An Open-Source SATA Core for Virtex-4 FPGAs

Cory Gorman, Paul Siqueira, and Russell Tessier
Department of Electrical and Computer Engineering
University of Massachusetts, Amherst, MA, USA

***Abstract*—In this demonstration, we present an open-source Serial ATA core designed for Virtex-4 FPGAs. This core utilizes the RocketIO Multi-Gigabit Transceiver (MGT) of the Virtex-4 to interface with hard drives at SATA Generation 1 (SATA I, 1.5 Gb/s) and Generation 2 (SATA II, 3.0 Gb/s) speeds. A full design hierarchy from host software to the physical layer is provided with the distribution to facilitate design use. A simple, FIFO interface allows for easy integration with other FPGA modules. The demonstration illustrates the correct write and read behavior of the core using a Xilinx ML405 board and a solid state disk. The peak transfer rate of the core for SATA I (130 MB/s) is demonstrated. Our goal for the demonstration is to educate the reconfigurable computing community regarding the availability of the core and to illustrate its capabilities.**

I. Introduction

The need for bulk, non-volatile storage in FPGA-based systems is significant for a large variety of applications [1]. FPGA system developers can most easily adapt these interfaces to applications using an open-source solution. In this demonstration, we present the successful operation of a Serial ATA (SATA) host controller core which can transfer data at close to peak SATA I protocol speeds. This interface is optimized for Xilinx Virtex-4 FPGAs, a widely-used FPGA family. Our demonstration involves 1024 consecutive data block write and read operations. The validity of the transfers is confirmed via messages sent to a computer screen.

Via a poster, our presentation also carefully describes the structure of our open-source synthesizable core which will be made available to the reconfigurable computing community. The core supports read and write operations with SATA-compliant hard drives and solid state drives (SSDs). It features an easy-to-use FIFO interface to allow for integration into larger hardware designs. An embedded Linux system (e.g. running on a MicroBlaze soft processor) can be used to create a complete file system on the disk, or the core can be used in a hardware-only setup.

Some features of the new core have been adapted from another open-source SATA design [2] that targets Virtex-6 designs. However, the limitations of this previous design have required the development of a number of new features for this new core:

1) Since the Multi-Gigabit Transceivers (MGT) of older Virtex-4 devices are more limited in their capabilities than their Virtex-6 counterparts, an entirely new Physical Layer design has been provided.
2) New error control circuitry has been added which re-sends data if a transmit error is detected.
3) A new clocking scheme has been developed to allow for the synchronization of read requests.
4) The design is integrated with a MicroBlaze soft processor to provide a flexible control interface.

To our knowledge, this core is the first open-source SATA core for Virtex-4 devices.

II. Background

A. SATA Protocol

SATA is a peripheral interface standard created in 2003 to replace Parallel ATA (PATA). The fastest PATA speed is 133 MB/s, while SATA Generation 1 (SATA I) operates at 150 MB/s. SATA has a number of other features that make it superior to ATA, including a smaller cable, fewer pins, and a lower operating voltage. The SATA protocol uses a layered architecture where each layer uses the services of the layer below it. The Application Layer typically represents the software using the SATA device and the Command Layer emulates PATA commands for backward compatibility. The Transport Layer handles creating and formatting data frames called Frame Information Structures (FISs) and valid sequences of FISs. Beneath that layer is the Link Layer, which encodes and encapsulates the FISs, handles control signals, and checks for FIS integrity.

The lowest layer of the stack is the Physical Layer, which handles the transmission and reception of the electrical signals and maintains data alignment. SATA uses low-voltage differential signaling (LVDS) to send and receive data and 8b/10b encoding for clock and data recovery. The encoding assigns a 10-bit character for each 8-bit data value. Instead of sending 1's and 0's relative to a common ground, the sent data value is based on the difference in voltage between two conductors sending data. The Physical Layer also establishes communication with the disk. This action is performed with out-of-band (OOB) signaling, where the communication lines of the transmitting pair are driven to the same voltage to create an absence of a signal difference. This action is performed in a pre-defined burst pattern to send an OOB primitive. Three OOB primitives (COMRESET, COMINIT, and COMWAKE) are defined for SATA, which are used as part of an initialization handshake to begin communication with a SATA device.

B. Xilinx SATA Core Distributions

A commercially-available SATA core for the Virtex-4 is described in a Xilinx application note [3]. A demonstration of the core is provided for the ML405 evaluation board, which contains a Virtex-4 device. This core design, termed

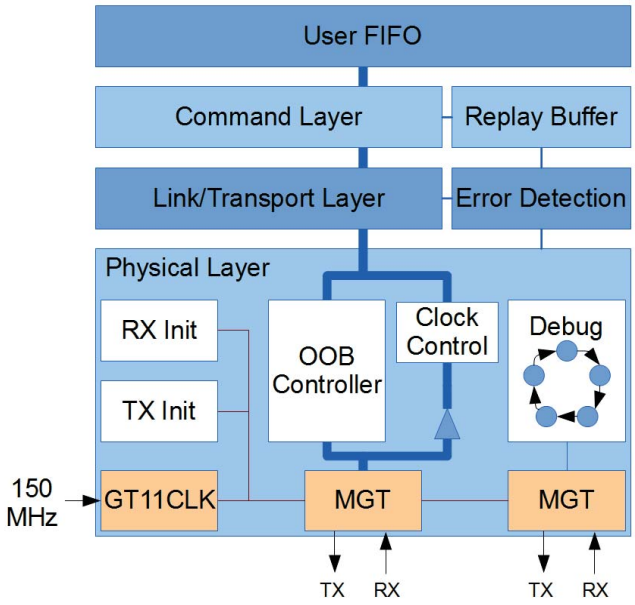

Fig. 1. SATA host controller block diagram

the embedded SATA storage system (ESS), is only available under a for-purchase license. Although the internal details of the core are unavailable, its presence indicates the feasibility of implementing SATA I and II interfaces on a Virtex-4 device.

In 2012, an open-source SATA core for Virtex-6 devices was made available to the reconfigurable computing community by researchers from the University of North Carolina, Charlotte [2]. This *UNCC core* was demonstrated using an ML605 board. The core implements all layers of the SATA protocol, combining the Link Layer and Transport Layer into one module. A simple, FIFO interface is presented to the Application Layer so that other hardware modules can easily read and write data. The UNCC core implements the DMA Read and DMA Write commands of SATA, but not other commands and features, such as power management. As described in the next section, portions of the upper layers of the UNCC core were used in this work, although modifications for debugging and error checking were added. Additionally, the Physical Layer hardware which controls the MGT was completely replaced.

The Groundhog SATA adapter [4] provides an additional open-source SATA option for FPGA users. This host bus adapter is targeted to Xilinx Virtex-5 devices. The core features support for native command queueing (NCQ), which allows for the out-of-order execution of commands. This feature forms part of the SATA II protocol.

III. SATA Host Controller Design

A block diagram of the new SATA host controller appears in Fig. 1. In the following subsections, the behaviors of the different blocks are detailed.

A. Physical Layer Design

The Virtex-4 RocketIO MGTs [5] are used for the high-speed serial communication required for SATA. These built-in FPGA hard blocks are designed for high speed I/O. The MGTs handle the electrical and low-level aspects of the Physical Layer, and can run at line rates from 622 Mb/s to 6.5 Gb/s. SATA I uses a line rate of 1.5 Gb/s and SATA II uses a rate of 3.0 Gb/s. The MGTs have their own clocking resources (e.g. PLL - phase-locked loops) on the FPGA, and require a dedicated, low-jitter reference clock. The ML405 board has a 150 MHz oscillator for this purpose. However, to support a line rate of 3.0 Gb/s, a 300 MHz oscillator is required, so SATA II is not demonstrated on this board. We have successfully used the core in SATA II mode on a Virtex-4 based board [6] used for radar signal processing.

The Virtex-4 device on the ML405 board has eight RocketIO transceivers, two of which are connected to SATA connectors. These two transceivers share clocking resources. RocketIO MGTs contain SERDES blocks to serialize and deserialize the data coming from and going to the FPGA fabric. They also contain a PLL for clock recovery from the incoming bitstream. For SATA I, the PLL's oscillator runs at 3000 MHz, and this frequency is divided down to form a 750 MHz clock. The SERDES uses both edges of this clock to achieve the line rate of 1.5 Gb/s. The MGT contains internal clock dividers and presents 37.5 MHz clocks to the FPGA fabric for sending and receiving data. The RocketIO also includes a built-in 8b/10b encoder and decoder block, which is used in this design. The built-in CRC cannot be used for the SATA protocol since all data, including CRCs, are scrambled at the Link Layer to reduce EMI.

Virtex-4 RocketIO MGTs include support for OOB signals, which is necessary for SATA. However, this support is not as robust as OOB support for the Virtex-6. The Virtex-6 GTX transceivers have built-in OOB primitive features that can detect and generate the COMRESET, COMINIT, and COMWAKE primitives used by SATA. The Virtex-4 MGT modules have only two OOB ports. One port indicates whether a signal is detected on the communication line, and the other is used to send an OOB signal by driving output pins to a common voltage. Thus, generating and detecting primitives on the Virtex-4 must be done with a user configuration of resources. In this design, multiple OOB modules were created to perform this action.

The RocketIO Wizard tool, part of the Xilinx ISE toolkit, was used to generate wrapper modules and set parameter values for the MGTs. The wizard also generates initialization modules for the MGT blocks. For this design, an interface width of 4 bytes (1 Dword) was chosen. This value is convenient because all Link Layer primitives are 32 bits wide. The (transmit) TX and (receive) RX FIFOs located in the MGT are bypassed to reduce latency and meet the timing requirements of SATA. Rather, specially-designed FIFOs are developed from FPGA block RAM resources to serve as interfaces between the Physical and Link Layers. The OOB controller contains sub-modules that handle generation and detection of the OOB primitives. This controller also handles ALIGN primitive insertion which helps determine the negotiated transfer speed. An OOB sequence issue arises since the

MGT's PLL is not able to lock to the incoming datastream within the time specified by the SATA protocol. Our core continues the OOB sequence anyways, waiting for the eventual lock. If the OOB start-up sequence fails, it will automatically be retried until it is successful.

B. Error Control Circuitry and Debugging

Debugging to ensure reliability was an important part of the design process. Integrated Xilinx Chipscope logic analyzer cores were included at each layer of the design. Additional debugging modules were also created to help the debugging process. These modules are included in the open-source distribution, but they can easily be turned off using a compile-time parameter. Circuitry to implement a SATA event logger was also added to the design. Chipscope, while useful, can only capture a limited number of consecutive samples. Events that occur far from each other in time generally cannot be captured. To overcome this limitation, we created a monitor module in the Link Layer that records significant events and sends them over the RS232 serial port. This monitor watches the TX and RX datapaths of the Link Layer and the main Transport Layer state machine. Also, it tracks the received data for newly received primitives and for alignment errors. Events are timestamped, so that the designer can get an idea of how far apart in time these events are occurring. This allows for debugging of problems that are too far apart in time for Chipscope to capture.

Although most data write transfers will finish without problems, we found that a small number (less than 0.001%) would result in an error. As written, the UNCC core does not have any error detection features. Errors cause the core to pause (or "hang") until it is reset. It is desirable for the core to recover from these errors gracefully, preferably in a way that is invisible to the rest of the design. To facilitate this recovery, a replay buffer has been added to the Command Layer. The replay buffer mirrors the most recently sent Data FIS. When an error occurs, a flag is raised. If the error occurred on a Data FIS, the Command Layer sends the data to the MGT from the replay buffer rather than from the User FIFO which stores the next value to be transmitted. Thus, to the rest of the hardware design, it appears as if there was no error; instead, it seems that the SATA core is operating more slowly.

A new handshaking signal has been added to the Link Layer that indicates successful transmission of a FIS. After transmission, the Command Layer flushes the replay buffer and sends the next FIS to the Link Layer. The replay buffer is 8 KB in size, which is the maximum allowable size for a single FIS. A bit error is the simplest error which leads to straightforward recovery. This action occurs when the SATA device reports a bad CRC or a parity error. With the new error detection and recovery features, the Command Layer simply starts a new transfer using the replay buffer.

C. Host Controller Clocking and Interface Control

The choice of clocking configuration for the host controller core is very important for reliability. One possibility is to use derivatives of the transmit (TXOUTCLK1) and receive (RXRECCLK1) clocks used by the MGT. Incoming data is written to a small clock-domain-crossing FIFO. Data in this FIFO is written using the receive clock and read using the transmit clock. This configuration raises the issue of FIFO underflow. The transmit clock is slightly faster than the receive clock, so data will be read slightly faster than it is written. A small module in the Physical Layer monitors for this underflow condition and delays the datastream when it occurs. At the top level, the SATA core presents a simple FIFO interface that other modules use to store and retrieve data. In our experimentation, a MicroBlaze soft core processor is used to perform write/read tests.

TABLE I
SATA CORE RESOURCE UTILIZATION

Resources	Our core	XAPP716 [7]
Slices	5,128 (61%)	6,839 (80%)
Block RAMs	7 (10%)	36 (52%)

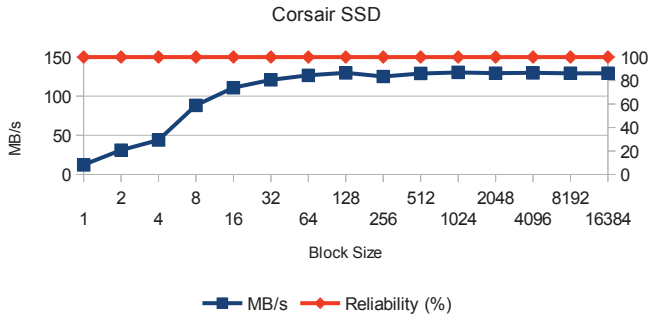

Fig. 2. Corsair SSD performance test. Block sizes on the x-axis are measured in sectors. Each sector contains 512 bytes

IV. Experimental Results and Demonstration

The SATA core was tested on an ML405 evaluation board and a custom board [6] that has a Virtex-4 XC4VFX140. Version 14.2 of the Xilinx ISE tools was used to synthesize the Verilog and VHDL code. The core was tested for functionality using a Corsair NOVA CSSD-V30GBA, a SATA Generation 2 SSD. The SATA core uses a modest amount of resources. Table I compares the resources used by our new core to the resources reported for the previous SATA core for a Virtex-4 FPGA [3]. Our SATA core uses fewer resources than the commercial core presented in XAPP716, although the earlier design also includes resources to connect the core to an embedded PowerPC processor.

A. Experimental Results

A MicroBlaze soft core processor is used to perform the tests. A small control module was created to interface the SATA core with the MicroBlaze. This allows the processor to set various parameters, such as sector count and address, and initiate data transfers. The control module also gives status information to the processor. A simple software test application runs the tests and returns data to a console window

Fig. 3. Virtex-4 SATA core demonstration setup

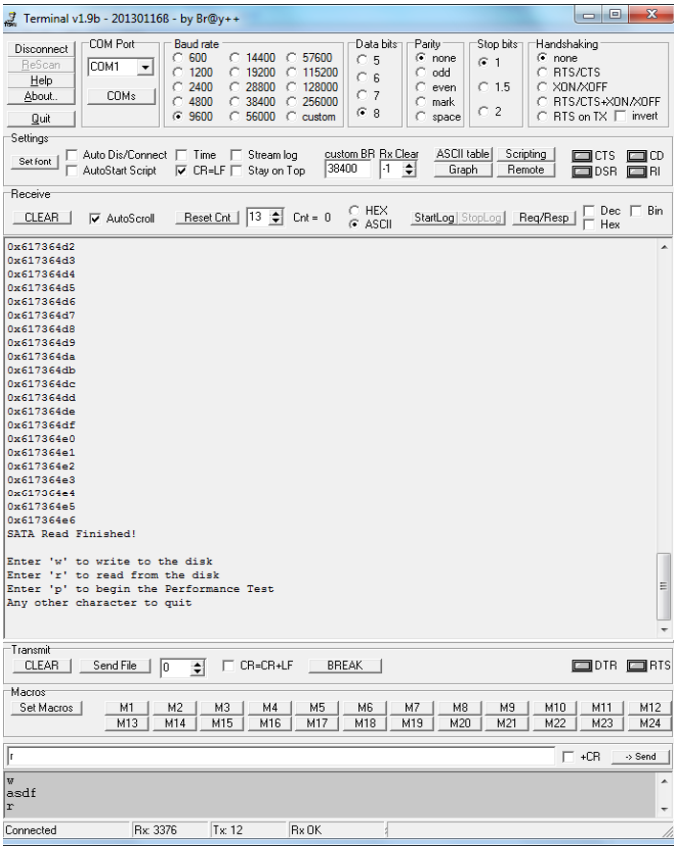

Fig. 4. Demonstration interface for SSD read verification

on a PC over the serial port. This application issues 500 write commands of varying block sizes. After each command, the number of cycles taken to complete the transfer is stored and used to calculate the average throughput in MB/s. A transfer is considered a failure if it takes more than 3 seconds to complete. In this case, the application resets the SATA core and continues. The data written to the disk in the test is read from a simple counter and fed into the user FIFO of the SATA core. The FIFO is always kept full to test the true maximum transfer rate.

As seen in Fig. 2, the SSD has a maximum write speed of 130 MB/s at block sizes of at least 128 sectors (64 KB). Larger block sizes take advantage of the larger data transfer per transfer initiation.

B. Description of Demonstration

In the demonstration, the correct operation of the core and its peak performance for SATA I is shown. The equipment shown in Fig. 3 is used for the demonstration. The SATA host controller core is connected to a MicroBlaze processor in the Virtex-4 FPGA which configures its operation. A counter is used to generate data at peak rate which is then transferred to a Corsair SSD at SATA I rates. A total of 1024 block writes is performed. The data is then read from the disk and each data value is verified. The read and expected data values are transferred to an attached laptop via a serial port and displayed on a terminal (Fig. 4) to illustrate the correct behavior. A peak transfer rate of 130 MB/s is demonstrated.

V. Conclusions and Future Work

In this demonstration, an open-source Serial ATA core designed for Virtex-4 FPGAs is presented. The core is shown to transfer information to and from a SATA I solid state drive at a rate approaching 1.5 Gb/s using a Xilinx ML405 board. Separately, successful SATA II transfers have been shown in our laboratory using a different board. User circuitry can interact with the new core via a FIFO interface. This SATA core is available for use by additional projects in the research community.

Acknowledgments

We thank Xilinx for the donation of the ISE 14.2 and ISIM software. The assistance of Justin Lu in preparing the demonstration is appreciated.

References

[1] K. Pocek, R. Tessier, and A. DeHon, "Birth and adolescence of reconfigurable computing: A survey of the first 20 years of field-programmable custom computing machines," in *Highlights of the First Twenty Years of the IEEE International Symposium on Field-Programmable Custom Computing Machines*, Apr. 2013, pp. 3–19.

[2] A. Mendon, B. Huang, and R. Sass, "A high performance, open source SATA2 core," in *International Conference on Field-Programmable Logic and Applications*, Aug. 2012, pp. 421–428.

[3] S. Tam and L. Jones, "Embedded serial ATA storage system," in *Xilinx Application Note 716*, Oct. 2006.

[4] L. Woods and K. Eguro, "Groundhog - a serial ATA host bus adapter (HBA) for FPGAs," in *IEEE International Symposium on Field-Programmable Custom Computing Machines*, Apr. 2012, pp. 220–223.

[5] "Virtex-4 RocketIO multi-gigabit transceiver: user guide," Xilinx Corporation, Application Note UG076, Nov. 2008.

[6] S. J. Lu, P. Siqueira, V. Vijayendra, H. Chandrikakutty, and R. Tessier, "Real-time differential signal phase estimation for space-based systems using FPGAs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, no. 2, pp. 1192–1209, Apr. 2013.

[7] "Serial ATA I/II host controller (SATA_HI). product specification," ASICS World Services. May 2008.