УДК 681.3

Hardware Methods to Increase Efficiency of Algorithms for Distributed Logic Simulation

Ladyzhensky Y.V., Teslenko G.A.

Abstract

Methods for hardware accelerations are discussed. A structural implementation of a combined time synchronization algorithm is offered. A structural diagram of specialized software system is presented to support distributed logic simulation.

1. Introduction

Project verification is an important design stage in creating of electronic devices. The state-of-the-art projects of digital systems are complex and vast. They demand effective resources for verification.

Simulation is used for the functional and timing project verification. It is run for error detections on a design stage.

Verification methods with simulation are characterized by heavy timing costs and a sequential execution way of operations. A prospective direction is a parallelization of simulation algorithms. A high performance and non complex parallel or distributed software and hardware for simulation are actual scientific and technical problems.

The goal of the research is developing a hardware method to increase efficiency of distributed event-driven logic simulation. The tasks of the research include analysis of an existed hardware acceleration methods and developing a structural implementation of the distributed simulation algorithm via a specialized processor.

2. Existed hardware acceleration methods

A reconfigurable computing system is used for simulation of complex VLSI circuits design [1]. An event driven paradigm is used. The simulation system is based on FPGA-emulator, which provides large blocks of reconfigurable logic. The system hardware is supported by a compiler that can compile a Verilog HDL (hardware design language) description of a design. An advantage of this method is hardware support of event processing mechanism and a hardware emulation of design on FPGA. A drawback of this method is an execution of only one simulation task at any given time. A simultaneous execution of events is impossible. Z-state and X-state are not supported in simulations.

The PRUS (Programmable Unlimited System) technology is used for hardware accelerating of logic simulation [2]. The main idea of this method is a using massively parallel processors to process Boolean equations. HDL design converts into an equivalent set of Boolean equations. These equations are mathematically optimized, converted to binary instruction code and distributed between processors. The simulation is executed synchronously. A general register of command address is used.

This approach has following advantages. Time of result for each logic operation can be calculated in advance because processors work synchronously. Thus, a timing analysis eliminates entirely.

This method allows emulate gates and flip-flops in RAM by HES technology. Therefore, the number of gates, flip-flops and connections between them is limited only by RAM size.

A distribution of Boolean equations performs faster than a placing and routing in FPGA for the same design.

A drawback of this method is using synchronous simulations. A considerable time may be expended for synchronization that lowers an efficiency of the hardware acceleration.

3. DES processor

The acceleration of simulation can be obtained by hardware implementation of a distributed simulation algorithm [3]. The main idea is a creation of a structural model from a set of functional units. This provides equivalent mapping algorithms for a computational processes and data processing operations. The algorithm is realized by a data movement from input to output of a circuit.

The combine synchronization algorithm is discussed for hardware implementation. A software implementation of this algorithm is shown a good potential for parallel digital simulation. A speedup for the combined algorithm is twice more the speedups of an optimistic or conservative synchronization algorithm [5].

A structural implementation of this algorithm is shown on fig.1.

The main advantage of this algorithm is a possibility of switching of behavior from optimistic to conservative and vice-versa [5,6]. A user can configure a behavior of logical processes both automatically and manually. An optimistic logical process will switch if it rollbacks too often, and a conservative logical process will switch if it blocks too often.

It is a specialized processor for a distributed logical simulation, DESP (Distributed Event-driven Simulation Processor). This device can be implemented using FPGA technology.

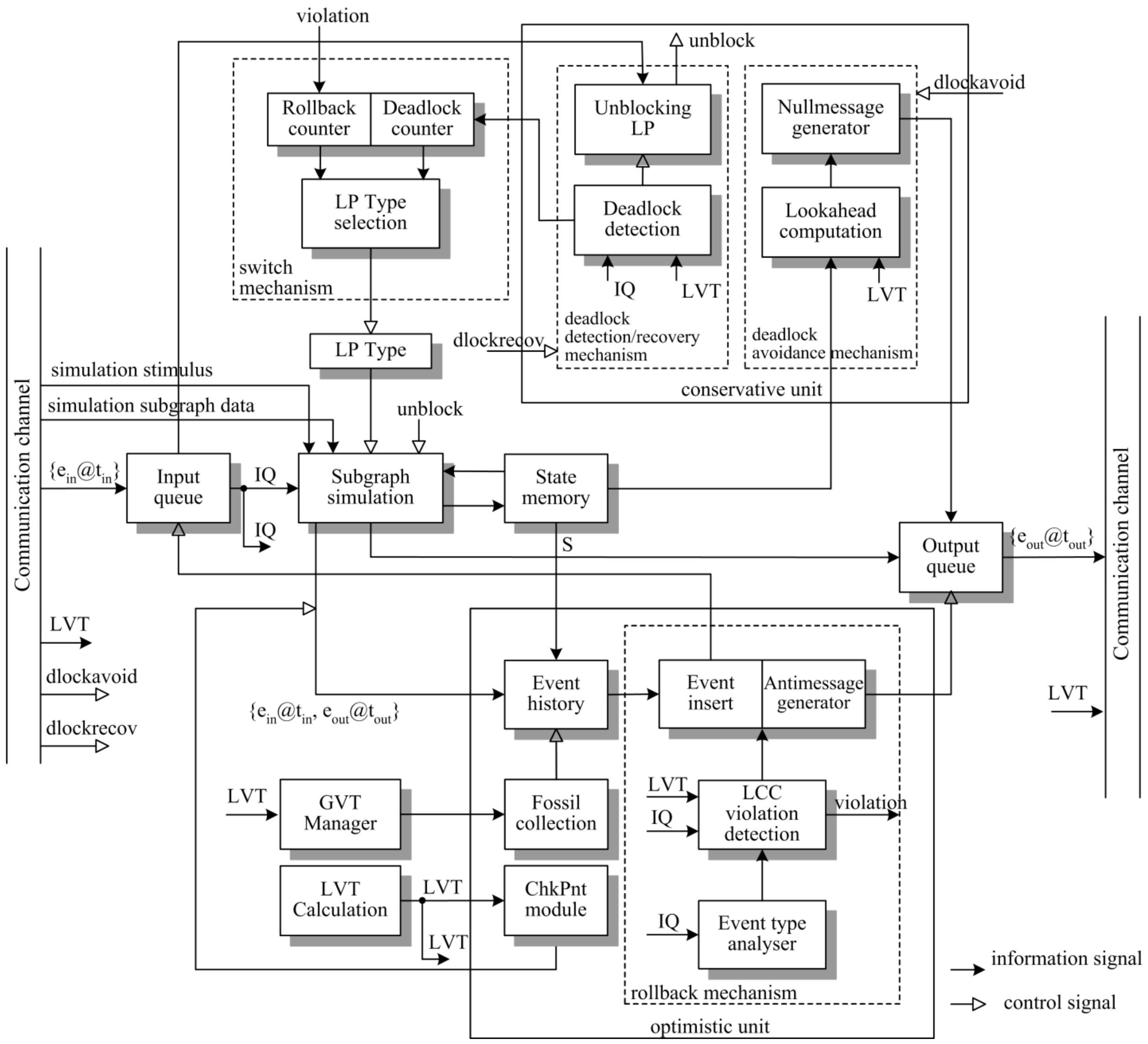

Figure 1 – Structure of a specialized processor for a distributed logic simulation

Let $\{e_{in}@t_{in}\}$ be an input event e_{in} to a logical process (LP) at the time of t_{in} , $\{e_{out}@t_{out}\}$ be an output event sent to others LPs at the time of t_{out}.

DESP contains an input queue block, an output queue block, a mechanism of switching types of synchronization, a subgraph simulation block, an optimistic and a conservative synchronization unit, a state memory block, a local virtual time (LVT) calculation block, a global virtual memory (GVT) manager.

A deadlock detection/recovery mechanism and a deadlock avoidance mechanism are implemented by a conservative synchronization unit. The switching of mechanisms is carry out by user. The “dlockrecov” and the “dlockavoid” control signals select corresponding mechanism.

The optimistic synchronization unit consists of a rollback mechanism, an event history block, a fossil collection block, a checkpoint module ChkPnt.

The subgraph simulation block performs simulation of a specified section of circuit.

LVT calculation block calculates a local virtual time. The GVT time determines as a minimum value of LVT among all LPs by GVT manager.

4. Specialized software system architecture

The simulation system consists of the specialized workstations. They are joined in a local area network or using a web-interface. Each workstation contains the DESP which can be implemented as an expansion card for a personal computer (fig.2).

Special software is needed for operating of the simulation system. The architecture of the software system is shown on fig. 3. The simulation process is performed in the following way.

First, a HDL project is converted to a graph netlist by an input data subsytem. Next, the obtained graph is cut on separate sections by netlist cutting subsystem. Each section is assigned to a single logical process. The DESP corresponds to a logical process in a simulation algorithm.

A software data communication subsystem is presented. It transforms a simulated section to a form which is necessary to load in the DESP. The “simulation data subgraph” and the “simulation stimulus” information signals (fig.1) loads simulated data to DESP. During simulation the data communication subsystem provides data exchanges with DESP and other workstations in the network.

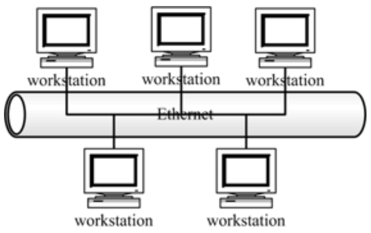

Figure 2 – The net of specialized workstations for distributed logic simulation

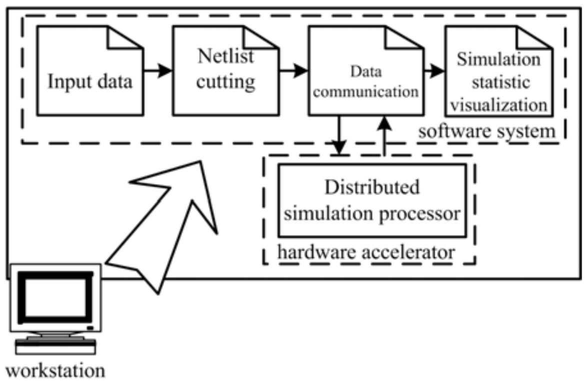

Figure 3 – Specialized software system architecture

Statistics during simulation processes are visualized by a simulation statistic visualization subsystem.

5. Conclusion

An offered structural implementation of the distributed synchronization algorithm via a specialized processor is a scientific novelty. The algorithm with dynamic synchronization of calculations is used.

The dynamic synchronization protocol and its hardware implementation can increase efficiency of distributed logic simulation.

Following topics are interesting for the further research:

- a switching criterion between the conservative unit and optimistic unit activity;
- analysis of a various simulated scheme implementation as a subgraph simulation block;
- a simulation model developing to verify a DESP activity correctness and a performance evaluation.

An offered method has next advantages as compared with a known technique [1,2]:

- an asynchronous distributed event-driven simulation may be considerably reduce a simulation time;
- a probability of using with an existed software logic simulator.

Reference

[1] Jerry Bauer, Michael Bershteyn, Ian Kaplan, Paul Vyedin. A Reconfigurable logic machine for fast event-driven simulation // Proc. 35th Design Automation Conference, 1998

[2] Stanley Hyduke, Eugene Kamenuka, Irina Pobezhenko, Olga Melnikova. Emulation processor network for gate-level digital systems // Proceeding of IEEE East-West Design & Test Workshop, 2005, pp.257-260.

[3] Ладыженский Ю.В., Тесленко Г.А. Аппаратный метод повышения эффективности алгоритмов распределенного логического моделирования цифровых систем. // Наукові праці Донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація». Випуск 106 – Донецьк: ДонНТУ, 2006. – 220с. – С.77-81.

[4] Динамическая теория информации. Основы и приложения/ В.П. Боюн –Киев: Ин-т кибернетики им. В.М. Глушкова НАН Украины, 2001. – 326с.

[5] C.J.R. Shi, D.Lungeanu. Distributed simulation of VLSI circuits via lookahead-free self-adaptive optimistic and conservative synchronization. In Proc. ICAAD, pages 500-504, Nov 1999.

[6] D.Lungeanu and C.-J.R. Shi. Parallel and distributed vhdl simulation. In Proc. DATE, pages 658-662, March 2000.