International Conference on Advance Studies in Engineering and Sciences
Sri Satya Sai University of Technology and Medical Sciences , Sehore (M.P.) ICASES-17
2nd December 2017, www.conferenceworld.in ISBN: 978-93-86171-83-2

Design and Development of Programming Model for Reconfigurable Computing using Intelligent Techniques

Swaroopa Duli[1] ,K. Bharath Reddy [2]

[1,2]Computer Science and Engineering, Sri Satya Sai
University of Technology and Medical Sciences, Sehore, (India)

ABSTRACT

Intelligent computing systems comprising microprocessor cores, memory and reconfigurable user-programmable logic represent a promising technology which is well-suited for applications such as digital signal, cryptography and encryption, etc. These applications employ frequently recursive algorithms which are particularly appropriate when the underlying problem is defined in recursive terms and it is difficult to reformulate it as an iterative procedure. It is known, however, that hardware description languages (such as VHDL) as well as system-level specification languages (such as Handel-C) that are usually employed for specifying the required functionality of reconfigurable systems do not provide a direct support for recursion.The Remote Xilinx environment is used to provide remote access to the Xilinx Integrated Software Environment (ISE). The main aim of the research presented is to highlight how users can access FPGA design resources from anywhere in combination with a potential remote FPGA lab. The architecture of the cloud based platform is described with a load analysis for the server. The remote environment is developed on the Ubuntu operating system by using Python and Hypertext Preprocessor (PHP) scripting languages. The open source Apache server is used for running Xilinx environment on a server and open source analysis tools are used to perform server load analysis for running Xilinx environment on the server system.

Keywords: Cryptography, Digital signal, Xilinx Integrated Software Environment, PHP, Apache server

I.INTRODUCTION

The changing technology landscape and fast evolution of application standards make it imperative for a design to be adaptable. Adaptability is a mandatory part of some major research initiatives such as cognitive radio, where the radio transceiver implementation takes cognition of its surrounding environment. Adaptability can be present in the device, in the circuit, in the micro architecture, or even in the runtime software layer or among all of these. In this study, we survey reconfigurable processors, which provide a complete spectrum of adaptability in the processor micro architecture. The survey is primarily intended for developers and users of adaptive digital systems. Reconfigurable computing is rapidly establishing itself as a major discipline that covers various subjects of learning, including both computing science and electronic engineering. Reconfigurable computing involves the use of reconfigurable devices, such as field programmable gate arrays (FPGAs), for computing

purposes. Reconfigurable computing is also known as configurablecomputing or custom computing, since many of the design techniques can be seen as customising a computational fabric for specific applications.

Reconfigurable computing systems often have impressive performance. Consider, as an example, the point multiplication operation in elliptic curve cryptography. For a key size of 270 bits, it has been reported that a point multiplication can be computed in 0.36 ms with a reconfigurable computing design implemented in an XC2V6000 FPGA at 66MHz. In contrast, an optimized software implementation requires 196.71 ms on a dual-xeon computer at 2.6 GHz; so the reconfigurable computing design is more than 540 times faster, while its clock speed is almost 40 times slower than the Xeon processors. This example illustrates a hardware design implemented on a reconfigurable computing platform. We regard such implementations as a subset of reconfigurable computing, which in general can involve the use of runtime reconfiguration and soft processors.

Since the introduction of programmable logic devices, the domain of reconfigurable computing has steadily gained acceptance first as the prototyping platform and second as the computing platform of choice. The transition of reconfigurable computing devices from the role of prototyping to computing is what presented in 2001 by Hartenstein in his study "A decade of reconfigurable computing: a visionary retrospective". It is mentioned that reconfigurable platforms are able to bridge the gap between processors and Application-Specific Integrated Circuits (ASICs) in terms of flexibility and performance. Since this work, notable research has been done in accelerator design (application-specific processors), multicore homogeneous and heterogeneous System-on-Chip (SoC) architectures, and smooth high-level synthesis of various kinds of computing devices. From the application domain, an ongoing blend between high-performance computing, general-purpose computing, and embedded computing is noticeable. This also prompts development of interesting architectures which combine a programmable processor with a reconfigurable device or place a processor as macro block inside a reconfigurable platform.

II.STATEMENT OF THE PROBLEM

The main purpose is to design and develop a programming model for reconfigurable computing using intelligent technique and examine the various problems associated into it.

III. DELIMITATIONS

1. The study is delimited to the reconfigurable computing.
2. The study is also delimited to programming models for reconfigurable computing.

IV. LIMITATION

The facts discussed in this study will be based entirely on the responses to the questionnaire and modeling of reconfigurable computing therefore, ascertaining the genuineness of the responses will identify as the limitation of the study.

V. HYPOTHESIS

- There is no significant difference between present and past reconfigurable computing.
- There is no significant difference between programming models for reconfigurable computing.

VI. DEFINATIONS AND EXPLANATION TERMS

6.1 Reconfigurable Computing

Reconfigurable computing is a computer architecture combining some of the flexibility of software with the high performance of hardware by processing with very flexible high speed computing fabrics like field-programmable gate arrays (FPGAs). The principal difference when compared to using ordinary microprocessors is the ability to make substantial changes to the datapath itself in addition to the control flow. On the other hand, the main difference with custom hardware, i.e. application-specific integrated circuits (ASICs) is the possibility to adapt the hardware during runtime by "loading" a new circuit on the reconfigurable fabric.

The concept of reconfigurable computing has existed since the 1960s, when Gerald Estrin's paper proposed the concept of a computer made of a standard processor and an array of "reconfigurable" hardware.The main processor would control the behavior of the reconfigurable hardware. The latter would then be tailored to perform a specific task, such as image processing or pattern matching, as quickly as a dedicated piece of hardware. Once the task was done, the hardware could be adjusted to do some other task. This resulted in a hybrid computer structure combining the flexibility of software with the speed of hardware.

6.2 Programming Model

A **Programming Model** refers to the style of programming where execution is invoked by making what appear to be library calls. Examples include the POSIX Threads library and Hadoop's MapReduce. In both cases, the execution model is different from that of the base language in which the code is written. For example, the C programming language has no execution model for thread behavior. But thread behavior can be invoked from C syntax, by making, what appears to be, a call to a normal C library.

What distinguishes a programming model from a normal library is that the behavior of the call cannot be understood in terms of the language the program is written in. The behavior of calls to the POSIX thread library cannot be understood in terms of the C language. The reason is that the call invokes an execution model that is different from the execution model of the language. This invocation of an outside execution model is the defining characteristic of a programming model.

6.3 Intelligent Techniques

Intelligent computing systems (ICS) comprising microprocessor cores, memory and reconfigurable user-programmable logic (usually, field-programmable gate arrays - FPGA) represent a promising technology which is well-suited for applications that require direct bit manipulations and are appropriate to parallel implementations, such as digital signal and image processing, cryptography and encryption, etc. In such ICS, the

reconfigurable part is periodically modified in response to dynamic application requirements. Creation and validation of scalable, distributed ICS architectures requires a closely coordinated hardware and software development effort in the areas of FPGA-based accelerators, runtime control libraries and algorithm mapping [1]. This paper focuses on the algorithm mapping.

VII.SIGNIFICANCE OF THE STUDY

The study consists of the followings:-

•System-level architectures for reconfigurable computing in either real-time or non real-time systems

•Heterogeneous architectures that integrate a mix of coarse, fine, special purpose, and general purpose hardware

•Implications and effects of new technologies, including nanotechnology on reconfigurable computing (and vice versa)

VIII. OBJECTIVES

- To study the reconfigurable computing.
- Architectures for high performance and/or low power configurable computing
- New spatial architectures with immense parallelism but different basic components than FPGAs
- Security enhancements for reconfigurable computing
- To measure the various issues and challenges involved into it.

IX.REVIEW OF RELATED LITERATURE

Herbordt et al (2008) Field-programmable gate arrays are widely considered as accelerators for compute-intensive applications. A critical phase of FPGA application development is finding and mapping to the appropriate computing model. FPGA computing enables models with highly flexible fine-grained parallelism and associative operations such as broadcast and collective response. Several case studies demonstrate the effectiveness of using these computing models in developing FPGA applications for molecular modeling.

Cray (2009) stated with multiple embedded processors and hardware engines (collectively referred to in this paper as computing elements, or CEs) interacting across multiple FPGAs, there is a need for a networking infrastructure and a distributed programming model. Most multi-FPGA systems have distributed memory (usually, each FPGA has its own external memory), and inside each FPGA there might be multiple CEs, each with their own local memory; hence, a programming model such as MPI seems adequate. With the appearance of tightly-coupled FPGAs and X86 processors sharing the main system memory, TMD-MPI has been extended to include communication with X86 processors, marking TMD-MPI's entrance to the HPRC world.

Chang (2010) (HERC) is a machine with supercomputer-level performance configured on a per-problem basis to match the structure of the algorithm and data flow of a computing task. In such a machine, all data and control paths; memory ports and controllers, and communication channels and controllers, are customizable to match a particular application's needs.

Tilak (2012) A reconfigurable computing environment is characterized by the ability of underlying hardware architectures or devices to rapidly alter (often on the fly) the functionalities of their components and the interconnection between them to suit the problem at hand. The area has a rich theoretical tradition and wide practical applicability. There are several commercially available reconfigurable platforms (FPGAs and coarse-grained devices) and many modern applications (including embedded systems and HPC) use reconfigurable subsystems. An appropriate mix of theoretical foundations and practical considerations, including algorithms architectures, applications, technologies and tools, is essential to fully exploit the possibilities offered by reconfigurable computing. The Reconfigurable Architectures Workshop aims to provide a forum for creative and productive interaction for researchers and practitioners in the area.

Singh et al (2013) Our general, parallel-programming paradigm for reconfigurable computing is an extension of the standard multithreaded programming model in which multiple software threads execute in the context of a common process, relying on thread library and OS support for inter thread communication and synchronization.4 We described the proposed extension with the help of a simple motivational example.

Patel et al (2014) Portability in a typical High-Performance Computer (HPC) is achieved by using standards-based operating systems and software layers (Middleware) that abstract machine-specific hardware from the software application. The UNIX operating system (and its variants) and the MPI [The MPI Forum 1993] library are examples of de-facto standards that enabled portable parallel applications across HPCs. For HPRC designs to be truly portable, these concepts must be extended to the FPGAs, as they are now part of the application.

Agarwal & George et al (2015) presented a parallel-programming model and a communication library for scalable, heterogeneous, reconfigurable systems. The multilevel PGAS model introduced in this paper is able to capture key characteristics of RC systems, such as different levels of memory hierarchy and differences in the execution model of heterogeneous devices present in the system. The existence of such a programming model will enable development of scalable, parallel applications for reconfigurable HPC systems.

XI.PROCEDURE

This Chapter deals with the procedure adopted for algorithm and selection of subjects, non measures, collection of data and the statistical techniques led for analysis of data.

10.1 Selection of Subjects

The study will discuss the reconfigurable computing & evaluate the stepping stones to a model & various algorithms and examine the various issues and challenges involved it. It involves

- New spatial architectures with immense parallelism but different basic components than FPGAs.
- Security enhancements for reconfigurable computing.
- New ideas of Intelligent techniques.
- Usage of Xilinx ISE .

10.2 Criterion Measure

Studies shall be descriptive in nature. Different data shall be prepared to ascertain the trend and impact. Data would be collected through different primary and secondary sources like:

PRIMARY SOURCES:-

1. Questionnaire
2. Interviews

SECONDARY SOURCES:-

1. Publications
2. Internet
3. Journals

REFERENCES

[1.] N. B. Asadi, C. W. Fletcher, G. Gibeling, E. N. Glass, K. Sachs, D. Burke, Z. Zhou, J. Wawrzynek, W. H. Wong, G. P. Nolan, "Para Learn: A Massively Parallel, Scalable System for Learning Interaction Networks on FPGAs," Proceedings of the 24th ACM International Conference on Supercomputing, Tsukuba, Ibaraki, Japan, 2010, Pages: 83-94.

[2.] R. Koenig, L. Bauer, T. Stripf et al., "Kahrisma: a novel hypermorphic reconfigurable-instruction-set multi-grained-array architecture," in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, pp. 819-824, 2010.

[3.] V. Govindaraju, H. Chen-Han, T. Nowatzki et al., "DySER: unifying functionality and parallelism specialization for energy-efficient computing," IEEE Micro, vol. 32, no. 5, pp. 38-51, 2012.

[4.] Wang, D. 2008. An Automated Flow to Generate Hardware Computing Nodes from C for an FPGA-Based MPI Computing Network. Engineering Science Undergraduate Thesis, University of Toronto.

[5.] Bachrach, J., Vo, H., Richards, B., Lee, Y., Waterman, A., Avizienis, R., Wawrzynek, J., and Asanovi, K., "Chisel: Constructing Hardware in a Scala Embedded Language," Proceedings of 2012 Design Automation Conference (DAC), 2012, San Francisco, pp. 1216-1225.

[6.] Lazzaro, J., Wawrzynek, J., "A Tilt Filter in a Servo Loop", Proceedings of the 133rd Audio Engineering Society (AES), 2012, San Francisco, CA.

[7.] McKeever, S.W., Luk, W., and Derbyshire, A.: 'Compiling hardware descriptions with relative placement information for parametrised libraries', Lect. Notes Comput. Sci., 2002, 2517.

[8.] F. Chen, T., Raghavan, R., Dale, J. N., and Iwata, E., Cell broadband engine architecture and its first implementation: a performance view. IBM Journal of Research and Development, 2007, 51, 5, 559-572.

[9.] K. Shahzad, A. Khalid, Z. E. RAa kossy, G. Paul, and A. Chattopadhyay, "CoARX: a coprocessor for ARX-based cryptographic algorithms," in Proceedings of the ACM/IEEE Design Automation Conference, 2013.

[10.] Rissa, T., Luk, W., and Cheung, P.Y.K.: 'Automated combination of simulation and hardware prototyping'. Proc. Int. Conf. on Engineering of Reconfigurable Systems and Algorithms (CSREA Press, 2004).

[11.] Cheng S., Lin, M., Liu, H., Scott, S., Wawrzynek, J., "Exploiting Memory-Level Parallelism in Reconfigurable Accelerators," 2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines (FCCM 2012), 2012, Toronto, Canada, pp. 157-160.