads and writes (local and remote).
local	This benchmark measures the overhead of manipulating pointers-to-shared with different layout specifiers. - shared [B] *ptr General block size, assume this is the baseline. - shared [0] *ptr Indefinite block size. - shared [1] *ptr Unit block size. This benchmark contains local shared accesses only.

coalesce	This benchmark measures the compiler's ability to overlap dependent-free accesses.
natural_ring	Based on stream_omp.c (by John McCalpin) and stream.c (by Sebastien Chauvin)
stream_strings	This program measures memory transfer rates in MB/s for simple computational kernels coded in UPC. Based on the stream_omp.c originally developed by John D. McCalpin.
vector	This benchmark measures if the UPC compiler exploits vectorization, coalescing, and caching for spatial locality.

2. The pointer arithmetic benchmarks provided in the Berkeley UPC test suite:

ptrmath-volatile	Various shared pointer arithmetic and remote access.

All the benchmarks were changed to provide:

1. Longer execution times that minimize the startup time and possible kernel process rescheduling overhead.
2. Automatic tabulation of results to produce an Excel spreadsheet as a final product.

Compiler and run-time configurations were carefully selected with emphasis on the following new compiler features:

1. **Shared pointer representation**
   Packed shared pointer representation v. Structure representation
2. **Shared data access routines**
   Inlined access routines v. Library call routines

A default optimization, *-O2*, was chosen for all configurations except one.

The following table lists the various configurations that were used in benchmarking. GCC UPC 4.0.3.5 (Configuration 1) was selected as a baseline for all comparisons.

	Compiler Version	Pointer Representation	Access Code Inline	Optimization	Label
1	GCC UPC 4.0.3.5	Struct	N/A	-O2	GUPC 4.0
2	GCC UPC 4.2.3.3	Struct	NO	-O2	GUPC Struct Lib
3	GCC UPC 4.2.3.3	Packed	NO	-O2	GUPC Lib
4	GCC UPC 4.2.3.3	Struct	YES	-O2	GUPC Struct
5	GCC UPC 4.2.3.3	Packed	YES	-O2	GUPC
6	GCC UPC 4.2.3.3	Packed	YES	-O3	GUPC (O3)

Figure 1 Benchmark configurations

Micro Benchmark Results

Benchmarks were run on a system with two dual core 2.2GHz Opteron processors with 8Gb of main memory, running the Fedora Core 8 Linux operating system. The system was run in “single user mode” with a minimal set of services running.

All of the benchmark results can be found in Appendix B together with raw benchmark results and configuration details.

STREAMS UPC BENCHMARK

The Streams benchmark shows significant improvements for the GCC UPC packed shared pointer representation with inlined shared data access routines.

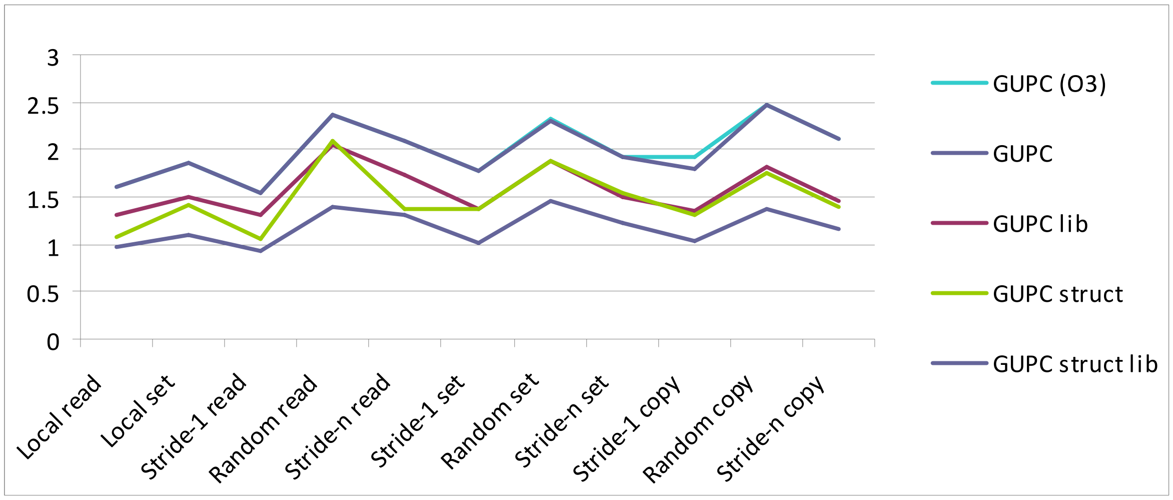


Figure 2 Streams UPC benchmark results relative to GUPC 4.0 configuration

The best result are achieved for the **GUPC** configuration (packed shared pointer, inline access routines, -O2 optimization) and **GUPC (O3)** (packed shared pointer, inline access routines, -O3 optimization).

BERKELEY POINTER ARITHMETIC

The pointer arithmetic benchmark also shows improvements over the previous version of the GCC UPC compiler.

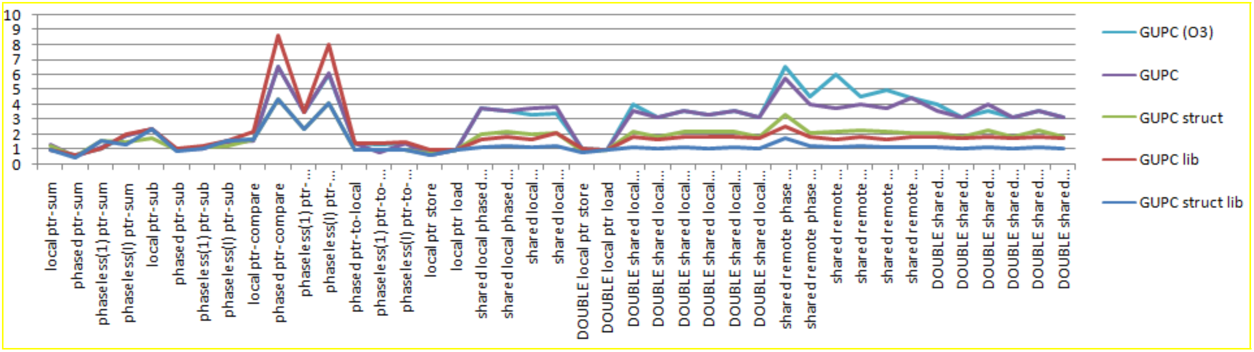


Figure 3 Berkeley pointer arithmetic benchmark results relative to GUPC 4.0 configuration

Again, the best results are achieved for **GUPC (O3)** (packed shared pointer, inline access routines, -O3 optimization) and **GUPC** (packed shared pointer, inline access routines, -O2 optimization).

Conclusions

Both the GCC UPC 4.2.3.3 compiler and the inlined run-time produce significant improvements over the previous version of GCC UPC, 4.0.3.5.

Overall, the combination of a packed shared pointer representation and inlining of the shared data access routines provides the best benchmark results for the following reasons:

1. The packed pointer representation is more efficient as its size is only 64 bits versus 128 bits for the structure representation. This improvement in execution efficiency comes at the cost of a few limitations: the maximum layout specifier (block size) is 65536, and the maximum number of threads of 4096.

2. Inlining of the shared data access routines minimizes the overhead of procedure calls. On the other hand, inlining the shared access routines produces larger code which might not be suitable for machines with memory restrictions on the compute nodes.

Appendix A - ACOVEA Benchmark Results

```
Acovea 5.1.1 (compiled Mar 31 2008 08:58:52)
config description: upc Opteron (AMD64/x86_64) (version )
application version: make xgcc (GCC) 4.2.3 20080529 (dev) (GCC UPC 4.2.3-3)
```

CG

```
Optimistic options:

                        -fno-align-loops  (1.567)
                                -ftracer  (2.031)
        -fvariable-expansion-in-unroller  (2.124)
                         -fno-math-errno  (1.567)
             -funsafe-math-optimizations  (1.938)

Pessimistic options:

                    -fno-schedule-insns2  (-2.145)
                    -fno-strict-overflow  (-2.145)
                           -ffloat-store  (-2.145)
                  -fprefetch-loop-arrays  (-2.145)
          -fbranch-target-load-optimize2  (-1.588)

Acovea's Best-of-the-Best:
make -s OPTIM=-O2 SETTINGS=-fno-thread-jumps -fno-crossjumping -fno-cse-skip-blocks -
fno-peephole2 -fno-sched-interblock -fno-sched-spec -fno-delete-null-pointer-checks -
fno-align-functions -fno-align-loops -fno-align-labels -fno-tree-pre -finline-
functions -funswitch-loops -fgcse-after-reload -fno-inline -ftracer -funroll-loops -
fmodulo-sched -fno-function-cse -fgcse-las -freschedule-modulo-scheduled-loops -ftree-
loop-ivcanon -ftree-vectorize -fvariable-expansion-in-unroller -fivopts -fno-math-
errno -funsafe-math-optimizations -fno-trapping-math -ffinite-math-only -fno-
signaling-nans -fcx-limited-range -finline-limit=500  OUTPUT=/tmp/ACOVEAD0A75466
INPUT=cg.c

Acovea's Common Options:
make -s OPTIM=-O2 SETTINGS=-ftracer -funroll-loops -fvariable-expansion-in-unroller -
fivopts -funsafe-math-optimizations -fno-signaling-nans  OUTPUT=/tmp/ACOVEADAEA3A82
INPUT=cg.c

-O3:
make -s OPTIM=-O3 SETTINGS= OUTPUT=/tmp/ACOVEA2A2AD13A INPUT=cg.c


A relative graph of fitnesses:

     Acovea's Best-of-the-Best: ***********************************         (0.223329)
       Acovea's Common Options: ************************************        (0.227566)
                           -O3: ****************************************    (0.246804)
```

EDMISTON

```
Optimistic options:

                        -fno-align-jumps  (2.305)
                        -fno-align-loops  (1.653)
                      -finline-functions  (2.741)

Pessimistic options:

                    -fno-schedule-insns2  (-2.59)
                            -fno-regmove  (-1.611)
                             -fno-inline  (-2.372)
           -fbranch-target-load-optimize  (-1.937)
                          -finline-limit  (-1.502)

Acovea's Best-of-the-Best:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O2 -fupc-threads-4 -fno-cse-
follow-jumps -fno-expensive-optimizations -fno-rerun-cse-after-loop -fno-caller-saves
-fno-peephole2 -fno-strict-aliasing -fno-strict-overflow -fno-delete-null-pointer-
checks -fno-reorder-functions -fno-align-loops -fno-tree-pre -fno-tree-vrp -finline-
functions -funswitch-loops -ffloat-store -fprefetch-loop-arrays -fpeel-loops -funroll-
loops -fbranch-target-load-optimize -fgcse-las -freschedule-modulo-scheduled-loops -
ftree-loop-im -ftree-vectorize -fvariable-expansion-in-unroller -fivopts -frerun-loop-
opt -funit-at-a-time -fno-math-errno -fno-signaling-nans -o /tmp/ACOVEA31B26B12
Edmiston.upc -lrt

Acovea's Common Options:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O2 -fupc-threads-4 -fno-peephole2
-finline-functions -funswitch-loops -funroll-loops -o /tmp/ACOVEA1F37B667 Edmiston.upc
-lrt

-O3:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O3 -fupc-threads-4 -o
/tmp/ACOVEAA531D3EC Edmiston.upc -lrt


A relative graph of fitnesses:

     Acovea's Best-of-the-Best: *************************************       (0.480189)
       Acovea's Common Options: **************************************      (0.483796)
                           -O3: ****************************************    (0.502014)
```

LU

```
Optimistic options:

                        -frerun-loop-opt  (1.708)
             -funsafe-math-optimizations  (2.66)

Pessimistic options:

                    -fno-schedule-insns2  (-2.52)
                   -fno-sched-interblock  (-1.886)
                         -fno-sched-spec  (-1.886)
                            -fno-regmove  (-1.991)
                           -ffloat-store  (-2.097)
          -fbranch-target-load-optimize2  (-1.991)

Acovea's Best-of-the-Best:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O2 -fupc-threads-4 -fno-optimize-
sibling-calls -fno-cse-skip-blocks -fno-gcse -fno-gcse-lm -fno-expensive-optimizations
-fno-rerun-cse-after-loop -fno-schedule-insns -fno-strict-aliasing -fno-delete-null-
pointer-checks -fno-align-functions -fno-align-loops -fno-tree-pre -funswitch-loops -
fno-inline -fpeel-loops -ftracer -funroll-loops -fbranch-target-load-optimize -
fmodulo-sched -fno-function-cse -fgcse-sm -freschedule-modulo-scheduled-loops -ftree-
loop-im -fivopts -frerun-loop-opt -freorder-blocks -funit-at-a-time -funsafe-math-
optimizations -o /tmp/ACOVEA05728C66 lu.upc -lrt

Acovea's Common Options:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O2 -fupc-threads-4 -fno-inline -
fpeel-loops -ftracer -fivopts -frerun-loop-opt -funsafe-math-optimizations -o
/tmp/ACOVEA972D52C4 lu.upc -lrt

-O3:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O3 -fupc-threads-4 -o
/tmp/ACOVEA79CBDE43 lu.upc -lrt


A relative graph of fitnesses:

     Acovea's Best-of-the-Best: ************************************       (0.710281)
       Acovea's Common Options: **************************************     (0.732395)
                           -O3: ****************************************    (0.745207)
```

MATRIX

```
Optimistic options:

                       -fno-thread-jumps  (1.693)
             -fno-optimize-sibling-calls  (1.5)
                       -fno-align-labels  (1.596)
                           -fno-tree-pre  (2.367)
                          -fmodulo-sched  (1.982)

Pessimistic options:

                               -fno-gcse  (-2.355)
                    -fno-schedule-insns2  (-2.162)
                           -ffloat-store  (-2.355)
                                -ftracer  (-1.68)

Acovea's Best-of-the-Best:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O2 -fupc-threads-4 -fno-thread-
jumps -fno-crossjumping -fno-optimize-sibling-calls -fno-cse-skip-blocks -fno-rerun-
cse-after-loop -fno-sched-interblock -fno-sched-spec -fno-strict-aliasing -fno-strict-
overflow -fno-align-functions -fno-align-labels -fno-tree-pre -fno-tree-vrp -
funswitch-loops -fgcse-after-reload -fprefetch-loop-arrays -fno-inline -funswitch-
loops -funroll-all-loops -fmodulo-sched -fno-function-cse -fgcse-las -freschedule-
modulo-scheduled-loops -ftree-loop-im -ftree-loop-ivcanon -fivopts -ftree-vectorize -
fno-math-errno -funsafe-math-optimizations -fno-signaling-nans -fcx-limited-range -o
/tmp/ACOVEA2B7C729D mat-main.upc -lrt

Acovea's Common Options:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O2 -fupc-threads-4 -fno-thread-
jumps -fno-crossjumping -fno-optimize-sibling-calls -fno-sched-interblock -fno-align-
labels -fno-tree-pre -fno-tree-vrp -fno-inline -fmodulo-sched -fno-signaling-nans -o
/tmp/ACOVEAF0F8F773 mat-main.upc -lrt

-O3:
/eng/upc/dev/nenad/gcc-upc-dev/wrk-packed/gcc/xupc -O3 -fupc-threads-4 -o
/tmp/ACOVEA534807C3 mat-main.upc -lrt


A relative graph of fitnesses:

     Acovea's Best-of-the-Best: *****************************               (0.220395)
       Acovea's Common Options: *********************************           (0.24399)
                           -O3: ****************************************    (0.279688)
```

Appendix B – Micro Benchmark Results

GCC UPC Micro Benchmarks Thu May 29 18:00:15 PDT 2008

🟩	better result (>1.2)
🟧	worse result (<0.83)

baseline	**Avg**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
REMOTE WRITE LATENCY (msec/double)	0.0847	0.98	0.87	0.85	0.95	1.05
REMOTE READ LATENCY (msec/double)	0.0882	0.94	1	1.04	1.14	1.08

coalesce	**Avg**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
REMOTE WRITE LATENCY (msec/double)	0.0201	1.09	1	0.91	1.81	1.57
REMOTE READ LATENCY (msec/double)	0.0175	0.98	1.22	1.18	1.61	1.59

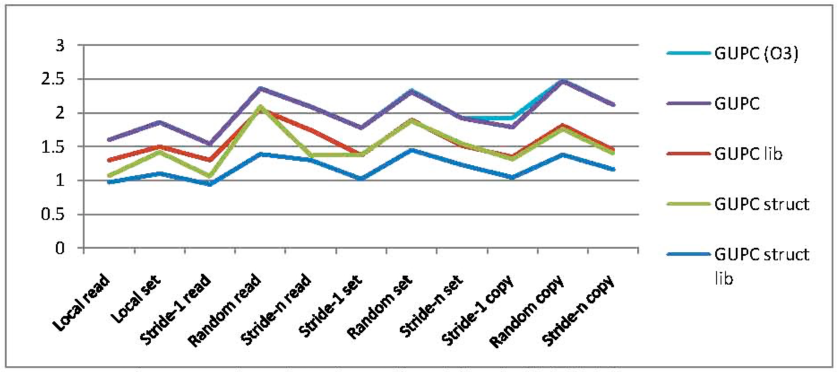
local	**Avg**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
pB WRITE (msec/double)	0.0246	0.83	0.85	0.88	0.96	0.96
pB READ (msec/double)	0.0219	0.76	0.66	0.72	0.83	0.83
p0 WRITE (msec/double)	0.0856	0.99	0.96	0.97	0.95	0.95
p0 READ (msec/double)	0.0919	0.99	1.61	1.46	1.65	1.65

natural_ring_upc	**Rate (Refs/s)**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
Random read	16152440	1	1.12	1.12	1.11	1.1
Random set	14618884	0.99	1.06	1.08	1.78	1.77

stream_strings_relaxed	**Rate (MB/s)**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
memget (local)	2280.7836	1	1	1	1	1
memput (local)	2271.8271	1	1	1	1	1
memset (local)	5476.4864	1	1	1	1	1
memcpy (local)	4534.873	1	1	1	1	1
memget (remote)	2540.0012	0.9	0.89	0.9	0.9	0.89
memput (remote)	2391.9271	0.95	0.95	0.95	0.95	0.95
memset (remote)	2921.8419	1.88	1.88	1.88	1.88	1.87
memcpy (remote)	3768.4249	1.2	1.2	1.21	1.2	1.2
memcpy (shared)	3760.9908	1.35	1.35	1.35	1.35	1.35

stream_strings_strict	**Rate (MB/s)**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
memget (local)	2281.2798	1	1	1	1	1
memput (local)	2272.0732	1	1	1	1	1
memset (local)	5479.3481	1	1	1	1	1
memcpy (local)	4530.8315	1	1	1	1	1
memget (remote)	2539.1939	0.9	0.89	0.9	0.9	0.89
memput (remote)	2393.77	0.95	0.95	0.95	0.95	0.95
memset (remote)	2922.5035	1.88	1.88	1.88	1.87	1.87
memcpy (remote)	3768.5942	1.2	1.2	1.2	1.2	1.2
memcpy (shared)	3770.2033	1.34	1.34	1.35	1.34	1.34

stream_upc	**Rate (Refs/s)**					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
Local read	69098913	0.97	1.3	1.07	1.6	1.6
Local set	64698957	1.1	1.5	1.42	1.86	1.86
Stride-1 read	69793397	0.94	1.3	1.06	1.54	1.54
Random read	12258599	1.39	2.05	2.09	2.36	2.36
Stride-n read	30171376	1.3	1.74	1.37	2.09	2.09
Stride-1 set	67384872	1.02	1.37	1.38	1.78	1.78
Random set	16773995	1.45	1.89	1.88	2.31	2.33
Stride-n set	44419893	1.23	1.51	1.54	1.92	1.92
Stride-1 copy	70312881	1.04	1.35	1.31	1.79	1.93
Random copy	13771503	1.38	1.81	1.76	2.47	2.48
Stride-n copy	45421412	1.16	1.45	1.4	2.12	2.12

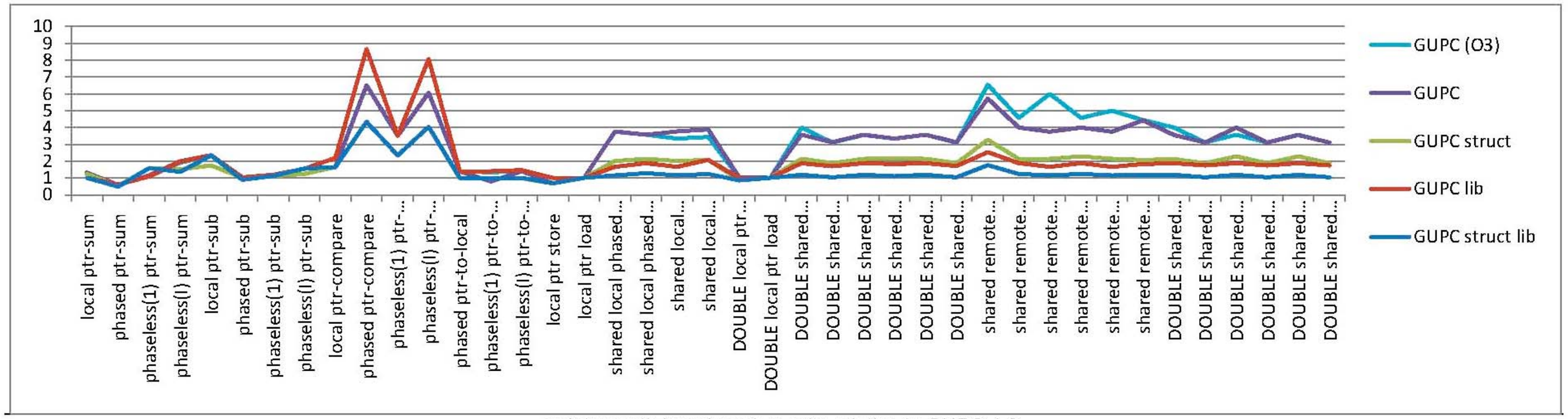


stream upc benchmark results relative to GUPC 4.0

vector	Avg					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
REMOTE WRITE LATENCY (msec/double)	0.0167	1.06	1.4	1.3	1.84	1.82
REMOTE READ LATENCY (msec/double)	0.0167	1.12	1.52	1.43	1.9	1.78

ptrmath	us/iter					
	GUPC 4.0	GUPC struct lib	GUPC lib	GUPC struct	GUPC	GUPC (O3)
local ptr-sum	0.003149	1.01	1	1.23	1.31	1
phased ptr-sum	0.011392	0.48	0.61	0.53	0.57	0.61
phaseless(1) ptr-sum	0.00767	1.58	1.09	1.58	1.14	1.08
phaseless(l) ptr-sum	0.009218	1.36	1.98	1.52	1.98	1.9
local ptr-sub	0.00319	2.34	2.33	1.75	2.34	2.33
phased ptr-sub	0.003645	0.89	1.04	1	1	1
phaseless(1) ptr-sub	0.003069	1.12	1.19	1.12	1.12	1.12
phaseless(l) ptr-sub	0.002833	1.56	1.55	1.24	1.56	1.55
local ptr-compare	0.002993	1.64	2.19	1.64	1.64	1.64
phased ptr-compare	0.011847	4.34	8.67	4.34	6.51	6.5
phaseless(1) ptr-compare	0.006379	2.34	3.5	2.34	3.5	3.5
phaseless(l) ptr-compare	0.011035	4.04	8.07	4.04	6.06	6.05
phased ptr-to-local	0.012759	1	1.4	0.97	1.4	1.4
phaseless(1) ptr-to-local	0.012759	1	1.4	0.97	0.79	1.33
phaseless(l) ptr-to-local	0.012758	1	1.47	0.97	1.4	1.4
local ptr store	0.000912	0.67	1	0.67	0.67	0.67
local ptr load	0.001367	1	1	1	1	1
shared local phased ptr store	0.013671	1.16	1.67	2	3.76	3.75
shared local phased ptr load	0.014581	1.28	1.88	2.14	3.56	3.56
shared local phaseless(1) store	0.01367	1.16	1.67	2	3.76	3.33
shared local phaseless(l) load	0.014127	1.24	2.07	2.07	3.88	3.44
DOUBLE local ptr store	0.001549	0.85	1.01	0.87	1.01	1.01
DOUBLE local ptr load	0.001367	1	1	1	1	1
DOUBLE shared local phased ptr store	0.014583	1.19	1.88	2.14	3.56	4
DOUBLE shared local phased ptr load	0.01276	1.04	1.7	1.87	3.12	3.11
DOUBLE shared local phaseless(1) store	0.014582	1.19	1.88	2.14	3.56	3.55
DOUBLE shared local phaseless(1) load	0.013671	1.11	1.83	2.15	3.34	3.33
DOUBLE shared local phaseless(l) store	0.014582	1.19	1.88	2.14	3.56	3.56
DOUBLE shared local phaseless(l) load	0.012765	1.04	1.71	1.87	3.12	3.11
shared remote phase ptr store	0.020844	1.76	2.54	3.27	5.72	6.54
shared remote phase ptr load	0.01456	1.23	1.88	2.13	3.99	4.57
shared remote phaseless(1) ptr store	0.013648	1.15	1.67	2.14	3.74	6
shared remote phaseless(1) ptr load	0.014561	1.23	1.88	2.29	3.99	4.57
shared remote phaseless(l) ptr store	0.013649	1.15	1.66	2.14	3.74	5
shared remote phaseless(l) ptr load	0.014105	1.19	1.82	2.07	4.42	4.43

DOUBLE shared remote phase ptr store	0.014561	1.19	1.88	2.13	3.55	4
DOUBLE shared remote phase ptr load	0.012741	1.04	1.74	1.87	3.11	3.11
DOUBLE shared remote phaseless(1) ptr store	0.014559	1.19	1.88	2.29	3.99	3.56
DOUBLE shared remote phaseless(1) ptr load	0.012739	1.04	1.74	1.87	3.11	3.11
DOUBLE shared remote phaseless(l) ptr store	0.014559	1.19	1.88	2.29	3.55	3.56
DOUBLE shared remote phaseless(l) ptr load	0.01274	1.04	1.75	1.87	3.11	3.11



pointer math benchmark results relative to GUPC 4.0

Micro Benchmarks Data

Name: GUPC 4.0
Info: GCCUPC 4.0.3.5 with optimization
Time: Thu May 29 17:14:23 PDT 2008
Configuration: conf/gccupc4-0.conf
Optimization: -O2
Threads: 4
MTU run option: -s 1000
MTU Benchamrks

baseline			
ARRAY LENGTH:	80000000		
NUMBER OF ACCESSES:	10000		
NUMBER OF TEST REPEATS:	1000		
Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0847	0.0845 (thread 0)	0.0848 (thread 1)
REMOTE READ LATENCY (msec/double)	0.0882	0.0881 (thread 0)	0.0883 (thread 3)
coalesce			
ARRAY LENGTH:	8000000		
NUMBER OF ACCESSES:	1000		
NUMBER OF TEST REPEATS:	1000		
Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0201	0.0201 (thread 3)	0.0201 (thread 1)
REMOTE READ LATENCY (msec/double)	0.0175	0.0175 (thread 3)	0.0176 (thread 2)
local			
ARRAY LENGTH:	8000000		
NUMBER OF ACCESSES:	100000		
NUMBER OF REPEATS:	1000		
Function	Avg	Min	Max
pB WRITE (msec/double)	0.0246	0.0246 (thread 3)	0.0246 (thread 0)
pB READ (msec/double)	0.0219	0.0219 (thread 3)	0.0219 (thread 0)
p0 WRITE (msec/double)	0.0856	0.0856 (thread 3)	0.0856 (thread 1)
p0 READ (msec/double)	0.0919	0.0919 (thread 3)	0.0919 (thread 0)

natural_ring_upc

Function	Rate (Refs/s)	Avg time	Min time	Max time
Random read:	16152440	0.0124	0.0124	0.0125
Random set:	14618884	0.014	0.0137	0.0144

stream_strings_relaxed

Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2280.7836	0.0176	0.0175	0.0179
memput (local):	2271.8271	0.0177	0.0176	0.018
memset (local):	5476.4864	0.0073	0.0073	0.0075
memcpy (local):	4534.873	0.0176	0.0176	0.0177
memget (remote):	2540.0012	0.0158	0.0157	0.0158
memput (remote):	2391.9271	0.0183	0.0167	0.0305
memset (remote):	2921.8419	0.0138	0.0137	0.0144
memcpy (remote):	3768.4249	0.0216	0.0212	0.0231
memcpy (shared):	3760.9908	0.0232	0.0213	0.0358

stream_strings_strict

Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2281.2798	0.0175	0.0175	0.0175
memput (local):	2272.0732	0.0176	0.0176	0.0176
memset (local):	5479.3481	0.0073	0.0073	0.0074
memcpy (local):	4530.8315	0.0177	0.0177	0.0177
memget (remote):	2539.1939	0.0158	0.0158	0.0164
memput (remote):	2393.77	0.0168	0.0167	0.0172
memset (remote):	2922.5035	0.0138	0.0137	0.0143
memcpy (remote):	3768.5942	0.0214	0.0212	0.0229
memcpy (shared):	3770.2033	0.0232	0.0212	0.0393

stream_upc

Function	Rate (Refs/s)	Avg time	Min time	Max time
Local read:	69098913	0.0036	0.0036	0.0036
Local set:	64698957	0.0039	0.0039	0.0039
Stride-1 read:	69793397	0.0036	0.0036	0.0036
Random read:	12258599	0.0204	0.0204	0.0204
Stride-n read:	30171376	0.0083	0.0083	0.0083
Stride-1 set:	67384872	0.0037	0.0037	0.0037
Random set:	16773995	0.0149	0.0149	0.015
Stride-n set:	44419893	0.0056	0.0056	0.0056

Stride-1 copy:	70312881	0.0071	0.0071	0.0071
Random copy:	13771503	0.0364	0.0363	0.0366
Stride-n copy:	45421412	0.011	0.011	0.011
vector				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
VECTOR LENGTH:	64			
NUMBER OF TESTS REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0167	0.0166 (thread 2)	0.0167 (thread 0)	
REMOTE READ LATENCY (msec/double)	0.0167	0.0167 (thread 2)	0.0167 (thread 0)	
ptrmath				
iterations 10000000				
Function	us/iter	total (sec)		
local ptr-sum	0.003149	0.031492		
phased ptr-sum	0.011392	0.113918		
phaseless(1) ptr-sum	0.00767	0.076698		
phaseless(I) ptr-sum	0.009218	0.092176		
local ptr-sub	0.00319	0.031898		
phased ptr-sub	0.003645	0.036452		
phaseless(1) ptr-sub	0.003069	0.030687		
phaseless(I) ptr-sub	0.002833	0.028334		
local ptr-compare	0.002993	0.029929		
phased ptr-compare	0.011847	0.118474		
phaseless(1) ptr-compare	0.006379	0.063795		
phaseless(I) ptr-compare	0.011035	0.110347		
phased ptr-to-local	0.012759	0.127592		
phaseless(1) ptr-to-local	0.012759	0.127593		
phaseless(I) ptr-to-local	0.012758	0.127584		
local ptr store	0.000912	0.009121		
local ptr load	0.001367	0.013667		
shared local phased ptr store	0.013671	0.136709		
shared local phased ptr load	0.014581	0.145814		
shared local phaseless(1) store	0.01367	0.136699		
shared local phaseless(I) load	0.014127	0.141265		
DOUBLE local ptr store	0.001549	0.015492		

DOUBLE local ptr load	0.001367	0.013674
DOUBLE shared local phased ptr store	0.014583	0.145829
DOUBLE shared local phased ptr load	0.01276	0.127597
DOUBLE shared local phaseless(1) store	0.014582	0.145818
DOUBLE shared local phaseless(1) load	0.013671	0.136705
DOUBLE shared local phaseless(I) store	0.014582	0.145817
DOUBLE shared local phaseless(I) load	0.012765	0.127654
shared remote phase ptr store	0.020844	0.208444
shared remote phase ptr load	0.01456	0.145597
shared remote phaseless(1) ptr store	0.013648	0.136483
shared remote phaseless(1) ptr load	0.014561	0.145608
shared remote phaseless(I) ptr store	0.013649	0.136489
shared remote phaseless(I) ptr load	0.014105	0.141053
DOUBLE shared remote phase ptr store	0.014561	0.145612
DOUBLE shared remote phase ptr load	0.012741	0.127408
DOUBLE shared remote phaseless(1) ptr store	0.014559	0.145594
DOUBLE shared remote phaseless(1) ptr load	0.012739	0.12739
DOUBLE shared remote phaseless(I) ptr store	0.014559	0.145592
DOUBLE shared remote phaseless(I) ptr load	0.01274	0.127401

Name: GUPC struct lib
Info: GCCUPC 4.2.3 - struct pointers and NO inline library calls
Time: Thu May 29 17:13:34 PDT 2008
Configuration: conf/gccupc_str_lib.conf
Optimization: -O2 -fno-upc-inline-lib
Threads: 4
MTU run option: -s 1000
MTU Benchamrks
baseline

ARRAY LENGTH:	80000000
NUMBER OF ACCESSES:	10000
NUMBER OF TEST REPEATS:	1000

Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0867	0.0866 (thread 0)	0.0868 (thread 2)	
REMOTE READ LATENCY (msec/double)	0.094	0.0939 (thread 0)	0.0941 (thread 3)	
coalesce				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
NUMBER OF TEST REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0184	0.0183 (thread 1)	0.0184 (thread 3)	
REMOTE READ LATENCY (msec/double)	0.0179	0.0179 (thread 1)	0.0179 (thread 2)	
local				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	100000			
NUMBER OF REPEATS:	1000			
Function	Avg	Min	Max	
pB WRITE (msec/double)	0.0297	0.0297 (thread 3)	0.0297 (thread 0)	
pB READ (msec/double)	0.0287	0.0287 (thread 1)	0.0287 (thread 3)	
p0 WRITE (msec/double)	0.0864	0.0864 (thread 1)	0.0864 (thread 3)	
p0 READ (msec/double)	0.0928	0.0928 (thread 1)	0.0928 (thread 0)	
natural_ring_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Random read:	16168006	0.0124	0.0124	0.0124
Random set:	14469853	0.0142	0.0138	0.0145
stream_strings_relaxed				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2271.4273	0.0176	0.0176	0.0177
memput (local):	2272.4733	0.0245	0.0176	0.0796
memset (local):	5491.3642	0.0073	0.0073	0.0075
memcpy (local):	4538.7374	0.0176	0.0176	0.0177
memget (remote):	2281.6831	0.021	0.0175	0.0486
memput (remote):	2266.5481	0.0198	0.0176	0.0372
memset (remote):	5483.2879	0.0073	0.0073	0.0073
memcpy (remote):	4530.0363	0.0177	0.0177	0.0177
memcpy (shared):	5060.695	0.0158	0.0158	0.0158
stream_strings_strict				
Function	Rate (MB/s)	Avg time	Min time	Max time

memget (local):	2270.0135	0.0176	0.0176	0.0177
memput (local):	2274.8151	0.0176	0.0176	0.018
memset (local):	5482.392	0.0074	0.0073	0.0078
memcpy (local):	4533.3413	0.0178	0.0176	0.0188
memget (remote):	2278.5533	0.0178	0.0176	0.0194
memput (remote):	2268.7243	0.0196	0.0176	0.029
memset (remote):	5480.243	0.0074	0.0073	0.0084
memcpy (remote):	4534.9342	0.0177	0.0176	0.018
memcpy (shared):	5059.7792	0.0162	0.0158	0.019
stream_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Local read:	67276787	0.0037	0.0037	0.0037
Local set:	71003250	0.0035	0.0035	0.0035
Stride-1 read:	65258651	0.0038	0.0038	0.0038
Random read:	17024272	0.0147	0.0147	0.0147
Stride-n read:	39111376	0.0064	0.0064	0.0064
Stride-1 set:	68678019	0.0036	0.0036	0.0036
Random set:	24264729	0.0103	0.0103	0.0103
Stride-n set:	54836105	0.0046	0.0046	0.0046
Stride-1 copy:	73005361	0.0069	0.0068	0.0069
Random copy:	19026619	0.0263	0.0263	0.0263
Stride-n copy:	52472089	0.0095	0.0095	0.0095
vector				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
VECTOR LENGTH:	64			
NUMBER OF TESTS REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0158	0.0158 (thread 2)	0.0158 (thread 0)	
REMOTE READ LATENCY (msec/double)	0.0149	0.0149 (thread 3)	0.0149 (thread 2)	
ptrmath				
iterations 10000000				
Function	us/iter	total (sec)		
local ptr-sum	0.003106	0.031063		
phased ptr-sum	0.023821	0.238211		
phaseless(1) ptr-sum	0.004863	0.048631		

phaseless(I) ptr-sum	0.006771	0.067713
local ptr-sub	0.001365	0.013653
phased ptr-sub	0.004094	0.040943
phaseless(1) ptr-sub	0.00273	0.027295
phaseless(I) ptr-sub	0.00182	0.018198
local ptr-compare	0.00182	0.018197
phased ptr-compare	0.00273	0.0273
phaseless(1) ptr-compare	0.002729	0.027294
phaseless(I) ptr-compare	0.00273	0.027295
phased ptr-to-local	0.012761	0.12761
phaseless(1) ptr-to-local	0.012737	0.12737
phaseless(I) ptr-to-local	0.012738	0.127379
local ptr store	0.001365	0.013647
local ptr load	0.001365	0.013651
shared local phased ptr store	0.011828	0.118281
shared local phased ptr load	0.011373	0.11373
shared local phaseless(1) store	0.011828	0.118276
shared local phaseless(I) load	0.011373	0.113732
DOUBLE local ptr store	0.00182	0.018196
DOUBLE local ptr load	0.001365	0.013651
DOUBLE shared local phased ptr store	0.012285	0.122847
DOUBLE shared local phased ptr load	0.012283	0.122829
DOUBLE shared local phaseless(1) store	0.012283	0.122831
DOUBLE shared local phaseless(1) load	0.012283	0.12283
DOUBLE shared local phaseless(I) store	0.012283	0.122831
DOUBLE shared local phaseless(I) load	0.012283	0.122833
shared remote phase ptr store	0.011832	0.118321
shared remote phase ptr load	0.01183	0.118304
shared remote phaseless(1) ptr store	0.01183	0.118304
shared remote phaseless(1) ptr load	0.01183	0.118299
shared remote phaseless(I) ptr store	0.011831	0.118308
shared remote phaseless(I) ptr load	0.011829	0.118294
DOUBLE shared remote phase ptr store	0.012286	0.12286
DOUBLE shared remote phase ptr load	0.012285	0.122848
DOUBLE shared remote phaseless(1) ptr store	0.012285	0.122854

DOUBLE shared remote phaseless(1) ptr load	0.012285	0.122855
DOUBLE shared remote phaseless(I) ptr store	0.012285	0.12285
DOUBLE shared remote phaseless(I) ptr load	0.012298	0.122984

Name: GUPC lib
Info: GCCUPC 4.2.3 - packed pointers and NO inlined library calls
Time: Thu May 29 17:11:04 PDT 2008
Configuration: conf/gccupc_lib.conf
Optimization: -O2 -fno-upc-inline-lib
Threads: 4
MTU run option: -s 1000
MTU Benchamrks

baseline

ARRAY LENGTH:	80000000		
NUMBER OF ACCESSES:	10000		
NUMBER OF TEST REPEATS:	1000		
Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0976	0.0973 (thread 3)	0.0979 (thread 1)
REMOTE READ LATENCY (msec/double)	0.0878	0.0878 (thread 0)	0.0879 (thread 2)

coalesce

ARRAY LENGTH:	8000000		
NUMBER OF ACCESSES:	1000		
NUMBER OF TEST REPEATS:	1000		
Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.02	0.0198 (thread 2)	0.0201 (thread 0)
REMOTE READ LATENCY (msec/double)	0.0144	0.0144 (thread 0)	0.0145 (thread 2)

local

ARRAY LENGTH:	8000000		
NUMBER OF ACCESSES:	100000		
NUMBER OF REPEATS:	1000		
Function	Avg	Min	Max
pB WRITE (msec/double)	0.0288	0.0288 (thread 3)	0.0288 (thread 0)
pB READ (msec/double)	0.033	0.0330 (thread 0)	0.0330 (thread 2)
p0 WRITE (msec/double)	0.0894	0.0894 (thread 3)	0.0894 (thread 0)

p0 READ (msec/double)	0.057	0.0570 (thread 0)	0.0570 (thread 2)	
natural_ring_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Random read:	18036138	0.0111	0.0111	0.0112
Random set:	15512340	0.0129	0.0129	0.013
stream_strings_relaxed				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2279.8538	0.0176	0.0175	0.0176
memput (local):	2270.9046	0.0176	0.0176	0.0176
memset (local):	5480.243	0.0073	0.0073	0.0073
memcpy (local):	4534.0763	0.0177	0.0176	0.0177
memget (remote):	2266.1501	0.0177	0.0177	0.0177
memput (remote):	2271.0583	0.0176	0.0176	0.0177
memset (remote):	5483.1087	0.0073	0.0073	0.0074
memcpy (remote):	4535.4246	0.0176	0.0176	0.0177
memcpy (shared):	5068.4156	0.0158	0.0158	0.0158
stream_strings_strict				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2278.4296	0.0176	0.0176	0.0176
memput (local):	2270.4129	0.0176	0.0176	0.0176
memset (local):	5480.064	0.0073	0.0073	0.0073
memcpy (local):	4535.1181	0.0176	0.0176	0.0177
memget (remote):	2264.8653	0.0177	0.0177	0.0177
memput (remote):	2271.3043	0.0176	0.0176	0.0176
memset (remote):	5484.0048	0.0073	0.0073	0.0073
memcpy (remote):	4539.29	0.0191	0.0176	0.0312
memcpy (shared):	5067.191	0.0158	0.0158	0.0162
stream_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Local read:	90122561	0.0028	0.0028	0.0028
Local set:	96749954	0.0026	0.0026	0.0026
Stride-1 read:	90809388	0.0028	0.0028	0.0028
Random read:	25186174	0.0099	0.0099	0.0099
Stride-n read:	52533868	0.0048	0.0048	0.0048
Stride-1 set:	92182505	0.0027	0.0027	0.0027
Random set:	31657992	0.0079	0.0079	0.0079

Stride-n set:	66899069	0.0037	0.0037	0.0037
Stride-1 copy:	94589870	0.0053	0.0053	0.0053
Random copy:	24866924	0.0201	0.0201	0.0201
Stride-n copy:	65946102	0.0076	0.0076	0.0076
vector				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
VECTOR LENGTH:	64			
NUMBER OF TESTS REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0119	0.0119 (thread 2)	0.0119 (thread 0)	
REMOTE READ LATENCY (msec/double)	0.011	0.0110 (thread 3)	0.0110 (thread 1)	
ptrmath				
iterations 10000000				
Function	us/iter	total (sec)		
local ptr-sum	0.003153	0.03153		
phased ptr-sum	0.01868	0.186803		
phaseless(1) ptr-sum	0.007013	0.070132		
phaseless(I) ptr-sum	0.004664	0.046643		
local ptr-sub	0.001367	0.013673		
phased ptr-sub	0.003493	0.034933		
phaseless(1) ptr-sub	0.002582	0.02582		
phaseless(I) ptr-sub	0.001822	0.018224		
local ptr-compare	0.001367	0.013673		
phased ptr-compare	0.001367	0.013666		
phaseless(1) ptr-compare	0.001823	0.018232		
phaseless(I) ptr-compare	0.001367	0.013666		
phased ptr-to-local	0.009113	0.09113		
phaseless(1) ptr-to-local	0.009113	0.091134		
phaseless(I) ptr-to-local	0.008658	0.086581		
local ptr store	0.000911	0.009115		
local ptr load	0.001368	0.013675		
shared local phased ptr store	0.008202	0.082016		
shared local phased ptr load	0.007746	0.077456		
shared local phaseless(1) store	0.008202	0.08202		
shared local phaseless(I) load	0.006835	0.068353		

DOUBLE local ptr store	0.001535	0.015345
DOUBLE local ptr load	0.001367	0.013673
DOUBLE shared local phased ptr store	0.007745	0.077452
DOUBLE shared local phased ptr load	0.007516	0.075161
DOUBLE shared local phaseless(1) store	0.007745	0.077454
DOUBLE shared local phaseless(1) load	0.007468	0.074682
DOUBLE shared local phaseless(I) store	0.007745	0.077452
DOUBLE shared local phaseless(I) load	0.00747	0.074696
shared remote phase ptr store	0.008191	0.081907
shared remote phase ptr load	0.007734	0.077339
shared remote phaseless(1) ptr store	0.008189	0.081891
shared remote phaseless(1) ptr load	0.007734	0.077339
shared remote phaseless(I) ptr store	0.008218	0.082179
shared remote phaseless(I) ptr load	0.007735	0.077348
DOUBLE shared remote phase ptr store	0.007734	0.07734
DOUBLE shared remote phase ptr load	0.007302	0.073021
DOUBLE shared remote phaseless(1) ptr store	0.007734	0.077338
DOUBLE shared remote phaseless(1) ptr load	0.007312	0.073119
DOUBLE shared remote phaseless(I) ptr store	0.007734	0.077337
DOUBLE shared remote phaseless(I) ptr load	0.007298	0.072977

Name: GUPC struct
Info: GCCUPC 4.2.3 - struct pointers and inline library calls
Time: Thu May 29 17:12:38 PDT 2008
Configuration: conf/gccupc_str.conf
Optimization: -O2
Threads: 4
MTU run option: -s 1000
MTU Benchamrks
baseline

ARRAY LENGTH:	80000000
NUMBER OF ACCESSES:	10000
NUMBER OF TEST REPEATS:	1000

Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0991	0.0991 (thread 2)	0.0992 (thread 1)	
REMOTE READ LATENCY (msec/double)	0.0845	0.0844 (thread 1)	0.0846 (thread 2)	
coalesce				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
NUMBER OF TEST REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.022	0.0219 (thread 2)	0.0221 (thread 0)	
REMOTE READ LATENCY (msec/double)	0.0148	0.0148 (thread 3)	0.0148 (thread 2)	
local				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	100000			
NUMBER OF REPEATS:	1000			
Function	Avg	Min	Max	
pB WRITE (msec/double)	0.0279	0.0279 (thread 3)	0.0279 (thread 0)	
pB READ (msec/double)	0.0306	0.0306 (thread 0)	0.0306 (thread 3)	
p0 WRITE (msec/double)	0.0881	0.0881 (thread 3)	0.0881 (thread 0)	
p0 READ (msec/double)	0.0628	0.0628 (thread 2)	0.0628 (thread 3)	
natural_ring_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Random read:	18022963	0.0112	0.0111	0.0112
Random set:	15825174	0.0127	0.0126	0.0128
stream_strings_relaxed				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2270.0442	0.0177	0.0176	0.0178
memput (local):	2272.8735	0.0176	0.0176	0.0176
memset (local):	5489.2082	0.0073	0.0073	0.0073
memcpy (local):	4541.3174	0.0176	0.0176	0.0177
memget (remote):	2279.606	0.0176	0.0175	0.0176
memput (remote):	2268.4789	0.0176	0.0176	0.0177
memset (remote):	5483.2879	0.0073	0.0073	0.0073
memcpy (remote):	4543.6542	0.0176	0.0176	0.0176
memcpy (shared):	5063.2914	0.0158	0.0158	0.016
stream_strings_strict				
Function	Rate (MB/s)	Avg time	Min time	Max time

memget (local):	2270.6587	0.0176	0.0176	0.0176
memput (local):	2272.9967	0.0176	0.0176	0.0176
memset (local):	5480.243	0.0073	0.0073	0.0073
memcpy (local):	4537.2645	0.0176	0.0176	0.0176
memget (remote):	2277.9037	0.0191	0.0176	0.0315
memput (remote):	2269.1232	0.0176	0.0176	0.0177
memset (remote):	5480.243	0.0073	0.0073	0.0073
memcpy (remote):	4535.3633	0.0176	0.0176	0.0177
memcpy (shared):	5072.5532	0.0158	0.0158	0.0158

stream_upc

Function	Rate (Refs/s)	Avg time	Min time	Max time
Local read:	73942317	0.0034	0.0034	0.0034
Local set:	91948088	0.0027	0.0027	0.0027
Stride-1 read:	73989275	0.0034	0.0034	0.0034
Random read:	25596251	0.0098	0.0098	0.0098
Stride-n read:	41233818	0.0061	0.0061	0.0061
Stride-1 set:	93041349	0.0027	0.0027	0.0027
Random set:	31541812	0.0079	0.0079	0.0079
Stride-n set:	68435974	0.0037	0.0037	0.0037
Stride-1 copy:	92012636	0.0054	0.0054	0.0054
Random copy:	24223809	0.0206	0.0206	0.0207
Stride-n copy:	63760664	0.0078	0.0078	0.0078

vector

ARRAY LENGTH:	8000000
NUMBER OF ACCESSES:	1000
VECTOR LENGTH:	64
NUMBER OF TESTS REPEATS:	1000

Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0128	0.0128 (thread 2)	0.0128 (thread 0)
REMOTE READ LATENCY (msec/double)	0.0117	0.0117 (thread 0)	0.0117 (thread 2)

ptrmath

iterations 10000000

Function	us/iter	total (sec)
local ptr-sum	0.002568	0.025677
phased ptr-sum	0.021382	0.213824
phaseless(1) ptr-sum	0.004861	0.048612

phaseless(I) ptr-sum	0.006052	0.060517
local ptr-sub	0.00182	0.018197
phased ptr-sub	0.00364	0.036396
phaseless(1) ptr-sub	0.002729	0.027294
phaseless(I) ptr-sub	0.002276	0.022756
local ptr-compare	0.00182	0.018197
phased ptr-compare	0.002729	0.027294
phaseless(1) ptr-compare	0.00273	0.027299
phaseless(I) ptr-compare	0.002729	0.027294
phased ptr-to-local	0.013193	0.131929
phaseless(1) ptr-to-local	0.013193	0.131927
phaseless(I) ptr-to-local	0.013193	0.131926
local ptr store	0.001365	0.013646
local ptr load	0.001365	0.013652
shared local phased ptr store	0.006824	0.068237
shared local phased ptr load	0.006824	0.068239
shared local phaseless(1) store	0.006825	0.068249
shared local phaseless(I) load	0.006824	0.068236
DOUBLE local ptr store	0.001782	0.017818
DOUBLE local ptr load	0.001365	0.013651
DOUBLE shared local phased ptr store	0.006824	0.068236
DOUBLE shared local phased ptr load	0.006824	0.068238
DOUBLE shared local phaseless(1) store	0.006824	0.068242
DOUBLE shared local phaseless(1) load	0.006369	0.063685
DOUBLE shared local phaseless(I) store	0.006824	0.068236
DOUBLE shared local phaseless(I) load	0.006824	0.068243
shared remote phase ptr store	0.006372	0.063718
shared remote phase ptr load	0.006825	0.068254
shared remote phaseless(1) ptr store	0.006371	0.063706
shared remote phaseless(1) ptr load	0.00637	0.0637
shared remote phaseless(I) ptr store	0.00637	0.063703
shared remote phaseless(I) ptr load	0.006825	0.068249
DOUBLE shared remote phase ptr store	0.006825	0.068247
DOUBLE shared remote phase ptr load	0.006825	0.068248
DOUBLE shared remote phaseless(1) ptr store	0.006371	0.063708

DOUBLE shared remote phaseless(1) ptr load	0.006826	0.068264
DOUBLE shared remote phaseless(I) ptr store	0.006371	0.063705
DOUBLE shared remote phaseless(I) ptr load	0.006825	0.068252

Name: GUPC
Info: GCCUPC 4.2.3 - packed pointers and inline library calls
Time: Thu May 29 17:10:17 PDT 2008
Configuration: conf/gccupc.conf
Optimization: -O2
Threads: 4
MTU run option: -s 1000
MTU Benchamrks

baseline			
ARRAY LENGTH:	80000000		
NUMBER OF ACCESSES:	10000		
NUMBER OF TEST REPEATS:	1000		
Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0892	0.0892 (thread 2)	0.0892 (thread 0)
REMOTE READ LATENCY (msec/double)	0.0772	0.0771 (thread 0)	0.0773 (thread 2)
coalesce			
ARRAY LENGTH:	8000000		
NUMBER OF ACCESSES:	1000		
NUMBER OF TEST REPEATS:	1000		
Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0111	0.0110 (thread 1)	0.0112 (thread 3)
REMOTE READ LATENCY (msec/double)	0.0109	0.0109 (thread 0)	0.0110 (thread 3)
local			
ARRAY LENGTH:	8000000		
NUMBER OF ACCESSES:	100000		
NUMBER OF REPEATS:	1000		
Function	Avg	Min	Max
pB WRITE (msec/double)	0.0255	0.0255 (thread 1)	0.0255 (thread 0)
pB READ (msec/double)	0.0263	0.0263 (thread 1)	0.0263 (thread 3)
p0 WRITE (msec/double)	0.0898	0.0898 (thread 3)	0.0899 (thread 0)

p0 READ (msec/double)	0.0556	0.0556 (thread 1)	0.0556 (thread 2)	
natural_ring_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Random read:	17922844	0.0112	0.0112	0.0112
Random set:	26058861	0.0081	0.0077	0.0082
stream_strings_relaxed				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2269.6143	0.0177	0.0176	0.0177
memput (local):	2272.5964	0.0178	0.0176	0.0192
memset (local):	5484.0048	0.0073	0.0073	0.0078
memcpy (local):	4535.9151	0.0245	0.0176	0.0794
memget (remote):	2276.2039	0.021	0.0176	0.0487
memput (remote):	2274.1367	0.0245	0.0176	0.0762
memset (remote):	5478.6324	0.0073	0.0073	0.0073
memcpy (remote):	4527.7135	0.0177	0.0177	0.0177
memcpy (shared):	5069.3345	0.0158	0.0158	0.0158
stream_strings_strict				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2271.0583	0.0176	0.0176	0.0177
memput (local):	2271.3043	0.0176	0.0176	0.0177
memset (local):	5473.449	0.0073	0.0073	0.0074
memcpy (local):	4530.7703	0.0177	0.0177	0.0177
memget (remote):	2275.9569	0.0176	0.0176	0.0176
memput (remote):	2266.3032	0.0177	0.0176	0.0177
memset (remote):	5474.1634	0.0073	0.0073	0.0074
memcpy (remote):	4534.0763	0.0177	0.0176	0.0177
memcpy (shared):	5058.7876	0.0158	0.0158	0.016
stream_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Local read:	110469448	0.0023	0.0023	0.0023
Local set:	120429080	0.0021	0.0021	0.0021
Stride-1 read:	107348075	0.0023	0.0023	0.0023
Random read:	28877640	0.0087	0.0087	0.0087
Stride-n read:	63178647	0.004	0.004	0.004
Stride-1 set:	120139322	0.0021	0.0021	0.0021
Random set:	38820332	0.0064	0.0064	0.0065

Stride-n set:	85270879	0.0029	0.0029	0.0029
Stride-1 copy:	125690860	0.004	0.004	0.004
Random copy:	33948781	0.0147	0.0147	0.0147
Stride-n copy:	96120268	0.0052	0.0052	0.0052

vector

ARRAY LENGTH:	8000000
NUMBER OF ACCESSES:	1000
VECTOR LENGTH:	64
NUMBER OF TESTS REPEATS:	1000

Function	Avg	Min	Max
REMOTE WRITE LATENCY (msec/double)	0.0091	0.0091 (thread 3)	0.0091 (thread 1)
REMOTE READ LATENCY (msec/double)	0.0088	0.0088 (thread 1)	0.0088 (thread 0)

ptrmath
iterations 10000000

Function	us/iter	total (sec)
local ptr-sum	0.00241	0.0241
phased ptr-sum	0.020016	0.200165
phaseless(1) ptr-sum	0.00673	0.067303
phaseless(I) ptr-sum	0.004658	0.046576
local ptr-sub	0.001365	0.01365
phased ptr-sub	0.003639	0.036391
phaseless(1) ptr-sub	0.002729	0.027294
phaseless(I) ptr-sub	0.00182	0.018196
local ptr-compare	0.00182	0.018202
phased ptr-compare	0.00182	0.018197
phaseless(1) ptr-compare	0.00182	0.018198
phaseless(I) ptr-compare	0.00182	0.018197
phased ptr-to-local	0.009098	0.090982
phaseless(1) ptr-to-local	0.016055	0.160546
phaseless(I) ptr-to-local	0.009098	0.090983
local ptr store	0.001365	0.013646
local ptr load	0.001365	0.013651
shared local phased ptr store	0.00364	0.036399
shared local phased ptr load	0.004094	0.040938
shared local phaseless(1) store	0.00364	0.036402
shared local phaseless(I) load	0.003639	0.036392

DOUBLE local ptr store	0.001535	0.015355
DOUBLE local ptr load	0.001365	0.01365
DOUBLE shared local phased ptr store	0.004094	0.040944
DOUBLE shared local phased ptr load	0.004094	0.040937
DOUBLE shared local phaseless(1) store	0.004094	0.040943
DOUBLE shared local phaseless(1) load	0.004094	0.040943
DOUBLE shared local phaseless(I) store	0.004094	0.040943
DOUBLE shared local phaseless(I) load	0.004094	0.040944
shared remote phase ptr store	0.003647	0.036472
shared remote phase ptr load	0.003646	0.036456
shared remote phaseless(1) ptr store	0.003646	0.036459
shared remote phaseless(1) ptr load	0.003645	0.036454
shared remote phaseless(I) ptr store	0.003646	0.036461
shared remote phaseless(I) ptr load	0.003189	0.031894
DOUBLE shared remote phase ptr store	0.004102	0.041021
DOUBLE shared remote phase ptr load	0.004102	0.041015
DOUBLE shared remote phaseless(1) ptr store	0.003646	0.036461
DOUBLE shared remote phaseless(1) ptr load	0.004102	0.04102
DOUBLE shared remote phaseless(I) ptr store	0.004101	0.041014
DOUBLE shared remote phaseless(I) ptr load	0.004101	0.041012

Name: GUPC (O3)
Info: GCCUPC 4.2.3 - packed pointers and inline library calls
Time: Thu May 29 17:11:45 PDT 2008
Configuration: conf/gccupc_O3.conf
Optimization: -O3
Threads: 4
MTU run option: -s 1000
MTU Benchamrks
baseline

ARRAY LENGTH:	80000000
NUMBER OF ACCESSES:	10000
NUMBER OF TEST REPEATS:	1000

Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0808	0.0806 (thread 0)	0.0810 (thread 1)	
REMOTE READ LATENCY (msec/double)	0.0819	0.0819 (thread 3)	0.0820 (thread 2)	
coalesce				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
NUMBER OF TEST REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0128	0.0127 (thread 2)	0.0128 (thread 0)	
REMOTE READ LATENCY (msec/double)	0.011	0.0110 (thread 0)	0.0110 (thread 2)	
local				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	100000			
NUMBER OF REPEATS:	1000			
Function	Avg	Min	Max	
pB WRITE (msec/double)	0.0256	0.0256 (thread 3)	0.0256 (thread 0)	
pB READ (msec/double)	0.0263	0.0263 (thread 3)	0.0263 (thread 0)	
p0 WRITE (msec/double)	0.0899	0.0899 (thread 1)	0.0899 (thread 3)	
p0 READ (msec/double)	0.0557	0.0557 (thread 3)	0.0557 (thread 2)	
natural_ring_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Random read:	17784155	0.0113	0.0112	0.0113
Random set:	25897160	0.0082	0.0077	0.0086
stream_strings_relaxed				
Function	Rate (MB/s)	Avg time	Min time	Max time
memget (local):	2273.1199	0.0176	0.0176	0.0176
memput (local):	2271.0275	0.0176	0.0176	0.0176
memset (local):	5487.0539	0.0073	0.0073	0.0073
memcpy (local):	4538.2463	0.025	0.0176	0.0835
memget (remote):	2267.7123	0.0177	0.0176	0.0178
memput (remote):	2271.581	0.0228	0.0176	0.0647
memset (remote):	5477.2015	0.0073	0.0073	0.0073
memcpy (remote):	4534.3827	0.0177	0.0176	0.0177
memcpy (shared):	5073.5503	0.0158	0.0158	0.0158
stream_strings_strict				
Function	Rate (MB/s)	Avg time	Min time	Max time

memget (local):	2275.8334	0.0211	0.0176	0.0488
memput (local):	2270.0135	0.0176	0.0176	0.0176
memset (local):	5475.7714	0.0094	0.0073	0.0258
memcpy (local):	4534.873	0.0177	0.0176	0.0177
memget (remote):	2266.6706	0.0177	0.0176	0.0177
memput (remote):	2269.3687	0.0237	0.0176	0.0721
memset (remote):	5470.5935	0.0074	0.0073	0.0079
memcpy (remote):	4530.5256	0.0246	0.0177	0.0796
memcpy (shared):	5061.382	0.016	0.0158	0.0173
stream_upc				
Function	Rate (Refs/s)	Avg time	Min time	Max time
Local read:	110271953	0.0023	0.0023	0.0023
Local set:	120484431	0.0021	0.0021	0.0021
Stride-1 read:	107667728	0.0023	0.0023	0.0023
Random read:	28908690	0.0087	0.0086	0.0087
Stride-n read:	62924628	0.004	0.004	0.004
Stride-1 set:	120125558	0.0021	0.0021	0.0021
Random set:	39074939	0.0064	0.0064	0.0064
Stride-n set:	85409791	0.0029	0.0029	0.0029
Stride-1 copy:	135501195	0.0037	0.0037	0.0037
Random copy:	34097260	0.0147	0.0147	0.0147
Stride-n copy:	96429649	0.0052	0.0052	0.0052
vector				
ARRAY LENGTH:	8000000			
NUMBER OF ACCESSES:	1000			
VECTOR LENGTH:	64			
NUMBER OF TESTS REPEATS:	1000			
Function	Avg	Min	Max	
REMOTE WRITE LATENCY (msec/double)	0.0092	0.0092 (thread 2)	0.0092 (thread 0)	
REMOTE READ LATENCY (msec/double)	0.0094	0.0094 (thread 3)	0.0094 (thread 0)	
ptrmath				
iterations 10000000				
Function	us/iter	total (sec)		
local ptr-sum	0.003152	0.031521		
phased ptr-sum	0.018683	0.186831		
phaseless(1) ptr-sum	0.00707	0.070699		

phaseless(I) ptr-sum	0.004851	0.048511
local ptr-sub	0.001367	0.013673
phased ptr-sub	0.003645	0.03645
phaseless(1) ptr-sub	0.002734	0.027337
phaseless(I) ptr-sub	0.001823	0.018232
local ptr-compare	0.001822	0.018225
phased ptr-compare	0.001823	0.018226
phaseless(1) ptr-compare	0.001823	0.018226
phaseless(I) ptr-compare	0.001823	0.018234
phased ptr-to-local	0.009114	0.091136
phaseless(1) ptr-to-local	0.009568	0.095681
phaseless(I) ptr-to-local	0.009113	0.091134
local ptr store	0.001367	0.013666
local ptr load	0.001367	0.013674
shared local phased ptr store	0.003645	0.036452
shared local phased ptr load	0.004101	0.041012
shared local phaseless(1) store	0.004101	0.041009
shared local phaseless(I) load	0.004101	0.041009
DOUBLE local ptr store	0.001537	0.015369
DOUBLE local ptr load	0.001367	0.013674
DOUBLE shared local phased ptr store	0.003645	0.036452
DOUBLE shared local phased ptr load	0.004101	0.041011
DOUBLE shared local phaseless(1) store	0.004102	0.041017
DOUBLE shared local phaseless(1) load	0.004101	0.041009
DOUBLE shared local phaseless(I) store	0.004101	0.04101
DOUBLE shared local phaseless(I) load	0.004101	0.04101
shared remote phase ptr store	0.003186	0.03186
shared remote phase ptr load	0.003185	0.031849
shared remote phaseless(1) ptr store	0.002275	0.022749
shared remote phaseless(1) ptr load	0.003185	0.031848
shared remote phaseless(I) ptr store	0.00273	0.027297
shared remote phaseless(I) ptr load	0.003185	0.031848
DOUBLE shared remote phase ptr store	0.00364	0.036399
DOUBLE shared remote phase ptr load	0.004095	0.040946
DOUBLE shared remote phaseless(1) ptr store	0.004095	0.040948

DOUBLE shared remote phaseless(1) ptr load	0.004095	0.040947
DOUBLE shared remote phaseless(I) ptr store	0.004095	0.040945
DOUBLE shared remote phaseless(I) ptr load	0.004095	0.040948

Benchmarks Configurations

Name: GUPC 4.0
Info: GCCUPC 4.0.3.5 with optimization
Time: Thu May 29 17:14:23 PDT 2008
Configuration: conf/gccupc4-0.conf
Optimization: -O2
Threads: 4
MTU run option: -s 1000

Name: GUPC struct lib
Info: GCCUPC 4.2.3 - struct pointers and NO inline library calls
Time: Thu May 29 17:13:34 PDT 2008
Configuration: conf/gccupc_str_lib.conf
Optimization: -O2 -fno-upc-inline-lib
Threads: 4
MTU run option: -s 1000

Name: GUPC lib
Info: GCCUPC 4.2.3 - packed pointers and NOT inlined library calls
Time: Thu May 29 17:11:04 PDT 2008
Configuration: conf/gccupc_lib.conf
Optimization: -O2 -fno-upc-inline-lib
Threads: 4
MTU run option: -s 1000

Name: GUPC struct
Info: GCCUPC 4.2.3 - struct pointers and inline library calls
Time: Thu May 29 17:12:38 PDT 2008
Configuration: conf/gccupc_str.conf
Optimization: -O2
Threads: 4
MTU run option: -s 1000

Name: GUPC
Info: GCCUPC 4.2.3 - packed pointers and inline library calls
Time: Thu May 29 17:10:17 PDT 2008
Configuration: conf/gccupc.conf
Optimization: -O2
Threads: 4
MTU run option: -s 1000

Name: GUPC (O3)
Info: GCCUPC 4.2.3 - packed pointers and inline library calls
Time: Thu May 29 17:11:45 PDT 2008
Configuration: conf/gccupc_O3.conf
Optimization: -O3
Threads: 4
MTU run option: -s 1000

-- end --