An Ethernet-Accessible Control Infrastructure For Rapid FPGA Development

Andrew Heckerling, Thomas Anderson, Huy Nguyen, Greg Price, Sara Siegal, John Thomas
MIT Lincoln Laboratory, Lexington, MA 02420
{heckerl, toma, hnguyen, gprice, ssiegal, jthomas}@ll.mit.edu

Introduction

Field Programmable Gate Array (FPGA) technology is widely used in many application areas such as intensive numerical processing, sophisticated control, and high-speed IO interfacing. One key factor for success in such vast diverse application space is the flexibility to customize (configure) the on-chip logic fabric and embedded specialized silicon macros to produce designs that are streamlined and highly integrated. This “flexibility”, on the other hand, also means a minimally-supported development environment, which makes application development and debugging more difficult. An analogy is a PC board with no operating system or BIOS. The lack of a minimum standardized control infrastructure on the FPGA is also a huge barrier for meaningful interaction with the outside world. As a result, input-output and control structures for FPGAs are frequently created from scratch for each project.

This paper addresses the above limitations with a computer-accessible control structure on the FPGA. This small-footprint control infrastructure provides the developer with a foundation and scaffolding to quickly build up an application. Through Ethernet, external software can observe and control internal states of the application function core being developed. The infrastructure is essentially a container for housing the application-specific intellectual property cores (IPs).

The container structure consists of four major FPGA components. The first major component is a UDP controller, which implements the UDP protocol and decodes packets into DMA commands. The second is the DMA controller, which executes DMA transactions on a Wishbone bus. The third is the Wishbone bus itself, which is a simple interconnect between the DMA controller and a variety of registers and peripherals. Finally, there are a number of Wishbone peripherals that are useful for FPGA development.

UDP Controller

The UDP controller receives packets from an Ethernet MAC and decodes properly addressed and formatted UDP packets into commands for the DMA controller. Once the command has been executed by the DMA controller, the resulting status and data responses are re-packaged into UDP messages and reported back to the network address that made the request.

UDP was chosen as a transport-layer protocol for efficiency reasons and because its stateless nature made it more suitable for implementation in digital logic than a more complicated protocol like TCP. The command-response protocol implemented on top of UDP was designed for simple translation into commands for the DMA controller.

DMA Controller

The DMA controller receives commands to read or write a block of address space and translates them into the required master read or write cycles on the Wishbone bus. The status of the completed read or write transaction and, in the case of a read, the resulting data is reported to the upstream controller that sent the command. In our design, this is the UDP controller, but the DMA controller is not UDP-specific. Transaction size can range from a single four-byte word to 8 KB of data, and the DMA controller supports both constant-address and ascending-address reads and writes.
WISHBONE Bus

The WISHBONE SoC Interconnect Architecture is an open source, public domain bus architecture for System-on-Chip architectures. The WISHBONE specification, interconnect IP, and a variety of peripherals are freely available from the OpenCores web site.

The communication rate with the computer reached 13 MB/s, the highest rate supported by our minimally-optimized software library. It is estimated that the FPGA infrastructure can exceed this limit by a wide margin and reach Gigabit Ethernet speeds or higher.

Results

On the Virtex 5-50SXT, the following resources were used:

<table>
<thead>
<tr>
<th>LUTs</th>
<th>FFs</th>
<th>BRAM Kbytes</th>
<th>Clock rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>5941/32640</td>
<td>10118/32640</td>
<td>112.5 / 594</td>
<td>125 MHz</td>
</tr>
</tbody>
</table>

The communication rate with the computer reached 13 MB/s, the highest rate supported by our minimally-optimized software library. It is estimated that the FPGA infrastructure can exceed this limit by a wide margin and reach Gigabit Ethernet speeds or higher.

This control infrastructure was used as a computer-FPGA interface for a FPGA-based processor in MicroTCA platform, as shown in Figure 3. The MicroTCA environment has a GigE hub connecting to all payload slots in the system via backplane high-speed connection.

Future Work

Possible future work includes extending the container framework to a multi-chip environment as shown below in Figure 4.

The container can also be further developed to support integration of FPGAs as co-processors in the PVTOL framework.

---

2 http://www.opencores.org/projects.cgi/web/wishbone/wishbone
3 http://www.opencores.org/projects.cgi/web/wb_builder/overview

---