

We now have four benchmarks for our processor performance. All are run the same way. After you load the binary into flash, then load your processor into the fpga, you will:

(1) Press key[3] to copy the contents of Flash into SDRAM. When that completes, red LED 0 will light.

(2) Press key[0] to run the program out of SDRAM.

Your overall performance will be judged by average (arithmetic) speedup over the baseline performance for the four applications. I have included not just the hex files, but also the .c and disassembly for each program to help you in your analysis and debugging.

Until some of you complete your service projects ;-} , our primary output are the PASS, FAIL, DONE primitives we wedged into the MIPS ISA. The results can be seen in the rightmost three 7-segment LEDs.

The four programs and their expected behavior are:

Nqueens – This is the 8-queens-on-a-chessboard puzzle. It increments a counter (and passes it to the PASS primitive) every time it finds a solution, then finally completes with DONE(counter), resulting in a display of “d60”.

Qsort – This is a simple, optimized quicksort. It sorts 20000 integers. When it completes successfully, it executes DONE(12<sup>th</sup> entry), resulting in a display of “d29”. This is your largest dataset.

Esift – This is the classic sieve of erasthenes, which finds all primes below a certain limit, by simply crossing out multiples of lower primes. The dataset is pretty small, because it uses a bitmap. It calls DONE(1) when it completes, if correct.

Coin – This is a puzzle involving coin flipping. It has a pretty small dataset. It solves it for 11 different input values, and calls PASS(n), where n is 1,2,3...11 (OB hex). Then finishes with DONE(11). So you’ll see “d0B”.