Cycle-Accurate 64+-Core FPGA-Based Hybrid Simulator

G.X. Liu, G.H. Li, P. Gao, H. Qu, Z.y. Liu, H.X. Wang, y.B. Xue, D.S. Wang

Cycle-Accurate 64^{+}-Core FPGA-Based Hybrid Simulator

G.X. Liu, G.H. Li, P. Gao, H. Qu, Z.Y. Liu, H.X. Wang, Y.B. Xue, D.S. Wang

***Abstract*— Nowadays, computer architecture researches mainly focus on the multicore hardware and software design. As compared with the traditional uniprocessor counterpart, the system complexity of multicore simulators is dramatically augmented, which is spurred by the increase in core number. Full-system fidelity, fast simulation speed, and cycle-level accuracy are the essential requirements of the advanced multicore simulator adopted as the research vehicle for studying and evaluating multicore systems. Tsinghua Emulation Accelerator of Multicore System or TEAMS is an FPGA accelerated cycle-accurate hybrid simulation platform with full-system fidelity. By leveraging state-of-the-art Xilinx Virtex 5 Field Programmable Gate Arrays (FPGAs) and the high volume on board SRAM and DRAM, the most concerned issues in multicore microprocessor research, i.e., Cache with data coherency and network on chip (NoC), are emulated and accelerated by hardware in our simulator. Therefore, developers can achieve the precise performance evaluations, which include the hardware cost, the power consumption and the process efficiency, of Cache and NoC in their designs. On the other hand, the functional simulation of each core is executed by the software simulator, which makes the proposed simulator suitable for a wide variety of instruction set processors. By using hardware-software and inter-FPGA synchronizing techniques, the proposed platform realizes the 64-processor cycle-accurate simulation with one FPGA rack. Together, the proposed techniques also enable the outstanding scalability. Without an overwhelming effort, our simulator can perform the 512-core simulation with four FPGA racks.**

***Index Terms*— multicore, hybrid simulation, Simics, FPGA, NoC, cache coherence**

I. Introduction

In recent years, the "power wall" and the limitation of instruction-level parallelism in typical applications have driven the microprocessor architecture and software researches onto the road of multicore architectures, which is the unique opportunity currently found for maintaining the computer performance trends by capitalizing on silicon technology advances. Historically, the simulation of computer architecture is mainly based on the software simulator. As compared with the Application Specific Integrated Circuits (ASICs) prototyping, the software simulators have the advantages in flexibility and fabricate cost. These features make the software simulators widely adopted in academia. For example, *PTLsim* is a cycle accurate x86 microprocessor simulator and virtual machine for x86 and x86-64 instruction sets; *SimpleScalar* is an uniprocessor micro-architectural simulator; *Simics*[1] is a multicore supported full-system simulator, which can run commercial operating systems and applications. As compared with the uniprocessor architectures, the main differences in multicore architectures come from Cache and NoC designs, and much endeavor has been devoted in these fields. To accommodate the precise evaluation requirements of Cache and interconnection network, the software cycle-accurate cache model (*GEMS*)[2] and NoC model (*GARNET*)[3] are developed by Wisconsin University and Princeton University, respectively.

However, as the research vehicle for studying computer architecture, the software simulators lack of the execution speed and the evaluation accuracy. The speed dilemma is further exacerbated when facing the cycle-accurate multicore simulations. For instance, on *SunFire v40z* Sever, it takes 4.3 hours for *GEMS* to simulate 16-thread 1024×1024 LU decomposition on 16-core system. When the core and thread numbers are both increased to 32, under the same conditions, the simulation time is extended to 11.8 hours. To improve the simulation speed, Chung [4], et. al, apply FPGA to accelerate the common-case behaviors of *Simics* and this approach achieves up to 1.2-4.0 times speed up ratio as compared with *Simics* fast mode. RAMP Gold project [5] applies the similar approach to accelerate the functional simulation of multicore system. Running on the advanced BEE3 platform [6], RAMP Gold garners over an order faster performance than *Simics*. Nevertheless, the aforementioned schemes merely enhance the functional simulation speed, therefore, the cycle-accurate models, including *GEMS* and *GARNET*, can not benefit from the work in literatures [4] and [5]. In terms of evaluation fidelity, since the mechanism gap between the model built with C language and the synthesizable model described with HDL, the models provided by *GEMS* and *GARNET* always contains some flaws that hinder them from efficiently implemented with circuits. For example, the message buffer in *GEMS* is heap structure, which automatically pushes the most recently message to the top, and the size of heap is infinite. In consequence, the simulation fidelity coming from the associated software models is severely degraded.

To address the above problems, we propose using FPGAs to fabricate the reconfigurable simulation testbed, which cooperates with *Simics* to implement the cycle-accurate hybrid simulator. The Cache and NoC designs of microprocessor are mapped to the FPGAs platform and the functional behavior of all cores are simulated by *Simics*. We can construct 64- to 512-core system from multiple FPGA racks. The proposed simulator provides an excellent environment to quantitatively analyze and evaluate the hardware and software of multicore system.

This paper presents the initial implementation of our hybrid

The authors are with Tsinghua University, Lab4-304, FIT Building, Tsinghua University, Beijing 100084, China (liuzhenyu73@tsinghua.edu.cn). This research is supported by fund from National Natural Science Foundation of China (60833004).

simulator. The hardware architecture is depicted in Section II. The simulation principle of our proposal is described and analyzed in Section III. Section IV reports the current research status on 64-core UltraSPARC workstation simulation. The conclusions and our future work are provided in Section V

II. TEAMS Hardware Architecture

The simulator platform is composed of one software simulator host (*Simics*) and several FPGA racks, as depicted in Fig.1. *Simics* runs on the host computer and the FPGA racks are in charge of the Cache and NoC emulation. Four-lane PCI-Express (PCI-E4X) connection is employed to carry out the communication between the host computer and the FPGA racks. The connection between two FPGA racks is mainly realized by the (PCI-E16X) interface. In this section, our discussion primarily concentrates on the hardware architecture of each FPGA rack.

Since our hardware is applied to speedup the simulation of Cache and NoC in the multicore system, the reconfigurable logics, on chip memories and Gigabit IO interface for inter-chip communication are precious. In contrast, the embedded hard processor (PowerPC Processor) are rarely required to our task. Therefore, we choose Xilinx XC5VLX220T FPGA, which has the best cost performance, to build the infrastructure. XC5VLX220T contains 34,560 Slices, 7,632K-bit, 6 Clock Management Tile (CMT), 16 RocketIO GTP transceiver, and 1 PCI-E interface block. In the following paragraphs, we will describe how these modules play their roles in our hardware design.

As shown in Fig.1, one FPGA rack mainly comprises one mother printed circuit board (PCB) and six daughter-PCBs attached on the mother-PCB via 188-pin Compact Peripheral Component Interface (cPCI). The mother PCB is equipped with one Xilinx XC5VLX220T FPGA, one PCI-E bridge (PEX8648), and one CPLD (XC95288XL). The mother-PCB also has one SD interface and one USB interface, which are connected to CPLD. Each daughter-PCB integrates one Xilinx XC5VLX220T FPGA, 2GB 400MHz DDR2 SDRAM connected to one DIMM slot, and 200MHz 512k$\times$36-bit SRAM (36-bit comprising 32-bit data and 4-bit ECC). The mother- and daughter-FPGA JTAG pods are connected in the daisy-chain style. The FPGA can be dynamically programmed by the host computer or through the on board SD Flash card and CPLD. Multiple FPGA configuration bit files can be stored in the SD Flash card and CPLD select the required version to program the FPGAs belonging to the rack. Moreover, SD Flash card also provides the persistent storage capacity for other FPGA-specific information. In addition, the USB interface coupled with CPLD provides another approach to other devices for the FPGA rack.

The communication between the mother- and the daughter-FPGAs is implemented through the dedicated generic parallel IOs (GPIO). In details, for each daughter-PCB, there are 98-bit IO signals directly connected with the central FPGA on the mother-board. These IO signals work at 150MHz clock speed, which provides maximum 14.4Gb/s IO-bandwidth for each channel between the daughter- and the mother-PCBs. In our design, we further provide the Gigabit serial IO connection between any two daughter boards. As we know, there are sixteen RocketIO GTP modules in each XC5VLX220T, then, these components are utilized to realize the Gigabit serial IO connection. For each end-to-end serial connection, our design provides bidirectional 5Gb/s IO bandwidth.

Finally, the PCI-E bridge PEX8648 enables the communications between the FPGA rack and the host PC, and the data exchange among FPGA racks. PEX8648 is a 48-lane PCI-E2.0 bridge, supporting packet cut-thru with a maximum latency of 140ns (16X to 16X). PEX8648 also provides end-to-end CRC protection and poison bit support to accommodate the high end-to-end data integrity requirement. The PEX8648's twelve ports can be flexibly configured. Specifically, in our design, eight of PEX8648's twelve ports are configured as seven 1X, one 4X, and one 16X interfaces. The seven PCI-E1X interfaces are connected to the seven FPGAs belonging to the rack. One PCI-E4X interface implements the communication between the host PC and the FPGA rack. In this way, the host PC can survey the internal status of any FPGA in the rack. The inter FPGA rack data exchange is realized via the PCI-E16X interface of PEX8648.

It should be noted that the mother-PCB and all daughter-PCB all have their own crystal oscillators. Consequently, the FPGA rack has two operational modes, i.e., synchronous mode and asynchronous mode. In the synchronous work mode, all PCBs share the clock signal of the mother PCB, which simplifies the interface circuits design, but the clock speed is low accordingly. In contrast, if the system works at the asynchronous mode, all PCBs apply the local oscillators as their own clock signals. With this scheme, each FPGA has high clock speed, but the hardware cost and the time overhead for the inter clock domain communication are increased. In the initial implementation, we adopt the synchronous mode for each FPGA rack.

III. Cycle-Accurate Simulation Mechanism

This section offers the abridged general view about the proposed partition of the target system across the FPGA emulator and the *Simics* software simulator, as shown in Fig.2. The schematic diagram considers a 4×4 two dimensional (2D)-mesh chip-scale multicores (CMP) as an illustrative example.

The CMP is organized as an array of replicated tiles, which are connected over 2D-mesh network. Each tile is composed of one core, the private L1 I/D-cahce and the shared L2 Cache. L2 Cache is tightly coupled to the local core and provides the fixed access latency. The L2 Cache also can be utilized by other tiles, but the associated access latencies are variable, which are affected by the source-destination distance and the network congestion. In our initial implementation, the generic MESI protocol is adopted to maintain the on-chip data coherence. The NoC router applies the *wormhole switching* and the deadlock-free routing algorithms for *wormhole switching*, such as XY, Odd-Even Turn, and DyXY etc.

Since the researches of multicore emphasis on Cache and NoC, we provide the partition scheme as shown in Fig.2. Specifically, the behavior of each core is mapped to *Simics* and

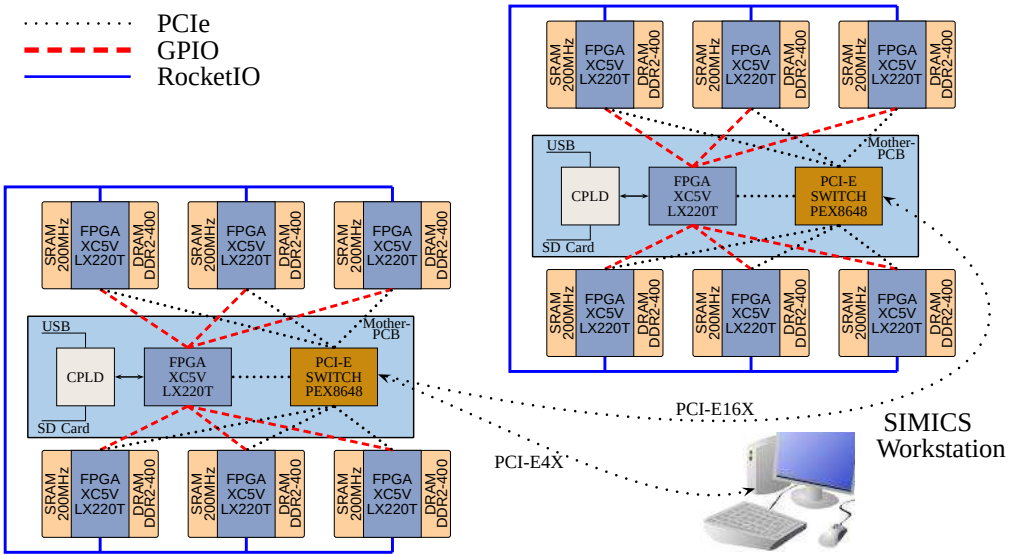

Fig. 1. Top block diagram of the proposed hybrid simulation accelerator

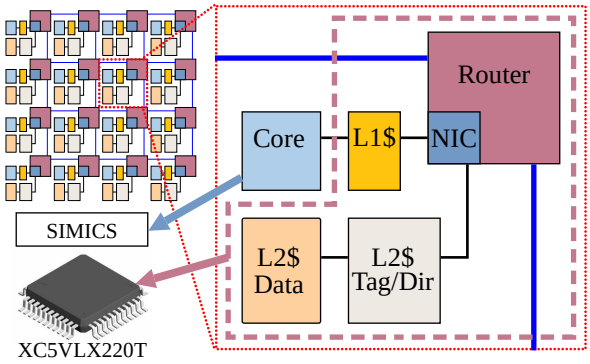

Fig. 2. Schematic diagram of 4×4 CMP connected over 2D-mesh network under simulation (L1$: L1 Cahce; L2$: L2 Cache; NIC: Network interface connector)

the others in the tile are mapped to FPGA hardware emulator. The proposed partition reconciles the confliction between the flexibility and the fidelity. With *Simics*, we can adaptively choice the desired instruction set processor. On the other hand, the concerned components in our research is designed with HDL language and implemented as the high-fidelity circuits models on the FPGA platform.

Our initial implementation is one 64-core CMP with 2D-mesh hybrid simulator. Because of the hardware limitation of single FPGA chip, it is required to partition the Caches and NoC routers into several XC5VLX220T chips. Now, our design confronts two prominent problems: First, we need to partition the whole 64-core NoC and Caches into daughter-PCBs and make the communication among these daughter-PCBs transparent to the simulation; Second, in order to realize the cycle-level accuracy, we must device the mechanism to synchronize the *Simics* simulator and the FPGA hardware emulator.

For our current prototype, considering the hardware resource limitation, especially the block RAM volume in one FPGA, the Caches in the FPGA emulator do not contain the data slice, *Simics* still access the required instructions and operands from the host PC. At present status, the Cache emulator in the hardware merely works as the timing model. Based on the preceding conditions, sixteen tiles can be mapped into one XC5VLX220T. Because we apply the 8×8 2D topology, the $2i$th- and $2i+1$th-column ($i \in \{0,1,2,3\}$) nodes are mapped into the NO.i daughter-PCB. Therefore, four daughter-PCBs are adopted to emulate the 2D multicore microprocessor and the FPGA on the mother-PCB works as the central controller, which is in charge of scheduling the working daughter-PSBs and synchronizing the hardware emulator with the host *Simics*.

With the proposed partition, there is another restriction coming from the off-chip pin capacity of FPGA. In our NoC router design, there are 76-bit signals connecting with its right-side router (38-bit inputs and 38-bit output). Totally, 624-bit signals are required to implement the connection between the ith and the $i+1$th daughter-PCBs. Obviously, the FPGA IO number limitation prohibits the direct connection implementation. We reconcile this dilemma by bartering timing for the IO bandwidth requirement. In details, the hardware simulation of one clock cycle is divided into two stages, i.e., “sim” and “sync” stages. During the “sim” period, the logic under test is given one cycle’s clock enable to complete the function of Caches and NoC routers within this cycle. However, in the “sim” stage, outputs of one FPGA do not affect the logics in its neighboring FPGAs. During the following “sync” stage, it takes multiple cycles to transfer the outputs signals of the 2×8 nodes in one FPGA to their neighbor nodes via GPIO, which locate in other FPGAs. It should be noticed that the clock signal of all logics under emulation is always disabled during the “sync” stage. Therefore, in this period, the tiles status is not affected by the data transfer between FPGAs. In our initial design, the “syn” procedure takes 18 clock cycles.

The synchronizing mechanism between *Simics* and the FPGA hardware emulator is illustrated by Fig.3. For *Simics* simulator, once it finishes the nth cycle function simulation

of all 64 cores, it stalls all processors and dispatches the Cache access requests to FPGA emulator. After receiving the acknowledge message from the hardware, *Simics* gets to know which processors get responds in this cycle, and then these processors are activated for the $n+1$th cycle's simulation. For the FPGA hardware emulator, after the $n+1$th cycle's emulation and the inter-PCB synchronization procedure, it starts the interface stage with *Simics* (H/S IF). During this procedure, the requests from software cores are used as the stimulus for the next cycle emulation, and the hardware also replies *Simics* with the completed Cache operations.

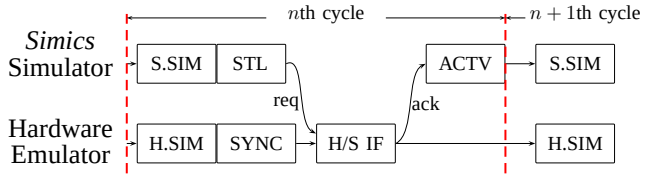

Fig. 3. The interface and cycle-accurate synchronizing mechanism between *Simics* and the hardware emulator (S.SIM: Software Simulation; H.SIM: Hardware Simulation; SYNC: Inter-FPGA synchronization; H/S IF: FPGAs process the messages from *Simics*; STL: Stall the cores with Cahce access requests; ACTV: Activate the cores with the Cache acknowledges.)

IV. Current Experimental Results

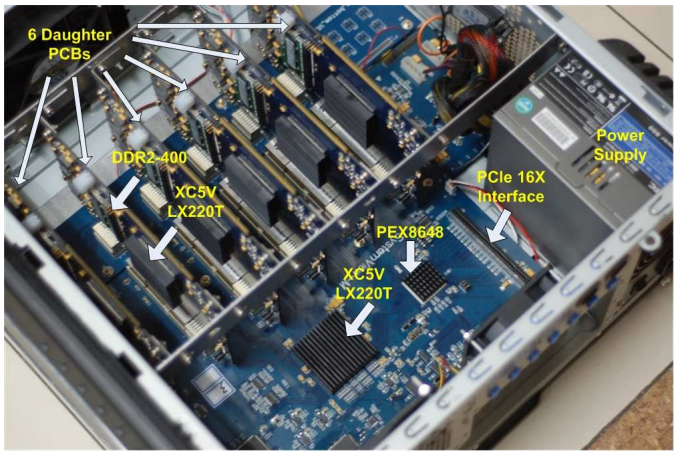

Fig. 4. Photograph of one TEAMS FPGA Rack

Now, we are developing 64-core hybrid simulator based on one FPGA rack, as shown in Fig.4. 64 tiles' Cahces and NoC routers are mapped to four daughter-PCBs. For the Cache design, MESI coherence protocol is adopted. The private L1 Cache is 1-way 512-line$\times$8-byte direct mapping organized and the shared L2 Cache applies 8-way set-associative architecture and each way is 512-line$\times$8-byte. In the first step prototyping, the data slice of L1/L2 Cache are temporarily omitted to save the on-chip memory scale. Under these conditions, L1-ICache, L1-DCache and L2-Cache consume 27,136-, 31,744-, and 503,808-bit memories, respectively. For the NoC router design, we apply 2 virtual channels for each port and the size of each virtual channel is 8×32-bit. The hardware utilization statistics of one FPGA XC5VLX220T is shown as Table I. The performance comparation between *GEMS* and our hybrid simulator based on 62-core microprocessor running two 32-thread programs is illustrated by Table II. With the same principle as *GEMS*, the simulation time of our platform is also proportional to the consumed cycles. Therefore, the timing reduction is consistent.

TABLE I
HARDWARE COST ANALYSIS OF ONE XC5VLX220T

	Router	Cache	Interface	Ultilization
Slice Registers	24992	21120	304	45%
Slice LUTs	62896	61056	608	90%
Block RAMs	0	208	0	98%

TABLE II
PERFORMANCE ANALYSIS WITH 32-THREAD APPLICATION SIMULATIONS

Test Sequence	1024x1024 LU	256k FFT	Water
GEMS(hours)	24.6	10.9	5.0
TEAMS(hours)	5.4	3.1	1.2
Timing Reduction(%)	78	72	76

V. Conclusions and Future Work

A cycle-accurate FPGA-accelerated hybrid simulator with full system fidelity is proposed in this paper. Different from previous FPGA-based emulators, our research aims at not only speeding up the simulation, but also providing the high-fidelity platform to verify and evaluate the performance of Cache and NoC designs, which are the most concerned research issues in multicore microprocessors. By working in concert with the full-system software simulator *Simics*, Cache and NoC implanted in the FPGA garner the full-system alike stimulus. Therefore, the researchers can get the high-fidelity evaluations of their designs by running commercial applications, while circumventing the construction of the entire target system, including CPU, memory, disk, IO, and network, etc. Furthermore, by leveraging the proposed inter-FPGA and inter-software/hardware synchronizing schemes, the cycle-accurate simulation is realized and the proposed platform achieves the high-level scalability. With one FPGA rack, we carry out the 64-core simulation. As compared with *GEMS*, more than 70% simulation time is saved by our testbed. Last but not least, the Cache and NoC designs verified by the proposed simulator can be fabricated directly. That is, the design gap between the traditional software simulator, such as *GEMS*, and the target circuit implementation is filled by our hybrid simulator. Our future research includes two targets: First, we are pursuing to improve the performance of one FPGA rack; Second, we will extend our platform to support the 512-core simulation by adopting multiple FPGA racks.

References

[1] P.S. Magnusson et al., "Simics: A full system simulation platform," *IEEE Trans. Comput.*, vol. 35, no. 2, pp. 50–58, Feb. 2002.
[2] Milo M. K. Martin et al., "Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset," in *Computer Architecture News*, Boston, Massachusetts, USA, Sept. 2005.
[3] N. Agarwal et al., "GARNET: A detailed on-chip network model inside a full-system simulator," in *Proceedings of ISPASS*, Boston, Massachusetts, USA, Apr. 2009.
[4] Eric S. Chung et al., "PROTOFLEX: FPGA-accelerated hybrid functional simulation," in *NSF NGS PI Workshop at IPDPS*, 2007, pp. 1–6.
[5] Z. Tan et al., "Ramp Gold: An FPGA-based architecture simulator for multiprocessors," in *In 4th Workshop on Architectural Research Prototyping at 36th ISCA*, 2009.
[6] J. Davis, C. Thacker, and C. Chang, "BEE3: Revitalizing computer architecture research," Tech. Rep., Microsoft Research, Apr. 2009.