The Transmogrifier-4: An FPGA-Based Hardware Development System with Multi-Gigabyte Memory Capacity And High Host and Memory Bandwidth

Joshua Fender, Jonathan Rose, David Galloway
The Edward S. Rogers Sr. Department of
Electrical and Computer Engineering
University of Toronto
{fender, jayar, drg}@eecg.utoronto.ca

Abstract

FPGA-based hardware development systems are extremely useful for exploring exciting applications in vision, graphics, and many other computationally intensive problems. Our experience with previous systems has shown that their memory capacity, inter-FPGA bandwidth, host-to-FPGA bandwidth, and memory bandwidth are all critical to the successful implementation of high performance systems. This paper presents the design, and implementation, of a new FPGA-based development system that was created with the goal of providing as much performance in these four areas as feasible. The design consists of 4 Altera Stratix FPGAs, providing a total of 316,160 four-input LUTS and 29.5Mb of on chip ram. This is supplemented with 8GB of external memory. The system has a measured 17.6GB/s of total aggregate memory bandwidth, and 154MB/s (read) and 266MB/s (write) measured host-to-FPGA bandwidth. This paper describes the motivating applications, major design issues, and performance of the Transmogrifier-4 field-programmable system.

1. Introduction

An FPGA-based rapid development system is a set of hardware and software components that enable hardware engineers to design and implement high speed digital systems both quickly and cheaply. Typically, the hardware components consist of a number of FPGAs, some memory, some peripherals, and a link to a host computer. The software components usually consist of a design tool flow, such as synthesis, placement and routing tools, and an IP library. Through the use of a properly designed hardware platform, an engineer can design and test many different digital systems without having to design a physical hardware platform for each. The only limitations on what is possible are those that arise from the hardware platform itself.

This paper presents a next-generation FPGA-based development system, called the Transmogrifier-4 (TM-4), which removes a number of the limitations found in previous development systems. In particular this paper will focus on improving four key areas: memory depth, memory bandwidth, inter-FPGA bandwidth, and host–to-hardware bandwidth.

This paper is organized as follows: Section 2 provides a brief examination of previous development platforms and describes which previous characteristics that are incorporated into the TM-4. Section 3 describes three application case studies that were used to determine where existing solutions could be improved and summarizes the design requirements of the TM-4. Section 4 presents the design of the TM-4, and Section 5 describes the performance tests and results while Section 6 concludes.

2. Background

The design of the TM-4 builds upon the knowledge gained from previous development platforms. In particular past platforms have shown the benefits of incorporating various numbers of FPGAs into a single platform, as well as providing a variety of different ways to interconnect multiple FPGAs.

Previous development systems have consisted of systems ranging from a single FPGA to dozens of FPGAs spanning multiple circuit boards. For example, the previous two generations of Transmogrifiers, the TM-2 [3] and TM-3 [1], had 32 FPGAs and 4 FPGAs respectively.

Our experience with the 32 FPGAs on the TM-2 showed that it was very difficult for a user to effectively use that many FPGAs, where as 4 FPGAs were much easier to use. Since modern FPGA densities provide large (and increasing) amounts of development logic, large numbers of FPGAs are no longer needed. This fact, combined with past experiences, suggests that the TM-4 should be designed with only 4 FPGAs.

Past development systems (in our own work and that of others) have connected multiple FPGAs in a number of

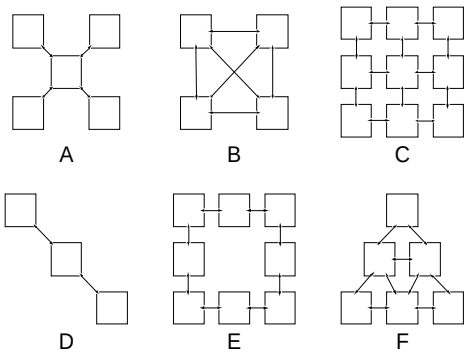

Figure 1: FPGA Interconnect Topologies

different topologies, shown in Figure 1, including: crossbar interconnections [2], [3], [4], [5], [6], [7], [8], hardwired fully interconnected schemes [9], [10], 2D [7], [11] or 3D meshes [12], linear interconnects [13], ring interconnections [14], and tree structures [15].

The selection of which topology is best suited for the TM-4 can be made by pruning out inappropriate options until only one is left. Of the 6 topologies, the last three, D, E and F, are only suited for special cases of applications where as the TM-4 is intended as a general-purpose machine. Topology C is not applicable to the TM-4 as it consists of only four FPGAs. This leaves only options A and B. Since topology B provides lower latency then topology A, with only the cost of additional pins, this topology is best suited for the TM-4.

3. Design Requirement Identification

Our previous generation of Transmogrifier, the TM-3, was used to implement a variety of applications including: ray tracing [16], protein identification [17], [18], and stereo vision [19], [20]. The process of designing these applications brought to light a number of shortcomings in the architecture and capabilities of the TM-3. In particular it was found that the TM-3 lacked host computer bandwidth, memory bandwidth, inter-chip bandwidth and memory depth. In addition, as Moore's law ensues, it is worthwhile to architect a next-generation system to leverage faster and larger capacity FPGAs.

The following subsections examine each of the three application case studies used to guide the design of the TM-4. This is followed by the final design requirements of the TM-4.

3.1. Case Study: Ray Tracing

Ray tracing is a method of rendering 2D images of a virtual 3D scene. The algorithm renders a 2D projection of a 3D scene by approximating the way that light rays propagate around the scene and eventually reach the viewer's eye. The light ray propagation model involves "tracing" the path that light rays travel back from the viewpoint, through the projection point and in to the scene.

A hardware ray tracing implementation [16] on the TM-3 was found to be limited several ways by the architecture of the TM-3. First, the available memory on the TM-3 was limited to only 6 megabytes. This allowed only relatively small 3D scenes to be rendered on the TM-3. The second limitation was memory bandwidth.

The TM-3's memory subsystem was built to run at 50Mhz. At this clock speed the memory could not provide 3D data as fast as the hardware could process it, and this turned out to be the performance-limiting factor in the design of the hardware ray tracer.

The final limitation of the TM-3 was the amount of host computer bandwidth. Both the dataset, which represents the 3D scene, and the resulting 2D image required data transfers between the TM-3 and its host computer. However, the available bandwidth, less the 2MB/s, meant that the TM-3 could process data much faster than it could communicate its results.

Experience from the ray tracing application suggested that the TM-4 should have more memory, more memory bandwidth, and more host computer bandwidth.

3.2. Case Study: Novel Protein Identification

An active area of research in proteomics involves the identification of biological proteins contained in a physical sample. The current approach attempts to identify the molecular makeup of proteins using a device known as a mass spectrometer. This device can take a protein, break it up into small pieces and identify the molecular make up of these small pieces. It is then necessary to assemble these "fragments" into a completed protein. One approach to assembling the fragments involves searching the human genome [17], [18].

The human genome contains a description of every possible protein, and as such can be used to reassemble the protein fragments previously obtained. The algorithm to accomplish this involves searching the entire human genome dataset, several gigabytes of data, and matching the fragments to certain proteins. Successive fragment searches each reduce the set of possible protein matches until only one protein is left. This protein should be the same as the protein in the physical sample.

A prototype created on the TM-3 showed that the algorithm could be easily parallelized to consume all available memory bandwidth. In addition it was found that a large amount of memory was also required to store genomic data. It requires approximately 1 gigabyte of data to store the 3.3 billion base pairs that make up the

human genome. In addition there is evidence that it might be necessary to search several different genome datasets at the same time. This would increase the amount of RAM required to between 2-4GB.

The experience from this application suggests that the TM-4 should have between 2-4GB of RAM and as much bandwidth as feasible.

3.3. Case Study: Stereo Vision

One of the fundamental problems facing the computer vision field is to extract depth information from an image or images of a scene. Stereo vision is one approach to this problem that works by mimicking the way human vision works: two cameras are aligned side-by-side, and each camera sees a slightly different version of the scene. These differences can be used to extract depth information. By utilizing simple geometric relationships between corresponding objects in each image, depth can be calculated. The difficult step is to identify the matching points between each image. One solution to the matching problem is to perform a large number of correlations between the pixels in each image. This approach works well but is very computationally intensive.

The TM-3 was used to accelerate a state-of-the-art the stereo vision computation [21] to the point where it could operate in real time [19], [20]. However, the logic area requirements necessary to meet real time performance were very high. The implementation of the stereo vision algorithm needed to be spread across all four FPGAs of the TM-3. It was found that the lack of communication bandwidth between each FPGA made partitioning the design difficult but in the end a functional stereo vision system was created.

The experience from this application suggests that the TM-4 should have as much inter-FPGA bandwidth as possible in order to simplify the problem of partitioning designs.

3.4. Design Requirements

The design of the TM-4 builds on the past generations of FPGA-development systems by incorporating the key characteristics identified from previous development systems with the improvements identified above. The following list summarizes the different design requirements of the TM-4.

1. Logic capacity: 4 of the largest available FPGAs
2. Interconnect Topology
    a. Fixed point-to-point
    b. As much inter-chip bandwidth as feasible
3. Memory
    a. 4GB or more
    b. As much bandwidth as feasible
4. External Interfaces
    a. 2 analog video in channels
    b. 2 digital video channels (IEEE-1394)
    c. VGA video out DAC
5. Host Computer Interface
    a. As much bandwidth as possible
    b. Simple to use for designers
6. Miscellaneous
    a. Must have a mechanism for remote access to the development platform.
    b. Should be reconfigurable as fast as possible
    c. Designed to minimize the risk of a design error-induced system failure
    d. Circuit board should conform to extended ATX form factor specification

The need to build a new development system, the TM-4, arose from the fact that no commercial development systems could meet the feature set, listed above. The following section will describe the design of the TM-4 and how it was motivated from the design requirements above.

4. The TM-4 Design

The following sections present the design of the TM-4, as motivated by the previously identified design requirements. A system-level block diagram will be introduced for the TM-4 and each block will then be examined in more detail. A complete description of the TM-4 can be found in [22].

4.1. Design Overview

The design of the TM-4 consists of two major subsystems: the FPGA development subsystem, and the interface subsystem. The development subsystem contains the portion of the TM-4 that is directly usable by designers in implementing their designs. The interface subsystem provides support functionality, including both control of the TM-4 and a communication channel from the host computer to the development subsystem.

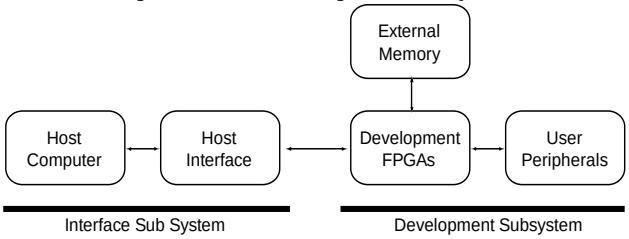

Figure 2: Top Level System Diagram

Figure 2 shows the division between development and interface subsystems. The development subsystem is composed of programmable logic, external memory, and user peripherals that are all available for designers to use. The interface subsystem consists of a host computer and a host interface.

Each of these five components, the programmable logic, the external memory, the user peripherals, the host interface and the TM-4 controller will are all described in the following subsections.

4.2. Programmable Logic

The programmable logic subsystem of the TM-4 is comprised of two different items: the FPGAs and the interconnection between them. The first two design requirements, 1 and 2, specified that the TM-4 should contain four of the largest FPGAs available, be fully interconnected using point-to-point connections and provide as much bandwidth as possible. At the time the components for the TM-4 were being selected the largest FPGAs available were the Altera Stratix and the Xilinx Virtex 2 Pro. Each FPGA had various advantages over the other, such as dedicated DDR SDRAM hardware for the Stratix and greater pin flexibility for the Virtex 2 Pro. The final selection of the Stratix FPGA for the TM-4 was guided by the fact that Stratix FPGAs where shipping before Virtex 2 Pro FPGAs.

Each of the four Altera Stratix S80 chips selected for the TM-4 provide 79,040 four-input lookup tables and flip-flops (logic elements), 7.4Mb of on-chip SRAM, 176 embedded 9x9 multipliers and 1203 I/O pins. When combined the total usable development area of the TM-4 is 316,160 logic elements, 29.6Mb of on-chip SRAM, and 704 embedded 9x9 multipliers.

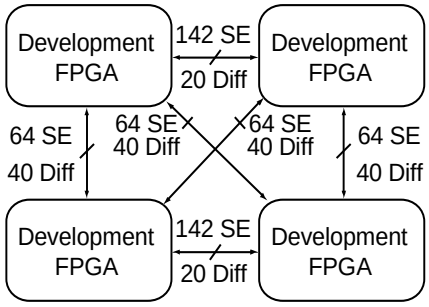

Figure 3: FPGA Interconnect Structure

The point-to-point interconnect topology used to connect the four FPGA is shown in Figure 3. The goal of this topology is to provide as much inter-FPGA bandwidth as feasible. The selection of the number of signals between each FPGA and the signaling standard used was based on both hardware limitations and physical circuit board issues.

For example, modern differential signaling standards were preferred as they provide higher bandwidth then single ended standards. However, there are only a limited number of differential pins available on the Stratix FPGA. This led to the need to incorporate single ended pins in addition to differential signals.

In total the bandwidth between any pair of FPGA varies between 56Gb/s and 66.5Gb/s.

4.3. External Memory Selection

The design requirements specified that the TM-4 should have at least 4GB of memory and have as much memory bandwidth as possible. This raises the questions of what type of memory technology to use, how it should be connected to the development FPGAs, and exactly how much.

The selection of memory technology was driven primarily by practical considerations. The amount of memory required, 4GB or more, meant that it was impractical to use SRAM, because of the number of components that would require. This meant that DRAM was the only practical choice as it provides much greater memory capacity for the same number of components than SRAM. There were two major types of memory module technology available at the time the TM-4 was being designed: DDR SDRAM and RAMBUS. Both technologies provided similar memory densities and bandwidths but DDR SDRAM could be more easily incorporated in the TM-4, due to the hardware support for this type of RAM in the Stratix FPGA [23].

The selection of how to many modules to use, and how to connect them to the development FPGAs, required a balance between performance and cost. The total amount of memory bandwidth is proportional to the number of independent memory modules provided. However, each module comes at the cost of power, space, and expense. Since each Stratix FPGA has hardware support for up to two DDR SDRAM modules, the question became one of either using 1 or 2 independent RAM banks per FPGA. Since memory bandwidth was one of the driving goals of the TM-4 it was decided to use two DDR SDRAM modules per FPGA, for a total of 8 modules in total.

Each of these ram slots can be populated with between 512MB and 2GB of ram running at up to 166MHz, the maximum specified speed for the Stratix FPGA. The standard configuration will contain 8 1GB modules and provide a total peak bandwidth of 17.8GB/s.

4.4. User Peripherals

The design requirements for the TM-4 identified video applications as one possible use for the TM-4, and as

such, specified that the TM-4 should contain two NTSC analog video-in channels, one VGA video-out channel and two independent IEEE-1394 buses.

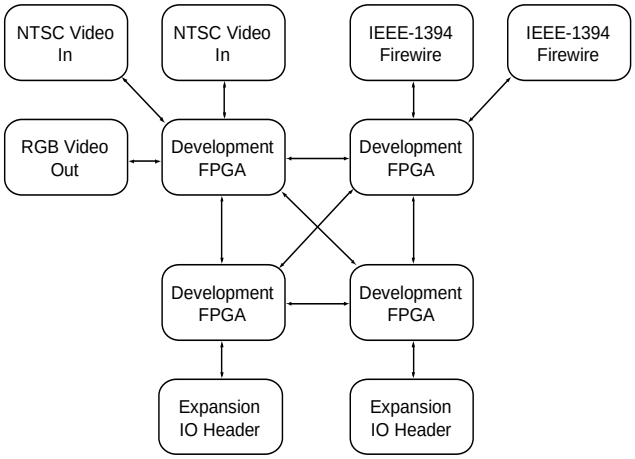

Figure 4: User Peripheral Connections

Since the first two peripherals, analog video-in and video-out, were both present on the TM-3, the same proven design was brought forward to the TM-4. The NTSC video-in channels were implemented using two Phillips SAA7111 decoder chips, and the VGA video-out channel was provided by an Analog Devices ADV7123 triple 10bit video DAC.

The IEEE-1394 bus is a new interface, which was not present on the TM-3. Its implementation is also more complicated due to the complicated communication protocols that it uses. The TM-4 was designed to implement as much functionality in hardware as possible, while still remaining flexible. We selected a 2 chip IEEE-1394 [24] solution. These chips provide both the physical and link layers of the IEEE-1394 networking protocol. Users of the TM-4 must implement the remaining layers using the development FPGA. This division, between hardware components and logic within the development FPGA, was selected to allow the user of the TM-4 sufficient flexibility to control the bus how they see fit. This meant allowing the user to fully control all networking layers above the link layer.

Figure 4 shows how the different peripherals are connected to the four FPGAs. The top left FPGA handles all the analog video peripherals, include 2 video-in channels and one VGA out channel, the top right FPGA handles the two independent IEEE-1394 buses. The two remaining FPGA do not have any specialized peripherals but do have I/O headers available for future expansion.

4.5. Host-to-FPGA Communication Channel

The design of the communication link between the host computer and the development FPGAs was driven by two design requirements: the link should have as much bandwidth as possible, and that it be easy for designers to use the channel. The latter was already solved in the design of the TM-3 by providing a set of IP blocks, and software [25] running on the host computer, that abstract away the complexities of communicating with a host computer. The first requirement, maximizing bandwidth, was the focus of the TM-4's communication channel design.

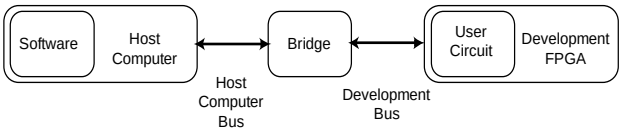

Figure 5: Host-to-FPGA Communication Channel

The communication channel between the development FPGAs and the host computer consists of a number of different components, as shown in Figure 5. For transfers from the host computer to the development FPGAs there are several steps: Software, running on the host computer, initiates the transaction by requests that data be transferred to the development FPGAs. This data is then be transmitted from the host computer to a bridge within the TM-4 itself. This bridge then passes the data on to the development FPGAs and ultimately to the circuit running within it. Transfers in the other direction take similar steps in reverse.

The communication channel consists of four major components: the physical hardware links between the host computer, the bridge chip, and the development FPGAs, the IP core which implements the bridge, the IP cores running on the development FPGAs and the software running on the host computer. Each of these will be examined in the following sections.

4.5.1. Physical Hardware Communication Links

The host communication channel contains two physical hardware links, the link between the host computer and the bridge chip, and the link between the bridge chip and the development FPGAs. Each of these links was designed to meet the design requirement of having as much host computer bandwidth as feasible.

The first link, between the host computer and the bridge chip, needed to use a standard interface that was available in commodity computers. The selected link, was the link that provided the greatest bandwidth, PCI. In particular 66Mhz 64bit PCI was selected. This link provides a theoretical peak bandwidth of 528MB/s.

The second link, between the bridge chip and the development FPGAs, need not have been a standard interface and was custom designed. The link selected was a bus consisting of 32 data bits that could run at a data rate up to a 100Mhz. The result was a communication link that could sustain transfers of nearly 400MB/s. The reasoning behind the bus width and speed were that the bus needed to be easily combinable into 64bit PCI words, by combining two 32-bit words, and that the bus should still run synchronously, by keeping the clock below 100MHz. The resulting 400MB/s bandwidth was not expected to be a bottleneck to system performance due to the fact that the PCI bus's overhead prevents it from reaching its theoretical peak bandwidth.

4.5.2. Host-to-Development Bridge

The connection between the host computer's PCI bus and the development FPGAs communication bus is bridged in the Interface FPGA. This FPGA contains a custom-designed logic core that performs the translation between the two buses. The PCI interface is implemented using an Altera PCI IP core. This core interfaces with two FIFO buffers, used for clock domain translation between the 66Mhz PCI bus and the 100Mhz development bus, and some simple development bus transaction logic.

4.5.3. Parameterizable Bus Interface Logic Cores

The physical communication link between the FPGA and the interface bridge incorporates a custom design bus protocol. In order to hide the complexity of interfacing with this bus, a set of parameterizable logic cores were created. These modules encapsulate all the functionality required to interface with the bus while presenting a simple handshaking based interface to the user. Instead of dealing with multi-cycle bus transactions the user only needs to interface with a simple three-wire handshake interface of one of the parameterizable cores.

4.5.4. Host Software

The final component of the host communication channel is the software that runs on the host computer. This software is responsible for providing a simple interface for communications between user software and user hardware in the development FPGAs. There are two different pieces of software that provides this functionality, a device driver and a software API.

The device driver is a kernel-mode Linux driver that handles all the hardware details necessary to communicate with the TM-4 without requiring any user intervention. The driver provides a simple software API that can be linked into user programs to enable a C program to directly communicate with the development FPGAs. This API provides the ability to both read and write to the FPGAs as well as to catch errors and to monitor the state of the TM-4.

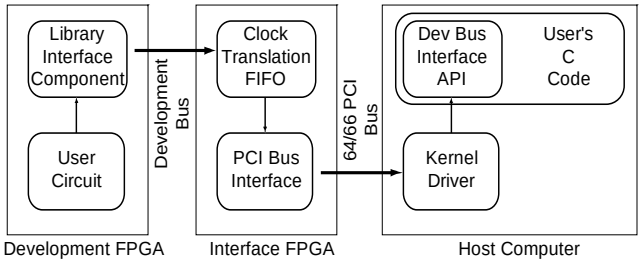

Figure 6: Development Communication Bus

Figure 6 summarizes all the communication steps necessary for communicating between the host computer and the development FPGAs.

4.6. Self Contained Development System

The final design requirement for the TM-4 is that it should be designed with an extended ATX form factor. This selection allows for the TM-4 to be housed in a standard PC case and act as a completely self-contained system.

The complete TM-4 system consists of the hardware motherboard, shown in Figure 7, a plug in single board computer, shown in the right of Figure 7, all of the peripherals that go with such a computer, DVD drive, hard drive, and networking, a Linux operating system, and various software design tools. All of these components are contained in a single standard PC case.

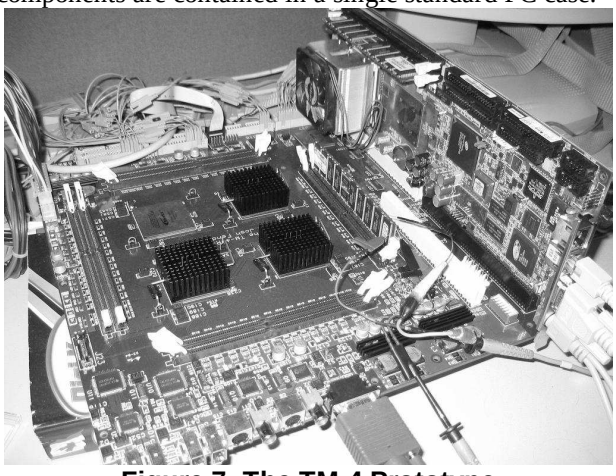

Figure 7: The TM-4 Prototype

5. Performance Results

A prototype of the TM-4 was built, as shown in Figure

7, and its performance was measured. The performance results of each of the primary goals of the TM-4, having as much memory bandwidth, inter-FPGA bandwidth and host-to-FPGA bandwidth as possible, are presented in the following three subsections. It should be noted that the goal of having as much memory capacity as feasible is correct by design, and as such, does not require explicit measurement.

5.1. Memory Performance

Memory performance of the TM-4 was measured by implementing a test circuit, on the development FPGAs that could both read and write from the attached DDR SDRAM. The test circuit consisted of a DDR SDRAM controller and a simple interface circuit that was designed to perform large block transfers. This access pattern was selected because it is representative of applications that work with data streams. A hardware clock counter was used to determine the time taken to complete the block transfers and a bandwidth was calculated.

It was found that under burst access conditions the TM-4 could provide a sustained memory bandwidth of 17.6GB per second total.

5.2. Inter-FPGA Performance

The next goal of the TM-4 was to provide as much inter-FPGA bandwidth as possible. The method used to measure the actual inter-FPGA bandwidth of the TM-4 was to determine the maximum data rate of a single LVDS channel and then to extrapolate this result to the entire set of channels. The procedure used to measure the data rate of a channel was to have two test circuits running on two different FPGAs. The first FPGA would transmit data while the second would receive it. Additionally, other LVDS channels would be driven with random data to simulate cross talk. The clock rate of the circuits was then increased until channel failure occurred.

Unfortunately, a design error involving the selection of clock pins on the Stratix FPGA limited the maximum speed of the LVDS channels to 462Mbps instead of the maximum rated 840Mbps, and enabled only half the LVDS channels to be used. However, under these conditions, the test circuit showed the TM-4 was capable of meeting this reduced speed of 463Mbps. At this speed the total measured aggregate bandwidth between FPGAs is either 577 or 1155MBps, depending on which pair of FPGA is considered.

A second revision of the TM-4 is being produced to fix this error.

5.3. Host-To-FPGA Performance

The final goal of the TM-4 was to provide as much host-to-FPGA bandwidth as feasible. This channel is implemented using a 64bit 66Mhz PCI bus with a maximum bandwidth of 528MBps. However, the expected actual performance of the TM-4 should be much lower due to various overheads.

The procedure used to measure the performance of the host-to-FPGA communication link was to use both software and hardware components. The software component would transfer a large block of data either to, or from, a corresponding hardware circuit on a development FPGA. These transfers were then used to measure the actual performance of the TM-4.

The measured write performance of the TM-4 was found to be 266MBps, and the measured read performance was found to be 154MBps. These performance numbers corresponds to a PCI bus utilization of 50% and 29% respectfully.

Through the use of a logic analyzer it was found that write performance was limited by the host computer's ability to provide data quickly enough to the PCI bus, as it was often found to be idle. Similarly it was determined that reads were limited by the handshaking protocol utilized in the development FPGAs. This handshaking overhead prevented the development FPGAs from transmitting sufficient data to the PCI bus.

5.4. Performance Summary

The three measurement procedures, presented in this section, show how the TM-4 design meets the goals of providing significant memory bandwidth, inter-FPGA bandwidth, and host-to-FPGA bandwidth. In total, the system has a measured memory bandwidth of 17.6 GB per second, an inter-FPGA LVDS communication channel bandwidth, between each pair of FPGAs, of up to 1.15 GB per second, and a host-to FPGA bandwidth of 266 MB per second for writes and 154 MB per second for reads.

6. Current Status

Although the TM-4 is a new system it does have a variety of functional applications, including both simple test applications, such as video in and out, and DDR SDRAM tests, and more complicated applications that have been ported from the TM-3, such as a real time edge detector and a procedural texture mapper. Figure 8 shows the real time edge detector circuit running on the TM-4.

Figure 8: A Real Time Edge Detector

Currently four additional TM-4s are being built for use by researchers at both the University of Toronto and McGill University. It is the goals of these researchers to implement new applications in both the areas of computational vision and bioinformatics, as well as other application domains.

7. Conclusions

This paper presented the design of an FPGA-based rapid prototyping system. The objective of this work was to provide a development platform with as much memory capacity, memory bandwidth, inter-FPGA bandwidth, and host-to-FPGA bandwidth as feasible. The resulting tests, on a prototype system, showed that the TM-4 was able to deliver large amounts of bandwidth in all of the categories.

It is the hope of the authors that the creation of this prototyping system will enable future researchers to implement designs not possible with previous technologies.

8. Acknowledgements

The authors would like to thank Altera, NSERC and Micronet, for providing funding and FPGAs to the TM-4 project, and Marcus van Ierssel for providing valuable technical support.

9. References

[1] The Transmogrifier-3: “http://www.eecg.utoronto.ca/~TM-3”, April 2005.

[2] D. Galloway, D. Karchmer, D. Chow, D. Lewis, J. Rose, “The Transmogrifier: The University of Toronto Field-Programmable System,” Second Canadian Workshop on Field-Programmable Devices, Kingston, June 1994.

[3] D. Lewis, D. Galloway, M. van Ierssel, J. Rose, P. Chow, “The Transmogrifier-2: A 1 Million Gate Rapid Prototyping System,” in IEEE Transactions on VLSI, Vol. 6, No. 2, June 1998. pp 188-198.

[4] C. Chang, K. Kuusilinna, B. Richards, R. Brodersen, “Implementation of BEE: a Real-time Large-scale Hardware Emulation Engine”, in FPGA '03, 2003.

[5] A. Ferrucci, M. Martin, T. Geocaris, M. Schlag, P. K. Chan, "ACME: A Field-Programmable Gate Array Implementation of a Self-Adapting and Scalable Connectionist Network", 2nd International ACM/SIGDA Workshop on Field-Programmable Gate Arrays, 1994.

[6] P. K. Chan, M. Schlag, M. Martin, "BORG: A Reconfigurable Prototyping Board Using Field-Programmable Gate Arrays", Proceedings of the 1st International ACM/SIGDA Workshop on Field-Programmable Gate Arrays, 1992. pp. 47-51.

[7] H. Hogl, A. Kugel, J. Ludvig, R. Manner, K. H. Noffz, and R. Zoz. “Enable++: A Second Generation FPGA-Processor for ATLAS,” ATLAS internal note DQS-NO-026, CERN, 1994.

[8] J.M. Arnold et al., "The Splash 2 Processor and Applications," Proc. Int'l Conf. Computer Design, CS Press, Los Alamitos, Calif.. 1993, pp. 482-485.

[9] Transmogrifier 3A, University of Toronto, “http://www.eecg.toronto.edu/~TM-3”, Jan 2005.

[10] D. Smith and D. Bhatia. “RACE: Reconfigurable and Adaptive Computing Environment,” In 6th International Workshop 117 on Field-Programmable Logic and Applications, Darmstadt, Germany, September 1996. pp. 87-95.

[11] W. Eatherton, T. Schiefelbein, H. Pottinger. “An FPGA-based Reconfigurable Coprocessor Board Utilizing a Mathematics of Arrays,” Technical report, University of Missouri--Rolla, Computer Science Department, 1995.

[12] G.M. Quenot, I.C. Kraljic, J. Serot, and B. Zavidovique. “A Reconfigurable Compute Engine for Real-Time Vision Automata Prototyping,” In IEEE Workshop on FPGAs for Custom Computing Machines, 1994. pp. 91-100.

[13] T. A. Petersen, D. A. Thomae, D. E. Van den Bout. “The AnyBoard: A Rapid-Prototyping System for Use in Teaching Digital Circuit Design,” In Proceedings, The First IEEE International Workshop on Rapid System Prototyping RSP-90, Computer Society Press, 1991. pp. 25-32.

[14] K. Bouazza, J. Champeau, P. Ng, B. Pottier, and S. Rubini. “Implementing cellular automata on the ArMen machine,” In P. Quinton and Y. Robert, editors, Proceedings of the Workshop on Algorithms and Parallel VLSI Architectures II, Bonas, France, June 1991. pp. 317-322.

[15] M. Wazlowski, L. Agarwal, T. Lee, A. Smith, E. Lam, P. Athanas, H. Silverman, S. Ghosh. “PRISM-II compiler and architecture”. In Proceedings of IEEE Workshop on FPGAs for Custom Computing Machines, Napa, California, April 1993. pp. 9-16.

[16] J. Fender, J. Rose, "A High-Speed Ray Tracing Engine Built on a Field-Programmable System," in IEEE International Conf. On Field-Programmable Technology, December 2003, pp. 188-195.

[17] A. Alex, J. Rose, R. Isserlin-Weinberger, C. Hogue, "Hardware Accelerated Novel Protein Identification," in Int'l Symp. on Field-Programmable Logic, Aug 2004, pp. 13-22.

[18] "Hardware Accelerated Protein Identification", Anish Alex, M.A.Sc. Thesis, University of Toronto, 2003. “http://www.eecg.toronto.edu/~jayar/pubs/theses/Alex/AnishAlex.pdf”

[19] A. Darabiha, J. Rose, W. J. MacLean “Video-Rate Stereo Depth Measurement on Programmable Hardware,” Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision & Pattern Recognition, June 2003, Madison, Vol. 1, pp. 203-210.

[20] "Video-Rate Stereo Vision on Reconfigurable Hardware," Ahmad Darabiha, M.A.Sc. Thesis, University of Toronto, 2003. “http://www.eecg.toronto.edu/~jayar/pubs/theses/Darabiha/AhmadDarabiha.pdf”

[21] D.J. Fleet, Disparity from local weighted phase-correlation, Int. Conf. on Systems, Man, and Cybernetics, pp. 48-54 v.1, 1994.

[22] "An FPGA-Based Hardware Development System with Multi-Gigabyte Memory Capacity And High Bandwidth," Joshua Fender M.A.Sc. Thesis, University of Toronto, 2005. “http://www.eecg.toronto.edu/~jayar/pubs/theses/Fender/JoshFender.pdf”

[23] Stratix Device Handbook, “http://www.altera.com/literature/hb/stx/stratix_handbook.pdf”, April 2005.

[24] Texas Instruments: TSB41AB2, “http://focus.ti.com/docs/prod/folders/print/tsb41ab2.html”, April 2005.

[25] TM-3 Ports Package, “http://www.eecg/~TM-3/”, April 2005.