An FPGA-based Conjugate Gradient Implementation for Dense Matrices

Antonio Roldao Lopes and George A. Constantinides
{ aroldao, g.constantinides }@ic.ac.uk

Electrical & Electronic Engineering Department,
Imperial College London,
London SW7 2AZ, U.K.

As Field Programmable Gate Arrays (FPGAs) have reached capacities beyond millions of equivalent gates, it becomes possible to accelerate floating point scientific computing applications [1]. One type of calculation that is commonplace in scientific computation is the solution of systems of linear equations. A method that has proven in software to be very efficient and robust for finding such solutions is the Conjugate Gradient (CG) algorithm. CG is attractive for FPGA-based computation, even for small dense matrices, due to the relatively small number of division operations required. In this abstract we present a highly optimized parallel and deeply-pipelined hardware CG implementation as depicted in Figure 1 [2], particularly suited for accelerating multiple small to medium sized dense systems of linear equations.

.

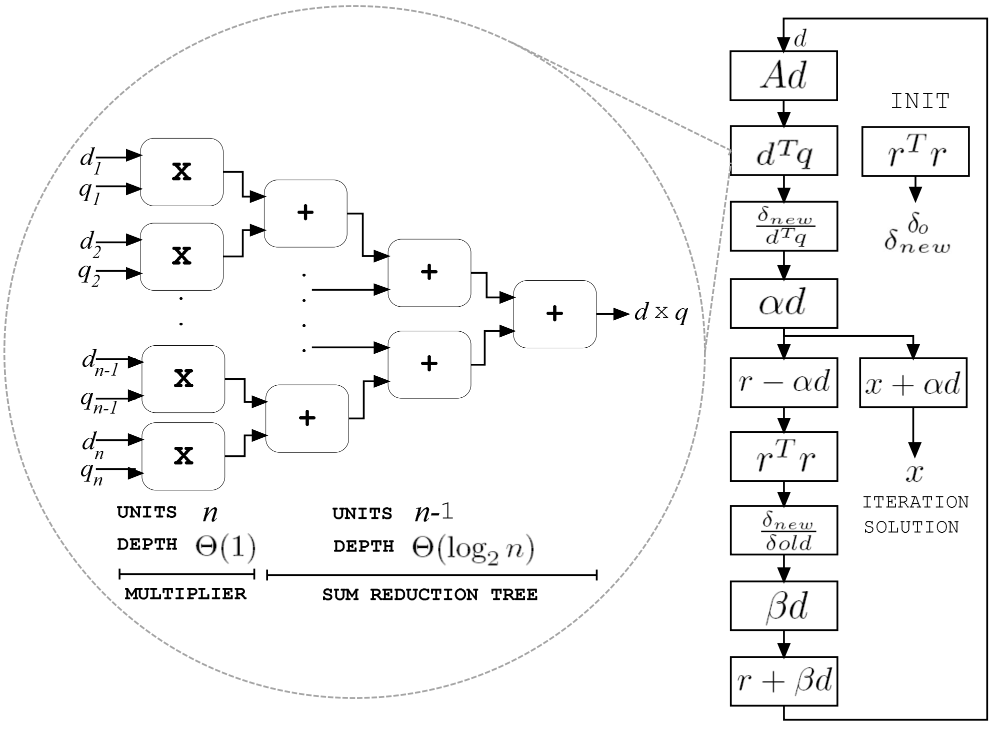

Figure 1 - Dataflow diagram displaying a parallel vector by vector multiplication circuit.

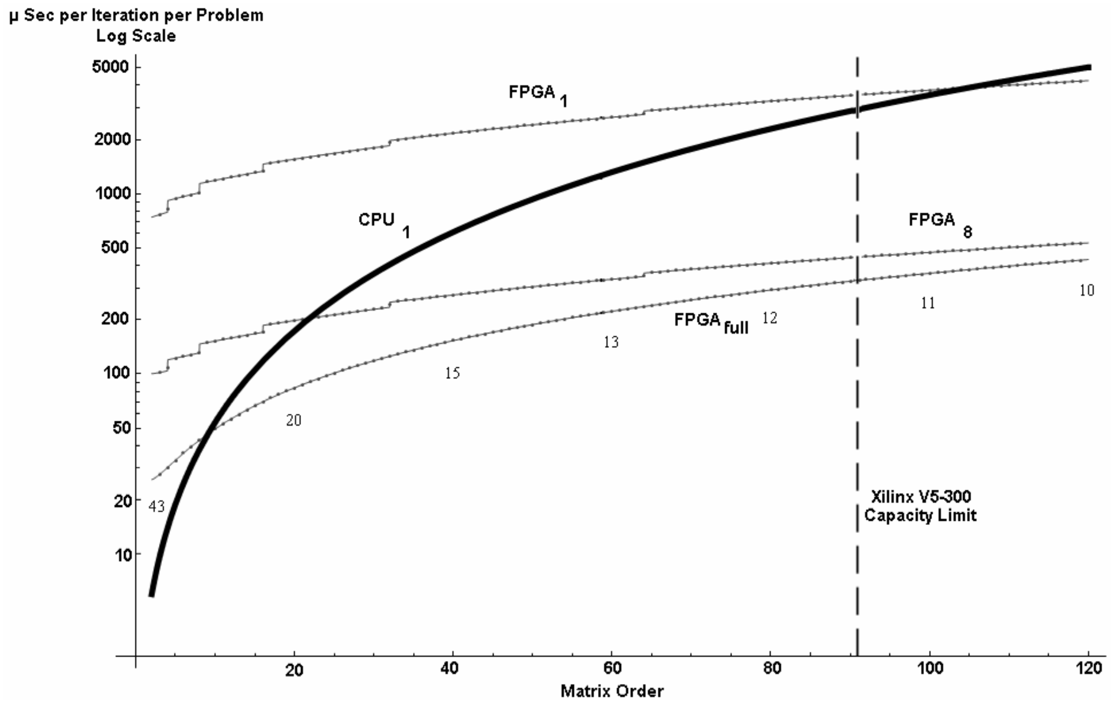

Figure 2 - Performance - Pentium IV at 3GHz CPU vs Virtex V5-330 FPGA. For the FPGA, there are three curves, corresponding to 1 problem in the pipeline, 8 problems in the pipeline, and a full pipeline. For the final case, the numbers under the FPGA curve indicate the number of problems required to fill the pipeline, which vary with matrix order.

Through parallelization it is possible to convert the computation time per iteration for an order n matrix from $O(n^2)$ cycles, for a software implementation, to $O(n)$. Due to the iterative nature of this method and a block based approach implementation, the I/O requirements of this architecture are scalable and converge to a constant value with the increase of matrix order.

Results on a VirtexII-6000 demonstrate sustained performance of 7.5 GFLOPS for dense matrices up to order 28 and projected results on a Virtex5-330 indicate sustained performance of 41 GFLOPS for dense matrices up to order 92. The former implementation operates at 119 MHz and its performance is comparable to high-end CPUs, whereas the latter operates at 215MHz and represents a significant speedup.

Depicted in Figure 2 is a comparison between the Pentium IV 3GHz peak theoretical performance and a Virtex V5-330 FPGA. In this figure it is possible to observe that the FPGA, with a fully loaded pipeline, becomes faster then the CPU for matrix orders above 10. For a single problem on the FPGA pipeline, the CPU becomes slower for matrix orders above 105.

Future work will be focused on optimizing the performance by exploring different number representations, and by exploiting the structure present in systems resulting from applications in control system synthesis.

[1] K. Underwood, "FPGAs vs. CPUs: Trends in Peak Floating-Point Performance", in *Proc. ACM Int. Symp. on Field-Programmable Gate Arrays*, 2004, pp. 171-180.

[2] A. Roldao Lopes and G. A. Constantinides, "A High Throughput FPGA-based Floating Point Conjugate Gradient Implementation", to appear in *Proc. Applied Reconfigurable Computing*, 2008.