Swarm and Evolutionary Computation 19 (2014) 68–81

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

Comparative study of system on chip based solution for floating and fixed point differential evolution algorithm

Rangababu Peesapati, Kiran Kumar Anumandla, Shravan Kudikala, Samrat L. Sabat *

School of Physics, University of Hyderabad, Hyderabad 500046, India

ARTICLE INFO

Article history:
Received 22 November 2013
Received in revised form
2 May 2014
Accepted 30 June 2014
Available online 8 July 2014

Keywords:
Differential evolution
Hardware accelerator
Floating point unit
System on chip

ABSTRACT

This paper presents performance study of scalable hardware accelerator for fixed and floating point differential evolution (DE) algorithms in field programmable gate array (FPGA) using programmable system on chip (PSoC) approach. The hardware intellectual property (IP) of the DE is interfaced as a Slave Unit (SU) as well as an Auxiliary Processor Unit (APU) with the PowerPC440 processor based System on Chip (SoC) platform on Xilinx Virtex-5 FPGA. Six numerical benchmark functions are optimized to validate the IP and its interface to processor. From the experimental results, it is observed that (i) Both SU and APU interfaces of fixed and float DE IPs have shown similar acceleration because of less communication overhead. (ii) Floating point DE has higher resource utilization compared to fixed point DE. (iii) Both interfaces of fixed and float DE SoC systems have shown similar power consumption. (iii) Finally as a case study, an Infinite Impulse Response (IIR) based system identification task with second and fourth order plant transfer functions is implemented on PSoC using the fixed and float DE IP cores with fabric co-processor bus (FCB) interface using APU controller. The experimental results reveal that the acceleration factor and resources utilization increases with the increase in problem complexity.

1. Introduction

Evolutionary algorithms (EAs) are extensively used for solving many diverse domains of science and real-time engineering applications to find an optimum solution of multimodal functions [1,2]. In the literature, many algorithms like Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Differential Evolution (DE) are reported to solve multimodal optimization problems. Among these, DE algorithm is popular due to its faster convergence and less computation complexity; due to this it has been successfully applied in different applications like parameter extraction [3], image segmentation [4], feature subset selection [5] and design of IIR filters [6,7]. In general, the evolutionary algorithms are implemented in the embedded processors for real time applications. Due to sequential execution nature of the processor, it takes longer execution time than the target time to complete a task. This can be overcome by implementing the algorithm in field programmable gate array (FPGA), because it supports concurrent execution. Due to this, the engineering communities are attracted to implement this algorithm in hardware [8]. EAs like GA, PSO and DE are implemented on a high end desktop computer/processors to solve the optimization problems. DE algorithm becomes computationally intensive with increase in the dimension/complexity of the problem thus increases the execution time for solving a task. This limits its real-time applications in embedded processors. So there is an increasing demand for its hardware realization. DE algorithm has been implemented in general purpose processors for an off-line simulation in desktop environment [9]. In modern era, the applications like system identification and evolvable hardware require the implementation of evolutionary algorithms in an embedded processor. Execution speed and resource consumption are the main performance indicators for an embedded system that depends on the implementation strategy and the platform. Thus, the main focus of this work is to compare different implementation strategies of DE algorithm in a FPGA.

For high accuracy and resolution, floating point arithmetic is preferred over fixed point arithmetic. Similarly for high bandwidth data transfer applications Auxiliary Processor Unit (APU) interfacing is preferred over Slave Unit (SU) interface. Recently the authors have reported the performance of floating point DE algorithm implemented in a PowerPC440 (PPC440) based system on chip platform using Slave Unit interface [10]. The performance of fixed point DE algorithm implemented in the same platform using APU interface is reported in [11]. But there is no cross comparison of performances between the fixed and floating point DE algorithms using different interfaces. The objective of this

* Corresponding author.
E-mail addresses: kiran.anumandla@gmail.com (K.K. Anumandla), slssp@uohyd.ernet.in (S.L. Sabat).

http://dx.doi.org/10.1016/j.swevo.2014.06.005
2210-6502/

paper is to compare the performance of both fixed and floating point DE algorithms with both APU and SU interfacing techniques while optimizing a set of test bench functions. In the first technique, the IP is interfaced using an Auxiliary Processor Unit (APU) with the PPC440 embedded processor and in other technique, it is interfaced as a Slave Unit (SU) with the shared local bus of embedded processor. The hardware accelerators are scalable in terms of the population size, number of generations and dimension, that can be modified by the user through the embedded processor. All the modules of DE including fitness function evaluation are implemented in the hardware, and used as a dedicated hardware accelerator to reduce the bus transaction time [10]. The objective of this work is to evaluate the SoC system performance in terms of execution time, resource utilization and power consumption while solving different optimization problems. Initially the acceleration is evaluated by solving numerical test bench functions and later the DE IP is used for solving a second and a fourth order IIR based system identification problem and implemented in FPGA. The optimized IIR filter coefficients are obtained using the DE hardware accelerator.

The rest of the paper is described as follows. Section 2 presents the literature survey about the hardware implementation of evolutionary algorithms. Hardware architecture of DE algorithm is described in Section 3. Programmable system on chip platform for DE accelerator with two types of bus interface is explained in Section 4. Section 5 presents the experimental setup. Section 6 describes the results and analysis of DE accelerators. Section 7 presents the IIR system identification task as a case study for real world application of DE followed by conclusions in Section 8.

2. Literature survey

In the recent years, evolutionary algorithms have been implemented in hardware. Ascia et al. discussed the importance of evolutionary algorithms in space exploration using GA and a parametrized SoC platform on two different processors was implemented with strict power consumption and performance constraints [12]. The two processors are the following: (i) Reduced Instruction Set Computing (RISC) R3000 Microprocessor without Interlocked Pipeline Stages (MIPS) processor and (ii) Very Large Instruction Word (VLIW) processor and a memory subsystem with two cache levels. It resulted in pareto-optimal configuration for optimum power consumption and execution time of a specific embedded application.

A Complete Hardware Evolution (CHE) of Genetic Algorithm (GA) was implemented on a single FPGA to evaluate single variable fitness functions [14], and reported that the execution speed is improved significantly as compared to its software implementation. A comprehensive review of the literature related to FPGA implementation of EAs is tabulated in Table 1. However, this implementation has no provision to configure the GA parameters like mutation, crossover rates, population size and number of generations, and it cannot be directly interfaced to higher dimensional fitness functions. To overcome the above-mentioned drawbacks in [14] a customized IP of GA was implemented in the Xilinx FPGA and attained the speed enhancement up to five times [13]. This core can be interfaced to any appropriate application-specific fitness evaluation module and the GA parameters are programmable. Along with GA, Particle Swarm Optimization (PSO) algorithm was implemented in the FPGA using software/hardware co-design principle and this was based on modular design architecture [15]. A particle updating accelerator module was implemented in the hardware and fitness evaluation was performed either on the soft-core processor or FPGA. Experimental results of [15] revealed that hardware execution is approximately 20 × faster when compared to its equivalent software implementation on NIOS-II processor.

Recently, the shortcomings of the software based PSO were addressed by Tewolde et al. [17]. A modular, flexible and reusable hardware architecture was reported and it accelerates the execution performance by overcoming the drawbacks of software implementation of the PSO algorithm on Freescale μ C and Xilinx MicroBlaze soft processor core. As a result, hardware PSO accelerated the performance with average speedups ranging from 359 × to 653 × over the Freescale processor, and achieved an acceleration of 37–52 × over MicroBlaze processor, running at the same clock frequency. In [24] the particle positions were updated on the FPGA logic while the fitness function was evaluated on Nios-II embedded processor. The modular design enhances the flexibility to modify the fitness function using software. Due to this flexibility, various embedded applications can be developed simply by changing the objective function. Although this approach has more flexibility, improvement in the execution speed is less. Later, an architecture for PSO algorithm was proposed in which an on-chip multiprocessing based on system-on-a-programmable-chip (SOPC) methodology is used to realize the PSO algorithm [16]. Later a hardware accelerator for PSO algorithm was reported [18] and validated its

Table 1
Review of existing literature on FPGA implementation of evolutionary algorithms.

Work	Algorithm	Processor (Hz)	IP Freq (Max Freq) (MHz)	FPU used	Speedup	Target board
[13]	GA	PowerPC (200)	50 (50)	No	5.16 ×	Xilinx Virtex-II Pro
[14]	GA	PC	– (50)	No	–	Xilinx SP3E
[15]	PSO	Nios-II (50)	50 (50)	No	20 ×	Altera DE2-70
[16]	PSO	4 Nios-II (50)	50 (76.3)	No	98 ×	Altera Stratix
[17]	PSO	Freescale (25)	25 (–)	No	359–653 ×	MC9S12DP256B
		MicroBlaze (25)	25 (42.5) & 25 (29.8)		37–52 ×	Xilinx Virtex-II Pro Xilinx SP3E
[18]	PSO	MicroBlaze (200)	– (233)	No	18–135 ×	Xilinx Virtex-6
[19]	GA	16 EM64T CPU (3.2G)	8–15 (–)	Yes	1.3–3 ×	Altera Cyclone
[9]	GA	CPU (2.7G)	190 (175)	Yes	7–116 ×	Xilinx Virtex-5
		GPU (Nvidia Quadro FX) (450M)	110 (100)		7.3–12.3 ×	
[20]	PSO	CPU(1.6G) with MATLAB	50 (94)	Yes	78–127 ×	Xilinx Virtex-5
[21]	PSO	MicroBlaze (50M)	50 (99)	Yes	6490–13,820 ×	Xilinx Virtex-5
		CPU (1.6G) with MATLAB			3.6–4.2 ×	
[22]	PSO	MicroBlaze (50M)	40 (40)	Yes	6465–13,888 ×	Xilinx Virtex-5
		CPU (1.6G)			1.4–3.1 ×	
[11]	DE	PPC440 (200M)	33 (65)	No	80–150 ×	Xilinx Virtex-5
[10]	DE	PPC440 (200M)	50 (120)	Yes	200 ×	Xilinx Virtex-5
[23]	FA	MicroBlaze (100M)	100 (130)	Yes	1156 ×	Xilinx Virtex-5