Homayoun

Reference 20

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

CUBE: A 512-FPGA CLUSTER

Oskar Mencer, Kuen Hung Tsoi, Stephen Craimer, Timothy Todman and Wayne Luk

Dept. of Computing, Imperial College London
{o.mencer,khtsoi,s.craimer,tjt97,wl}@doc.ic.ac.uk

Ming Yee Wong and Philip Heng Wai Leong

Dept. of Computer Science and Engineering
The Chinese University of Hong Kong
{mywong,phwl}@cse.cuhk.edu.hk

ABSTRACT

Cube, a massively-parallel FPGA-based platform is presented. The machine is made from boards each containing 64 FPGA devices and eight boards can be connected in a cube structure for a total of 512 FPGA devices. With high bandwidth systolic inter-FPGA communication and a flexible programming scheme, the result is a low power, high density and scalable supercomputing machine suitable for various large scale parallel applications. A RC4 key search engine was built as an demonstration application. In a fully implemented Cube, the engine can perform a full search on the 40-bit key space within 3 minutes, this being 359 times faster than a multi-threaded software implementation running on a 2.5GHz Intel Quad-Core Xeon processor.

1. Introduction

Reconfigurable gate array technology has been used in many areas for both research and industrial applications; examples include cryptographic systems, architectural exploration, multimedia processing, physical or financial simulation and system emulation. The major advantage of reconfigurable platforms over general purpose processors and ASICs is the balance between circuit level specialization and programming flexibility.

The available resources in field programmable gate array (FPGA) devices increase each year due to Moore's Law with the addition of embedded RAM, DSP block and processor core, but the demand for more programmable resources is even higher as more sophisticated systems are being implemented. A common solution is to use multiple FPGA devices for a single design. In such an environment, design partitioning, data communication and logic configuration become increasingly complicated with the number of devices employed.

Although research on computing systems with large numbers of parallel ICs or large numbers of processing elements on a single IC has been well studied, studies with large numbers of reconfigurable devices have not been fully explored. Practices applied to systems with small numbers of devices are not applicable to systems with hundreds of FPGAs. In particular, issues concerning the clock distribution scheme, data communication paths, configuration requirements and the increasing cost of system debugging requires new ideas and techniques on both hardware construction and development flow.

The lack of a cost effective massive FPGA cluster framework has become an obstacle for researchers exploring the properties and applications on this class of system. In this paper, we describe a massively-parallel reconfigurable platform designed for both advancing research in the field and solving real-world applications. The major contributions of this work include:

- A novel massively-parallel FPGA architecture, called the Cube, is proposed. The architecture balances scalability, flexibility and fault tolerance, providing a low cost, high density and versatile research platform for large scale parallel reconfigurable clusters.
- A Single Configuration Multiple Data (SCMD) programming paradigm is used in the Cube for efficient FPGA configuration. Using SCMD, all 512 FPGAs can be programmed to the same bitstream in parallel within a few seconds which is suitable for constructing large scale systolic processor array.
- A prototype of the Cube platform was built and tested. The hardware consists of multiple boards each hosting 64 Xilinx FPGA devices. The complete system is a 512-node FPGA systolic array with 3.2 Tbps inter-FPGA bandwidth. A standard I/O interface and simulation framework are also provided.
- A key search engine was implemented in the Cube to demonstrate the computing power of the system. With 49152 independent key search cores, it can fully search the 40-bit key space of the RC4 encryption algorithm in 3 minutes.

Section 2 reviews previous work on massively-parallel processing (MPP) platforms. Section 3 details the architecture and design details of Cube platform. Section 4 presents a fully functional 64-FPGA module, the critical component of the Cube platform. Section 5 describes and evaluates an RC4 key search engine for the Cube. Finally, Section 6 presents conclusions and describes future directions of the Cube project.

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

2. Related Work

The basic idea of MPP is to partition the problem into subtasks and distribute them to different nodes called processing elements (PEs). Total processing time is reduced as computations in the PEs are in parallel. This section reviews some contemporary MPP systems.

In 1994, the first prototype of the GRAPE-4 [1] system for computing the N-body problem in astrophysics was presented. In 1995, the measured peak performance of a completed GRAPE-4 system was reported as 1.08 Tflops [2]. The system had 40 modules, each carrying 48 Hermite AcceleRator Pipeline (HARP) processors. The HARP was a dedicated ASIC for gravitational force computations running at 15MHz. All modules in GRAPE-4 were connected to a central host station through a shared bus. In 2002, the GRAPE-6 system with 1728 to 2048 processors achieved 64 Tflops [3]. Each processor in GRAPE-6 had 4 independent force pipelines. The processors were connected in a hierarchical network including switch boards and Gigi-bit Ethernet. In 2005, an SIMD architecture, Network on Chip (NoC) and other approaches were proposed for the new GRAPE-DR system [4] which targeted Pflops performance. The current GRAPE hardware designs are specialized for gravitational force computations and do not support more general applications.

The Berkeley Emulation Engine 2 (BEE2) system was developed for event-driven network simulation in 2004 [5]. In BEE2, five Xilinx Virtex-II Pro 70 FPGAs were hosted on a single Print Circuit Board (PCB). A star topology was used to connect the four computational FPGAs in a 64-bit ring and a control FPGA as the center of the star network. All connections between FPGAs and on-board memories ran at 200MHz. Computationally intensive tasks ran on the outer ring while the control FPGA ran a Linux OS and managed off-board I/Os. The asymmetry between the control FPGA and computation FPGA complicated the programming model.

In 2006, COPACOBANA, a low cost cryptanalysis system using large numbers of FPGAs was described [6]. In the system, 6 Xilinx Spartan-3-1000 FPGAs were grouped in a DIMM module. All modules were connected by a shared 64-bit data bus on a DIMM backplane. In a 2007 implementation [7] the system running at 136MHz can search a full 56-bit DES key space in 12.8 days. New versions of the hardware described in 2008 used more powerful Xilinx Virtex-4 FPGAs. This system is scalable in physical form but not logically. Users can add more DIMM modules as needed to expand the system but are limited by the global shared bus. Unlike cryptanalysis, most applications require communication between PEs, where the shared bus architecture becomes a bottleneck.

In 2007, Nvidia released their C compiler suite, CUDA, for Graphic Processing Units (GPU) [8]. Users can use the standard C language to utilize the massively-parallel threading feature in GPU for general purposes computation. In a GPU chip, simple PEs executing linear threads communicate to each other through shared memory. GPUs are increasingly attractive to both academia and industry due to ease of programming and high-performance floating point units. The scalability of GPUs is largely limited by their dependency on a host computer system; data communication overhead between the host and GPU through a PCIe interface makes it difficult to integrate large numbers of GPU chips with low latency over a dedicated high speed network.

3. System Architecture

In this section, the details of the Cube architecture are presented. Fig. 1 shows the block diagram of a complete Cube platform. Each FPGA is considered as an individual PE. All PEs are connected in a systolic chain with identical interfaces between them. There are no storage components in the system except for the PEs' internal memories. Also, there are no global wires for data communication and clock distribution. All FPGAs can be configured with the same design concurrently. Each PE accepts data from the previous one, processes them and passes them to the next PE. There are several advantages to this approach.

- **Scalability:** A centralized shared bus/memory architecture is not suitable for scaling up to massive amount of elements due to resource conflicts. The cost of full point-to-point topologies such as cross-bars increases exponentially with the number of PEs and thus become prohibitively expensive in systems with hundreds of PEs. In the Cube platform, a linear systolic interface is used which has cost which is linear with the number of PEs.
- **High Throughput:** Synchronizing high frequency clocks between 64 FPGA devices on a single board or across multiple boards is difficult. In the Cube system, short tracks between neighboring FPGA devices for clock and data distribution can easily achieve over 100MHz clock rates for inter-PE communication. Also, minimizing the overhead of handshaking and traffic switching results in low latency and deterministic communication channels.
- **Rapid Development:** Design partitioning and workload distribution in a large scale FPGA cluster are eased by a unified interface and by each PE playing a symmetric role in the system. All FPGAs can be programmed to the same configuration making for constant time configuration, rather than having time proportional to the number of PEs.
- **Low cost:** On-board tracks in the Cube are less expensive than the high speed switches and backplanes employed previous systems. Also, the regular layout in the Cube avoids the expensive and time consuming processes of testing and verifying the signal integrity

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

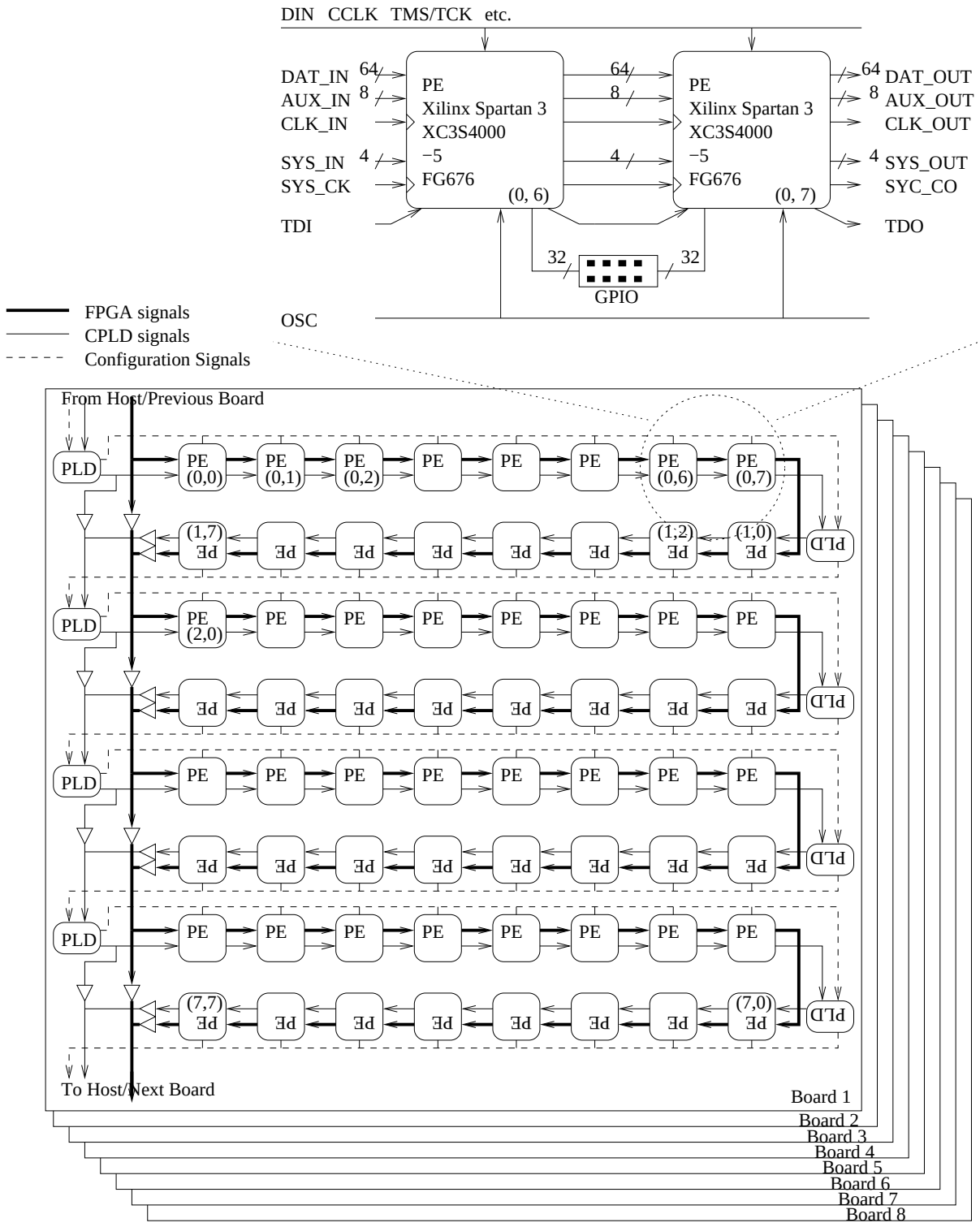

Fig. 1. Cube architecture.

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

of the boards.

3.1. The FPGA Systolic Array

Each module in the Cube platform hosts 64 Xilinx FPGAs (XC3S4000-5-FG676) arranged in an 8 by 8 matrix as shown in Fig. 1. Each FPGA has a unique ID indicating the logical X-Y location of the device. Eight FPGAs are grouped together in a row and have independent configuration inputs and power supplies. The complete system consists of 8 connected boards in a cabinet forming an $8 \times 8 \times 8$ cluster of 512 FPGAs, and thus named Cube.

There are two systolic buses in the system: the `PE` bus is the major data communication channel between PEs and the `SYS` bus is used for system control. Each PE has a 64-bit data bus I/O (`DAT`), an 8-bit auxiliary bus I/O (`AUX`) and a dedicated clock line I/O (`CLK`), connecting the previous PE to the next PE. The `SYS` bus, with 1 dedicated clock line and 4-bit data I/Os, goes through all PEs and CPLDs. All these buses connect adjacent PEs only. These short point-to-point parallel buses significantly simplify the programming model and higher inter-PE bandwidth is achieved. The requirements on PCB layout and FPGA I/O interface are also relaxed for this topology compared to gigahertz serial communications in other designs. On the other hand, the systolic chain enables multiple boards to be cascaded for better scalability without reducing the I/O clock rate. The PE bus was designed to work at 100MHz and thus providing 6.4Gbps data bandwidth between PEs with additional control and handshaking signals on the `AUX` bus.

All these buses are freely available for user designs. The buses are also routed from/to external headers for communication between host and board or between multiple boards. In most applications, `CLK_IN` is driven by previous PE or external source from headers. The clock is then replicated for use internally and forwarded to the next PE through a delay locked loop digital clock manager (DCM) in the FPGA. In the design, the `DAT` and the `AUX` buses can easily match the wire length of the clock line for improved I/O thoughput.

The internal logic of the PE can only be used after the input clock source is stable. There is a delay between DCM reset and when the clock output is usable. The long distribution lines of the global reset and the long cascaded chain of 64 DCMs make it impossible to synchronize the DCM locking sequence of a 64-FPGA module concurrently. To solve this problem, the `LOCKED` output of the current DCM is used to reset the following DCM. This proceeds in a sequential fashion as described in [9]. Global synchronization of clock signals is feedforward in nature and skew is dependent on the performance of the DLLs. An additional 25MHz oscillator is provided in each row for increased flexibility. This clock source is broadcasted to the row and shared by both FPGAs and the row associated CPLD.

3.2. Configuration of FPGAs

In the Cube platform, different FPGA configuration schemes are provided under SCMD for minimum configuration time and maximum flexibility. Considering the number of FPGAs and the size of the board in our design, commodity programming equipment cannot provide sufficient driving power to configure all devices concurrently. Thus a CPLD (Xilinx CoolRunner-II XC2C256-VQ100) is installed in each row to control and drive the configuration signals. Both JTAG and Slave Serial (SS) programming modes are supported by selecting the `M1` input to the FPGA through the CPLD. This can be controlled by on board DIP switches or external host through the `SYS` bus. The CPLDs are programmed by a separated JTAG chain.

Slave serial (SS) mode provides the fastest way to configure all FPGAs in parallel. The SS configuration signals are sampled and buffered by internal Schmitt triggers in the CPLDs and thus all associated FPGAs receive clean and synchronized signals. As shown in Fig. 2, there are three output links from each CPLD for SS configuration. Two of these links broadcast the signals to FPGAs in the odd and even position of the row, while the third link sends the SS signal to the CPLD of the next row. This provides extra flexibility for enabling user to program different configurations in odd and even FPGAs. It is also possible to program different configurations to different rows of FPGAs.

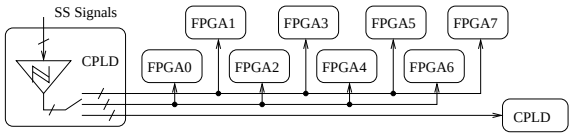

Fig. 2. Slave Serial Configuration Mode in Cube.

JTAG mode allows users to program individual FPGAs and read back internal values. Using JTAG in conjunction with SS mode enables users to configure most FPGAs in parallel rapidly and change the contexts of some FPGAs later. For example, it may be necessary to change the head and tail of the systolic chain in certain applications.

3.3. External Interface

The first and the last FPGAs in the systolic chain are connected to external headers on the 64-FPGA board. Both the `PE` and the `SYS` buses are available. For both input or output, there are 78 signal lines grouped into three standard IDE headers which can be connected to external devices or another module in the Cube through standard IDE ribbon cables.

There are also three pairs of programming headers for CPLD JTAG, FPGA JTAG and FPGA Slave Serial configuration. By bridging the output headers of the current module to the input headers of the next module, a single set of programming cables can be used to configure 512 FPGAs

DOCKET ALARM
Find authenticated court documents without watermarks at docketalarm.com.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.