PRELIMINARY Assignment 4: Recursive DGEMM Blocking

Due: Monday 02/14/05, Multiplication Factor: 2.0

In this assignment, you should implement a recursively blocked DGEMM, as discussed in class. To get the files you need, perform:

```
cp ~whaley/teach/cis5930/ASG4/* .
make archdirs
```

The only file you are allowed to modify is `dgemm_rec.c`, which is where you should implement your recursive matmul. You will send a completed `dgemm_rec.c` to `whaley@cs.fsu.edu` by 10AM by the due date.

Your first order of business is to get correct answers in a purely recursive scheme (recur down to 1). Then, try to increase performance by finding better stopping criteria. You will be graded on both correctness and relative (to best code) performance.

You should write your recursive gemm so that changing the macro `STOP_NB` varies where the recursion is stopped. If the largest DGEMM dimension is `STOP_NB` or less, the recursion is ended, and the kernel is called. The kernel routines (for doing the unblocked DGEMM) can be found in `dgemm.c`. Note there are specialized PIII and Sparc kernels that can be called in certain cases, and a generic (unoptimized) code that can be used when they are not. Examine the Makefile for the build of this kernel, and note that we use CPP macros to vary α and β behavior. Why?

Your routine should get the correct answer, regardless of the setting of α and β. To test a bunch of differing scalars with a square problem of size 300, issue:

```
./xdgemmtst -a 3 0 1 2.1 -b 3 0 1 0.8 -n 300
```

With your recursive blocking, your performance should hold up even as problem size is increased. To run problems in the range [20,200:20], with extra reps for accuracy, issue:

```
./xdgemmtst -F 20 -N 20 200 20
```

You must also produce correct answers (and decent performance) for non-square matrices. To test a gemm with M=50, N=20, K=77, issue:

```
./xdgemmtst -m 50 -n 20 -k 77
```

With no arguments, square problems are run between [100,1000:100]. You can get some usage info by issueing: `xdgemmtst --help`.