DySER porting on Virtex-7

Ziliang Guo Mona Jalal Zuyu Zhang
Project Report
CS/ECE 752 Advanced Computer Architecture I
The University of Wisconsin-Madison
ziliang@cs.wisc.edu, jalal@wisc.edu, zuyu@cs.wisc.edu

Abstract

The original implementation of integrating DySER with a OpenSPARC processor was done using the Virtex-5 FPGA platform. That effort faced a major limitation wherein the Virtex-5 platform did not have enough area to hold the entire OpenSPARC processor along with a full DySER block with its associated functional units (FU) and interconnects. As a result, benchmarks and studies had to rely on fixed configurations of DySER FUs instead of the dynamically routeable configuration that a full block would offer. The Virtex-7 platform, however, provides approximately eight times the logical cells of Virtex-5, which we believe will be sufficient to allow a full DySER block to be synthesized. Challenges to this effort include the need to completely replace the peripheral modules that used to provide the OpenSPARC platform with access to main memory and connectivity, such as Ethernet and the serial port, with versions that are supported on the Virtex-7 platform.

1. Introduction

With the increasing density of modern semiconductor fabrication techniques, there has been a drive to find better ways to make use of all the silicon area available. A significant challenge is how to continue improving performance while keeping power usage reasonable or perhaps even decreasing it. One strategy that has received significant attention is the use of accelerator co-processors such as GreenDroid [2], Navigo [5], and DySER [3, 4]. These co-processors are used to perform computation on specialized hardware, which in theory allows for better performance and lower power usage due to their lack of support for general purpose computation.

In this paper we focus on implementing the OpenSplyser processor on the Virtex-7 FPGA VC707 evaluation board [8]. OpenSplyser is an OpenSPARC processor with a DySER core integrated into its execution pipeline. The OpenSPARC processor is the open source version of the UltraSPARC T1 released by Sun Microsystems. The UltraSPARC T1 is a multi-core design wherein each core is capable of supporting four threads concurrently. This ability is ignored in the OpenSplyser design. DySER includes an array of heterogeneous functional units connected with a routable interconnect fabric. This interconnect allows a DySER core to be dynamically configured, with the functional units aligned into custom execution pipelines depending on application workload. A compiler for generating code to run on an OpenSplyser processor [1] also exists, though it requires a developer manually specify blocks of data that are to be fed into a DySER block.

The Virtex-7 FPGA has approximately two million logic cells, over eight times that of the Virtex-5 FPGA originally used to prototype the OpenSplyser design. A key limitation to using the Virtex-5 FPGA was the inability to incorporate a full DySER core in the OpenSplyser design due to area limitations. As a result, evaluation of DySER was done by creating fixed execution pipelines of DySER functional units and interconnects points called HardDysers and incorporating this fixed design into the OpenSPARC processor. This fixed pipeline design needed to be generated for every application or benchmark one would want to run on OpenSplyser, making the testing process extremely tedious.

The rest of the paper is organized as follows. Section 2 describes our proposed design for FPGA porting. Section 3 provides our methodology for porting and section 4 presents our recent porting results. Section 5 discusses related work and Section 6 concludes.

2. Design

OpenSplyser is OpenSPARC T1 processor [6] with a scalable DySER core integrated into its execution pipeline, shown in Figure 1. The original Virtex-5 based OpenSplyser design used several peripheral modules to provide Ethernet, main memory, the serial port support, and the built in DDR2 module for main memory. Ethernet and the se-

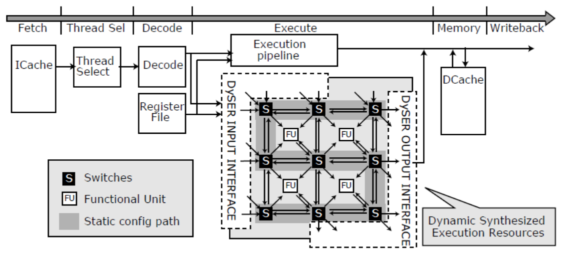

Figure 1. DySER integrated in OpenSPARC

rial port were tied into an interrupt system connected to a Microblaze embedded processor provided by Xilinx for use on its FPGA products. The OpenSPARC processor also relied upon the Microblaze processor to act as the main memory controller. Communication between the OpenSPARC's cache and main memory was handled by a crossbar that bridged the OpenSPARC processor and the Microblaze processor using Fast Simplex Links (FSL). Unfortunately, all of the peripherals had to be replaced when porting to the Virtex-7 platform, as Xilinx had dropped support for the older modules in favor of newer designs utilizing the AXI interface.

The following lists the Xilinx IP modules used and the hardware that map to on the VC707 evaluation board.

- AXI Ethernet Embedded IP module using a SGMII interface mapped to the Marvell Alaska Tri-Speed PHY device.
- AXI UARTLite mapped to the Silicon Labs CP2103GM USB to UART bridge device.
- AXI DDR3 module mapped to the Micron MT8JTF1286HZ-1G6G1 1GB DDR3 SODIMM.

Communication with the Ethernet and the serial port modules were handled with AXI interconnect and interrupt controller modules that were tied into the Microblaze processor. Pin assignments for the Virtex-7 FPGA were done based off of the VC707 evaluation board data sheet [7] for all of the above components.

3. Methodology

The project effectively includes three steps. The first is to create a working Virtex-7 project incorporating OpenSPARC and to ensure this system works by itself. In addition to verifying the basic diagnostic programs bundled with the OpenSPARC package released by Sun works, the ultimate test would be to see if the Ubuntu Linux distro compiled for the SPARC architecture is able to boot and function.

Once the basic Virtex-7 OpenSPARC design is verified to work, the original work involving the HardDyser designs should be replicated on the Virtex-7 as a sanity check. These can act both as a comparison to the original Virtex-5 experiments but also as a reference for benchmark results for when the full DySER core is integrated.

The final step would be to integrate a full DySER core into the Virtex-7 OpenSPARC design and rerun the benchmarks firstly to confirm correct output and secondly to see if the performance characteristics demonstrated by the HardDyser designs match that of a full DySER core. This is especially important as there may be subtle influences on performance when computations need to traverse the entire DySER core versus running through the fixed HardDyser pipeline.

DySER in its current iteration is most easily configured to run programs involving parallel but independent computations. As such, the example programs prepared to test against it are mostly parallel matrix operations whose inputs and outputs can be loaded in parallel into and out of DySER.

4. Results

Two designs were ultimately created, one to prototype basic functionality of the AXI peripherals attached to a Microblaze processor and another incorporating the OpenSPARC processor as well as connecting the Microblaze to the cache crossbar. Attempts to debug the baseline design with Xilinx's tools proved problematic as they were unable to properly access hardware breakpoints that were supposed to have been incorporated into the design. Test programs constantly stalled on the processor and attempts to retrieve outputs from the serial interface also failed. The cause is still under investigation, as Xilinx's documentation is not very clear on how the serial module should be configured in order to communicate with a workstation.

Figure 2 shows the layout of IP cores using Xilinx EDK 14.3.

Attempts to validate the OpenSPARC design with the crossbar diagnostic program supplied by Sun failed outright, with the cause still under investigation. As the crossbar control and diagnostic program source codes are little more than arrays of hex values, it would probably be easier to attempt to reverse engineer the compiled binaries than try to interpret the source code. One potentially significant complication is the use of the fast simplex links to connect the Microblaze with the crossbars, which could cause conflicts with the newer AXI interconnects. A timing conflict was also reported by Xilinxs tools, though the cause appeared to be from clocks generated by the tools. This could be because in the original Virtex-5 design, the entire system was clocked at a different frequency and in the porting effort the clocks controlling the fast simplex link and cross-

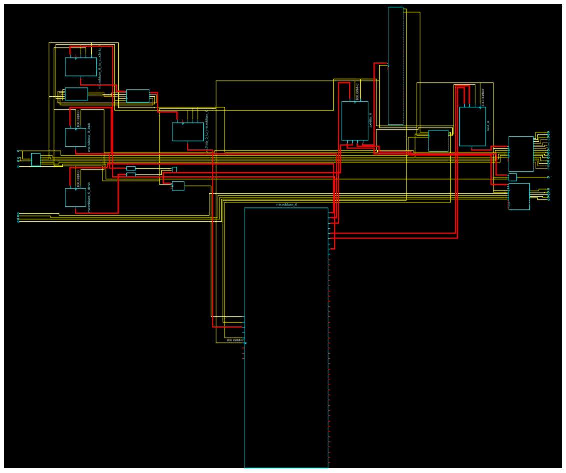

Figure 2. **IP cores layout using Xilinx EDK 14.3**

bar dealing with the cache and main memory were set to match the Virtex-5 design clocks.

5. Related work

Silicon scaling and its resulting utilization wall have driven researchers to consider new methods for improving performance and energy efficiency. Other approaches also exist to try and harness the increasing amounts of area afforded by modern manufacturing methods. One of such alternative methods is Conservation Cores (C-Cores), which place a heavier emphasis on energy efficiency. These C-Cores however are highly specialized, as their execution pipelines are generated from analysis of a specific application they are intended to support. As such, they are more akin to the HardDyser designs that the Virtex-5 FPGA was limited to than the highly dynamic full DySER core.

6. Conclusions

While the project was unable to achieve its goal of implementing an OpenSPARC processor with a full DySER module on the Virtex-7 platform, the team did learn a great deal about the Xilinx FPGA infrastructure. At present, the team is working on verifying that the baseline OpenSPARC processor with the new AXI peripherals works on the Virtex-7 evaluation board. This would require determining what the proper configuration for the UART module should be, as well as potentially reverse engineering and rewriting the crossbar control code to work with the newer Microblaze processor.

Further evaluation also needs to be done on the timing constraints the Xilinx tools warned about, again with respect to the fast simplex links and the crossbar. As the crossbar was provided as part of the OpenSPARC package, attempting to update it to use the newer connections on the Virtex-7 would be extremely difficult if not outright infeasible.

If the updated OpenSPARC project can be confirmed to work by itself on the Virtex-7 FPGA, the next step would be to incorporate a full DySER core and attempt to evaluate its performance characteristics.

Acknowledgments

The authors would like to thank Zachary Marzec for DySER tutorial, Chen-han Ho for FPGA part, and Prof. Karu Sankaralingam for his guidance and resource support.

References

[1] J. Benson, R. Cofell, C. Frericks, C.-H. Ho, V. Govindaraju, T. Nowatzki, and K. Sankaralingam. Design, integration and implementation of the dyser hardware accelerator into opensparc. In *HPCA*, pages 115–126, 2012.

[2] N. Goulding-Hotta, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, P. Huang, M. Arora, S. Nath, V. Bhatt, J. Babb, S. Swanson, and M. Taylor. The greendroid mobile application processor: An architecture for silicon's dark future. *Micro, IEEE*, 31(2):86 –95, march-april 2011.

[3] V. Govindaraju, C. Ho, T. Nowatzki, J. Chhugani, N. Satish, K. Sankaralingam, and C. Kim. Dyser: Unifying functionality and parallelism specialization for energy efficient computing. *Micro, IEEE*, PP(99):1, 2012.

[4] V. Govindaraju, C.-H. Ho, and K. Sankaralingam. Dynamically specialized datapaths for energy efficient computing. In *High Performance Computer Architecture (HPCA), 2011 IEEE 17th International Symposium on*, pages 503 –514, feb. 2011.

[5] M. Hempstead, G.-Y. Wei, and D. Brooksand. Navigo: An earlystage model to study power-contrained architectures and specialization. In *5th Workshop on: Modeling, Benchmarking and Simulation (MoBS)*, Austin, Texas, United States, 2009.

[6] Opensparc t1 processor design and verification user's guide.

[7] VC707 Evaluation Board for the Virtex-7 FPGA User Guide.

[8] Xilinx 7 series fpgas overview.