

Reconfigurable Computing Cluster Project: Phase I Brief

Andrew G. Schmidt, William V. Kritikos, Siddhartha Datta, Ron Sass
University of North Carolina at Charlotte
9201 University City Blvd. / Charlotte, NC 28223-0001
{andrewgschmidt,will.kritkos}@gmail.com,
{skdatta,rsass}@uncc.edu

1. Introduction

The Reconfigurable Computing Cluster (RCC) Project (previously described in [3]) is investigating the feasibility of using Platform FPGAs to build cost-effective High-Performance Computing (HPC) systems that will scale to a PetaFLOP. Towards that goal the first prototype, named *Spirit*, consisting of 64 Xilinx ML-410 development boards has been assembled at The University of North Carolina at Charlotte and is currently in operation. This short paper aims to describe a number of recent developments and report some key performance results. Specifically, (a) additional functionality added to the custom networking board that was not originally anticipated (or reported), (b) *Spirit's* programming model for computational scientists, and (c) the performance of two compute accelerators (an FPU and e^{-x}) that have been implemented. None of these results have previously been discussed. An additional educational role for the cluster has also emerged since the last report. FPGA Session Control (FSC) [2] is the software to support this new role.

2. Network Functionality

Spirit's original design called for three networks connecting Xilinx ML-310 developer boards. The three networks included: a custom high-speed network data communication, a Gigabit Ethernet network for administration (and comparisons), and a USB network (with RS232-to-USB converters) to aggregate all 64 nodes' serial consoles on the head node. Two important aspects of this design has changed. First, the node FPGAs have been upgraded to Virtex-4's (using Xilinx ML-410 development boards). Second, the RS232-to-USB functionality has been directly implemented on the custom network board. In addition a serial interface, this gives every node built-in JTAG and diagnostic capabilities.

The direct point-to-point network is in the final stages of development. The previously reported Bit Error Ratio (BER) results [3] were completed on ML-310 development boards. Further tests [1] found that bit rates of 3.125 Gbps were achievable with no bit errors for Serial ATA cables up to 7.5 meters in length. However, 10-meter cables were only able to support 2.5 Gbps with no bit errors.

The BER tests have been repeated on the ML-410s. Because

the ML-410 uses a different reference clock than the ML-310 (250 MHz MGT versus 156.25 MHz), the testable bit rates on the ML-410 are limited to 2.0, 2.5 and 4.0 Gbps. As expected — up to 10 meter cables can sustain 2.5 Gbps with no bit errors. However, no cable length is able to sustain 4.0 Gbps (with no bit errors). The best achievable BER for 1 meter cables, the most common length needed, is 1×10^{-9} at 4.0 Gbps which is unacceptable for the link layer. These results are not altogether surprising. SATA cables (Amphenol SpectraStrip) were not engineered to operate above 3 Gbps.

The third version of our custom network adapter board will be the one installed in all 64 nodes has evolved considerably. Previously, it was a passive board that simply routes the signals from the FPGA's MGT to SATA connectors through the ML-410's personality module interface. This version of the board will greatly reduce the trace lengths and FR4 losses which may facilitate higher achievable bit rates. Its physical dimensions will be adjusted to fit the physical constraints of the 1U ML-X10 chassis used in the cluster. Perhaps most importantly, this new board will also include active components that will (i) eliminate the need for an RS232-to-USB converter, (ii) add a USB-JTAG interface, (iii) provide digital I/O to USB. The need for this functionality became apparent over the last year working with the cluster. For example, it enables users to use Integrated Logic Analyzers (such as ChipScope) on any arbitrary node of the cluster and the digital I/Os allow remotes users to "see" status LEDs over the network. It also greatly enhances the ability to use the cluster in an educational setting.

3. Programming Model

The raw computational capacity of an FPGA is unrivaled in general-purpose devices. Most often, the greatest challenge to achieving high performance is not the number of function units but rather keeping all of the potential function units busy — that is, managing bandwidth in the design. Hardware engineers routinely take datapaths and timing into account but this is a foreign concept to the domain specialists (computational scientists) that will use *Spirit*. So a fundamental question for the RCC Project revolves around the programming model. What, if any, programming model will unlock *Spirit's* potential?

A High-Level Language (HLL) that is automatically compiled into a hardware design would be ideal. However, many of the commercial and research tools available today fall far short of our expectations. Current tools either fail to discover the parallelism or substantially change the programming model in ways that may make sense to computer scientists but not to the ordinary domain specialist.

Our approach is multi-disciplinary: by teaming computer engineers with domain specialists, the goal is to keep each stakeholder engaged in their area of expertise. Based on past collaborations the exploratory and evolutionary nature of science manifests itself in two modes. (1) the domain specialist is continually developing and improving their code. (2) the application is relatively stable and the discovery process is at a higher level.

In the first case, the computer engineers on the team can develop base systems. As the domain specialists on the team react to discoveries, computer engineers improve the architecture. Moreover, as computer engineers improve the architecture, they provide feedback to domain specialists which can, in turn, influence their algorithms. In this way, the application and the architecture co-evolve.

The second case is more like a traditional custom-computing machine architecture where the computer engineer produces a specialized architecture for the application. However, the RCC Project takes this one step further. By employing coarse grain reconfiguration, an adaptable architecture is being developed. The first dimension of configurability that we are looking at is latency (few queries on large subject databases) versus throughput (large number of queries). Before synthesis, the domain specialist can choose their preference. Then — based on the primary (RAM) and secondary (disk) memory bandwidths, programmable logic resources available, and the domain specialist’s goals — an architecture is generated to match the desired characteristics.

4. Accelerators

At the outset of the project it was anticipated that conventional Platform FPGA cores (buses, bridges, memory controllers etc.) would not support the bandwidth needed to scale to a PetaFLOP. Indeed, we have verified this is the case and it provides the motivation for developing new mechanisms for keeping application-specific accelerators fully utilized.

During initial tests it became apparent that any accelerator needs to have the ability to not only process data provided by the processor, but also use DMA to add the ability of accessing data directly to the hardware core. There also is a need to either interrupt the processor to read the results or to write the results back to main memory. While this functionality is the basis for commodity computing, FPGAs have the significant advantage of being customizable. As a result buffer sizes and transaction types can be more finely tuned per application than for an ASIC design.

One accelerator test was to determine the utilization percent-

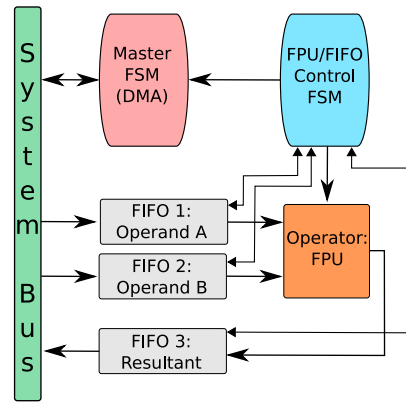


Figure 1. Custom Hardware Core with Direct Memory Access

age of a Floating Point Unit (FPU). Figure 1 shows the block diagram of the Buffered DMA FPU hardware core design. Two input FIFOs store up to 1024 operands for the floating point operation. The FPU/FIFO Control FSM is responsible for triggering the Master FSM to perform a read from memory and writing data from the resultant fifo back to main memory (DMA). Non-DMA transfers utilize an interrupt to notify the processor to either supply additional data or that data is available to be read back. Adjusting the FIFO depths and interrupt triggers provides important data to analyze and determine the feasibility of using this model. Likewise, an e^{-x} function unit has been developed and just as with the FPU maximizing its utilization requires intelligent interfacing designs.

5. Conclusion

The goal of this interim report was to share answers to several questions that have been raised by the community since the project was introduced. The new data presented here suggests that the original goals are achievable. Further, based on our initial experience with the cluster, new features have been introduced offering additional functionality.

References

- [1] W. V. Kritikos. Feasibility of serial ata cables for the physical link in high performance computing clusters. Master’s thesis, University of Kansas, May 2007.
- [2] Y. Rajasekhar, Y. Phatak, A. G. Schmidt, W. V. Kritikos, and R. Sass. Fpga session control (fsc): Providing remote access to a cluster of fpgas. In *To Appear: Proceedings of the 16th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM’08)*. IEEE Computer Society, 2008.
- [3] R. Sass, W. V. Kritikos, A. G. Schmidt, S. Beeravolu, P. Beeraka, K. Datta, D. Andrews, R. S. Miller, and J. Daniel Stanzione. Reconfigurable computing cluster (rcc) project: Investigating the feasibility of fpga-based petascale computing. In *Proceedings of the 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM’07)*, pages 127–140. IEEE Computer Society, 2007.