AN ENTRY-LEVEL PLATFORM FOR TEACHING HIGH-PERFORMANCE RECONFIGURABLE COMPUTING

Pablo Viana, Dario Soares, Lucas Torquato

LCCV - Campus Arapiraca
Federal University of Alagoas

ABSTRACT

Among the primary difficulties of carrying out integration of digital design prototypes into larger computing systems are the issues of hardware and software interfaces. Complete operative systems and their high-level software applications and utilities differs a lot from the low-level perspective of hardware implementations on reconfigurable platforms. Although both hardware and software development tools are more and more making use of similar and integrated environments, there is still a considerable gap between programming languages running on regular high-end computers and wire-up code for configuring a hardware platform. Such a contrast makes digital design too hard to be integrated into software running on regular computers. Additional issues include programing skills at different abstraction levels, costly platforms for reconfigurable computing, and long learning curve for using special devices and design tools. Hence coming up with an innovative high-performance reconfigurable solution, besides its attractiveness, becomes a difficult task for students and non-hardware engineers. Thus we propose a low-cost platform for attaching an FPGA device to a personal computer, enabling its user to easily learn to develop integrated hardware/software designs to accelerate algorithms for high-performance reconfigurable computing.

1. INTRODUCTION

Recent discovery of huge oil and gas volumes in the pre-salt reservoirs of Brazil's Santos and Carioca basins have fueled the concern about the new challenges and dangers involved in the off-shore exploration. During the last two decades, the high cost and risk involved in the activity have pushed the research community to come up with innovative solutions for building and simulating virtual prototypes of structures, under the most realistic conditions, to computationally evaluate the performance of anchors, risers (oil pipes) and underwater wells before to experience the open sea.

Nowadays, the highly-detailed numeric models that help engineers to develop and improve new techniques for oil and gas exploration, demand a considerable computing throughput, pushing research labs on this field to invest on state of the art solutions for high-performance computing. The term *High-Performance Computing (HPC)* is usually associated with scientific research and engineering applications, such as discrete numerical models and computational fluid dynamics. HPC refers to the use of parallel supercomputers and computer clusters, that is, computing systems comprised of multiple processors linked together in a single system with commercially available interconnects (Ethernet Gigabit, Infiniband, etc.). While a high level of technical skill is undeniably needed to assemble and use such systems, they need to be daily operated and programmed by non-computer engineers and students, who are intrinsically involved in their specific research fields.

Specialist engineers in oil and gas production offshore have developed in the last 15-20 years programing skills to implement their own programs and build their prototypes using state of art programing techniques and following updated rules of software engineering. They had to learn about design patterns, multi-threaded programming, good documentation practices, and started to develop many other skills for adopting open-source trends on modern cluster programming. In order to explore the processing resources even more effectively, HPC engineers are also supposed to be able to take advantage from the parallelism of graphical processing units (GPUs), that boost processing power of most supercomputers figuring on the the Top 500 List [3] .

Researchers on HPC seems to be aware about the need for continuously pursuit new alternatives to overcome some of the main computing challenges. Power consumption, temperature control and the space occupied by large supercomputers are the main concerns for the next-generation systems [2]. In this context, the promises of reconfigurable computing seems to be suitably matched to the demand for innovative solutions for high performance computing. Beyond the basic advantages on size, and energy consumption of popular reconfigurable platforms, like the Field Programmable Gate Arrays (FPGAs), it is expected that reconfigurable computing can deliver no precedent performance increase, compared to current approaches based of commercial, off-the-shelf (COTS) processors, because FPGAs are

essentially parallel and might enable engineers to freely construct, modify, and propose new computer architectures. It is expected that the parallel programing paradigm shall become much more than just parallel threads running on multiple regular processors. Instead, we may expect that innovative designs may involve also the development of some Processing Units (PU), specifically designed to deal with parts of an algorithm, in order to reach the highest performance.

Hardware specialists from computer engineering schools have been extensively prepared to develop high-performance designs on state-of-the-art reconfigurable devices, such as encoders, filters, converters, etc. But paradoxically, most of the people interested in innovative high-performance computing are not necessarily hardware engineers. In contrast, these users are non-computer engineers and other scientists, with real demands on high performance computing. These professionals know deeply their needs on HPC and probably would be empowered if they could wire-up by themselves innovative solutions from their own desktops. This class of users would need to be capable of rapidly building and evaluating prototypes even without the intrusive interference of a hardware designer.

Since there is a considerable learning curve to master hardware design techniques and tools, it's straightforward to point out the problem of shortly training people from other knowledge areas with specific hardware design skills. In order to tackle the problem, we involved Computer Science undergraduate students to assist non-computer engineers on improving the performance of a given existing system, by mixing the legacy software code with hardware prototypes of Intellectual Property cores (IP cores). We then proposed to develop modules specifically designed to be easily attached to a regular desktop machine, through a common USB (Universal Serial Bus) interface. Such an approach enable non-computer engineers to experience the reconfigurable computing benefits on their native code, by inserting calls to the remote procedures implemented in the FPGA device across the USB interface. On the other hand, the proposed platform allows computer science and engineering students to develop reconfigurable computing solutions for real-world problems. As the result, we propose a integrated platform for introducing students and engineers on high-performance reconfigurable computing.

This paper is organized as follows. Section 2 is devoted to discuss the hardware issues that motivated us to propose a simplified platform for teaching reconfigurable computing, by defining templates and a protocol to integrate logic design to a general purpose computer system. Section 3 illustrates the utilization of reconfigurable computing platform on engineering applications, and finally in Section 4 we discuss the achieved results, future improvements on the integrated platform and the next applications on high-performance computing.

2. LOW-COST VERSATILE RECONFIGURABLE COMPUTING PLATFORM

2.1. Hardware Issues

Reconfigurable computing FPGA-based platforms, from distinct manufacturers of logic and third parties, are fairly available on the market. Most of them support a varied number of interfaces to connect the board with other external devices, such as network, VGA monitors, ps2 keyboards, as well as high performance interconnect standards such as PCI express, Gigabit Ethernet, etc.

Basically, state-of-the-art platforms offers high density programmable logic devices with millions of equivalent gates and support high-speed interconnects, among other facilities, allowing the user of such platforms all the versatility needed to develop complex designs such as video processors, transceivers, and many other relevant projects. These platforms are suitable for small or complex prototypes, and their price range from $\$500 - \$5000$, not including all the necessary software design tools. Although this category of platforms offers an attractive support to a great variety of experiments and prototype designs, the total cost for acquiring a number of boards, becomes its adoption prohibitive on classes. On the other hand, there are on the market low cost platforms, equipped with medium density devices (around 500k equivalent gates), and priced under $\$200$, such that most schools and training centers can afford. There are, however, restrictions on the interface support offered by these platforms that may restrict their utilization to stand-alone devices.

As a participant of the Xilinx University Program (XUP), our platform (Figure 1) is based on the low-cost donation board *Xilinx Spartan 3E FPGA*, available at our laboratory for teaching purposes. Although this specific board contains several devices and connectors around the FPGA chip to allow students to experiment projects integrated with network, video (VGA), keyboard, serial standard RS-232 and some other interfaces, the USB connector present on the board can only be used for programming the logic devices (FPGA and CPLD).

Initially, we tried to access the logic resources on the board from a personal computer over the network interface. The board has an RJ-45 Ethernet connector, as well as a physical layer chip. But the user needs to implement the Data Link layer in order to provide a MAC (Media Access Control) to the network, besides the next Network Layer that implements the basic communication functionalities across the network into the FPGA. This first try rapidly became a hard solution to implement, since most of our students were not familiar with digital design yet.

The second alternative tried to take advantage from the 100-pin Hirose connector available on the board to implement an wide interface to an external device. The external

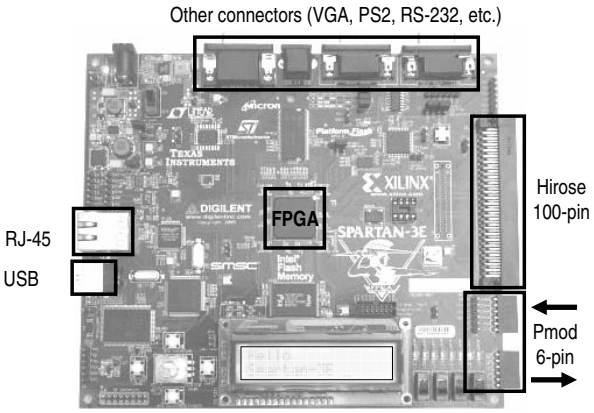

Fig. 1. FPGA Platform: Xilinx Starter Kit Spartan-3e

device should have a friendly interface to a personal computer through a standard USB interface version 2.0. Due to specificities of the 100-pin connector, this hard to find adapter would demand a hard to wire device with one hundred pins to connect. Again, in order to keep our approach as simple as possible, we decided to adopt the three 6-pin Pmod connectors. The 6-pin connectors can be found on the local market and are easy to connect to a few ports of a small microcontroller with USB capabilities. Two out of six pins are dedicated to source power (+5V and GND) and the other four pins can be used for general purpose. Then, only 12-pins are actually available for interconnecting the board through the Pmod connectors.

We then proposed a 4-bit duplex interface based on a microcontroller Microchip PIC18F4550, capable of transferring bytes to and from the FPGA board by dividing each byte word (8-bit) into 2 nibbles of 4-bit. Four pins for writing to the FPGA, other four pins to read from the FPGA and other four pins to control the writing/reading operation. Since the whole procedure of the microcontroller to read the USB port, send to the FPGA, read from the board and finally send over the USB takes just a little less than 1 microsecond, transfers between the FPGA and the Microcontroller can reach the maximum theoretical throughput of 1MByte/s.

The microcontroller board utilized as interface was rapidly built because we utilized a pre-defined platform, namely *USB Interface Stargate* [1]. The Stargate platform is intended to facilitate designers to propose new HID (Human-Interface Device) products. The board offers analog and digital input/output pins, and is capable of easily communicate with a personal computer by the USB version 2.0 making use of native *Device Drivers* of most popular Operating Systems. We simply made some minor adjustments on the Stargate's firmware to implement the defined protocol to communicate with the FPGA through the 12-pin interface. In order to directly connect the Stargate to the FPGA board,

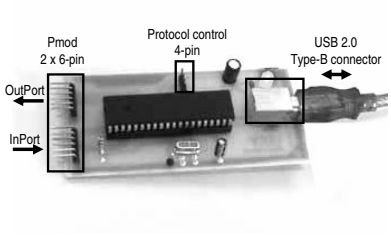

Fig. 2. Scorpion Interface Board: Modified USB Interface Stargate to interconnect USB to the FPGA.

some minor changes on its layout had to be made, eliminating the analog pins and replacing the input and output pins on the same corner. The updated firmware and the proposed layout matching the physical requirements of this project motivated a new name for this specific platform:*Scorpion Interface board* (Figure 2).

2.2. Defining a Communication Protocol

We proposed a simple protocol to enable communication between the FPGA board and the interface USB built in the microcontroller (PIC18F4550). The firmware program PIC is intended to send and receive Bytes to and from the FPGA board, according to a established protocol, exclusively defined for the purposes of this integration.

Basically, the firmware is a loop code waiting for data coming from the USB interface with the PC. As soon as a byte arrives, the word is split up into the low-half and high-half nibbles (Figure 3). The less significant portion is put available at the *OutPort* to the FPGA and the bit flag *Write-enable* goes high. The data stays available until FPGA's flag *Write-done* is raised. Then, Scorpion interface board makes available the second portion bits and reset *Write-enable* down. The half-byte data is read by the FPGA board, who clears *Write-done*, ending the USB to FPGA writing operation.

If there is any data made available by the FPGA, the *Read-enable* flag will be set 1. Then, Scorpion reads the lower bits to *low-half* and sets *Read-done* high. Now the FPGA makes the higher portion bits available at the *InPort* and resets *Read-enable*. The microcontroller at Scorpion read the data and concatenates both parts to recover the whole byte before send it over the USB to the PC.

2.3. Integration Wrapper

In order to help the logic designer be more focused on the design of the processing unit itself, we proposed a parameterizable wrapper template, which hides the communication protocol between the FPGA and the Scorpion boards. The data input coming from the Scorpion board is shifted along the input operator registers (Op InReg) to become readily available to the Processing Unit (PU). The PUs must be defined as combinatorial logic, that is, output data will be

```
While(true)
{
  Wait (until receive from USB);
  OutPort = low_half;
  Write_enable = 1;
  Wait (until Write_done == 1)
  OutPort = high_half;
  Write_enable = 0;
  Wait (until Write_done == 0)
  if(Read_enable == 1)
  {
    low_half = Read(InPort);
    Read_done = 1;
    Wait (until Read_enable == 0)
    high_half = Read(InPort);
    Read_done = 0;
  }
  SendToUSB(high_half+low_half);
}
```

Fig. 3. Protocol defined in the Scorpion interface firmware

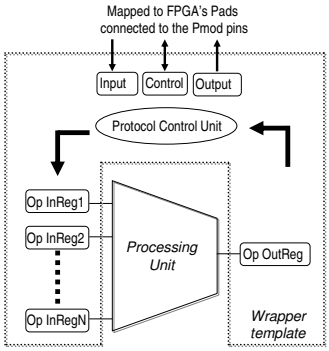

Fig. 4. Wrapper template around the PU.

available when all the inputs are sourced at the input registers. The output register will keep the data available as long as the input data have not changed (Figure 4).

This restriction simplifies the rapid development of simple Processing Units and their respective integration to the platform. The integrated PU implemented on the FPGA becomes available to the end-user, who can immediately evaluate the hardware/software implementation and explore innovative design options.

3. INTEGRATING RECONFIGURABLE LOGIC INTO AN USER APPLICATION

The proposed platform for integrating the traditional development on general purpose computer and library modules implemented in reconfigurable hardware (FPGAs), is here essentially the task of including the *libhid* library on the C/C++ code and make use of the Application Program Interface (API) designed to send and receive bytes over the USB port (Figure 5). The API enable the use of the the proposed platform, offering an abstraction layer to hide from

```
sendFPGA(buffer, size);
size = receiveFPGA(buffer);
```

Fig. 5. Basic functions to send and receive data to and from the FPGA

the user the enumeration steps of the Scorpion interface as a device connected to an USB port.

Our illustrative example and first exercise to get started on the proposed platform requires the student to design an Adder/Subtractor in the FPGA. After have written the HDL code and synthesized the project into the FPGA, the user must send 3 bytes over the interface to the FPGA, which are: The operation value (referring to an *ADD=0x00* or *SUB=0x01* operation), the first and the second operand. Such an operation must be carried out by enclosing all the values in a buffer and send it to the FPGA. Next, the size returned by *receiveFPGA* will determine when the result is available at the buffer.

4. CONCLUSION

Computer Science students were able to design functional modules of processing units in VHDL, synthesize their codes and configure the FPGA device by using the tools from the University Program donation at the lab classes. The proposed processing units typically included functional modules, such as arithmetic operators and statistical estimators. At this present time, collaborators from a partner research laboratory involved in the non-computer engineering projects, such as oil and gas production, can make use of the proposed modules, by attaching the proposed platform with a pre-configured FPGA with the functional implementation of a given operator to compare the results obtained from the hardware implementations. Although performance issues are not the major contributions of this paper, thanks to the easy-to-use proposed platform, students and engineers are more and more considering reconfigurable computing on their academic and research projects. Our next steps include the use of high-end platforms with high-density FPGAs and high-speed interconnects to propose alternative solvers to existing scientific computing libraries.

5. REFERENCES

[1] USB Interface: http://www.usbinterface.com.br

[2] Experimental Green IT Initiative Launches on Recycled HPC System , Scientific Computing (2009)

[3] TOP500 Supercomputing list, available at: http://www.top500.org/