A Fully Data-Driven Reconfigurable Architecture with Very Coarse-Grained Execution Units

Yuzhong Jiao[1,2], Xin'an Wang[1,2], and Xuewen Ni[2]

[1] Reconfiguring DSP Research Center, Shenzhen Graduate School of Peking University
Shenzhen 518055, P.R. China
[2] Element of Microelectronics, Peking University
Beijing 100871, P.R. China
jiaoyz04829@szcie.pku.edu.cn, wangxa@szpku.edu.cn,
nxw@pku.edu.cn

Abstract. There is a clear turning point in the development history of reconfigurable architectures. Larger execution units (EU) used to be adopted in special domain applications to improve the cost performance of programmable architectures. However, after the granularity of EUs came up to the level of arithmetic logic unit (ALU) and multiplication accumulation unit (MAC), the trend almost stopped. At present, a great number of reconfigurable architectures make use of simple Von-Neumann-architecture processing elements (PE) with such EUs as ALU and MAC. Actually, today's application algorithms are far different from the previous counterparts with the development over the last decades. Larger operation units can be extracted from common application algorithms. Without the coherent enhancement of EUs, it is difficult for reconfigurable architectures to replace the application specific integrated circuits (ASIC) used for most of current high-throughput applications. In order to further improve the performance/cost ratio, this paper presents a novel architecture with very-coarse-grained EUs and fully-data-driven mechanism.

Keywords: Reconfigurable architecture, Processing element (PE), Execution unit (EU), Very-coarse-grained, Fully-data-driven.

1 Introduction

An important force driving the electronics industry is the growing demand for increasingly complex multimedia computing and baseband processing devices [1]. Based on the natural parallelism of the algorithms in the devices [2], [3], [4] and [5], a great number of reconfigurable architectures have been proposed by the industry and academia to fulfill such a demand over the last decade [6], [7], [8] and [9]. These architectures combine the advantages of the efficiency (in terms of area, power and performance) of ASICs and the flexibility of the programmability of conventional programmable processors with Von Neumann architecture. The granularities of the reconfigurable architectures have evolved from gate-level circuits to more coarse-grained functional blocks [10] and [11] or even program-controlled PEs [12], [13] and

[14]. With the advancement of the granularities of basic PEs, the computation capacities of reconfigurable architectures have increased significantly.

In this paper, we propose a novel reconfigurable architecture with very-coarse-grained EUs. Both the control circuits of PEs and the interconnection network are very simple. Weakly programmability is provided to realize enough functions in some special domain applications. In order to obtain a high-cost-performance reconfigurable architecture, data-driven concept is used not only in interconnection networks but also in each PE.

The paper is organized as follows: in section 2 the adaptability of reconfigurable architectures to current high-throughput applications is analyzed; in section 3 the novel architecture with fully-data-driven mechanism and very-coarse-grained EUs is presented; section 4 gives the configurations of the proposed reconfigurable architecture; section 5 presents the design flow of special domain applications by using the new architecture; the performance analyses are shown in section 6 and finally the conclusion is given.

2 Design Consideration

Although lots of reconfigurable architectures have been designed and have revealed the potential high computing performance, they have achieved rather limited success. The main reason is that their performance/cost ratios are not good enough for most high-throughput custom devices. Thus, reconfigurable architectures are limited in the narrow fields such as the signal processing of radar and base station, and the prototyping development of high-throughput systems [11], [15] and [16].

In fact, ASICs or traditional system-on-a-chips (SOC) are used more widespread in the computation-intensive signal processing systems. However, ASIC approaches often have high nonrecurring engineering (NRE) costs, and take considerable time, even months, to fabricate. And with the development of microelectronics technologies, the NRE cost is increasing. On the other hand, the lifetime of electronic products has shortened. Therefore, programmability must be emphasized in high-performance architectures to reduce risks and to increase profits.

A trend is easily seen that new reconfigurable architectures will be more flexible to program than their previous counterparts. That is, the programmability of reconfigurable architectures continues increase. Since the advancement of technologies enables more complex logic blocks to be integrated on one chip, the basic PEs in reconfigurable architectures vary from simple controllable logic blocks to configurable ALUs and MACs, and further to smaller RISC-type programmable DSP cores. Now, a great number of reconfigurable architectures use one array structure and contain hundreds of simple DSP cores [9]. However, the basic EUs (like addition and multiplier, or ALU and MAC) in PEs seem to have been holding a relatively steady size and do not change as quickly as PEs. Maybe this can account for the limited successes of reconfigurable architectures.

The primary aim of reconfigurable architectures is to exploit the concurrency of algorithms and to reduce control overhead for conventional microprocessors, and configuration overhead for FPGAs. Originally, the larger execution units were designed

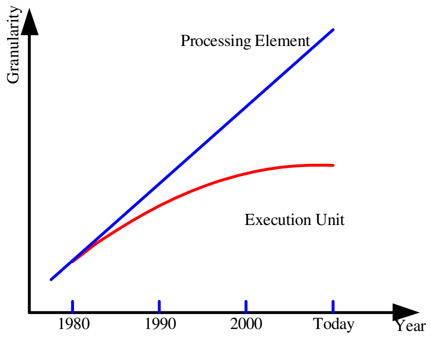

Fig. 1. The trends of PEs and EUs

on the basis of the data flow graphs (DFG) of applications, or optimized for some special blocks [17], [18] and [19]. However, such attempts almost came to a standstill after the emergence of ALUs and MACs [20] and [21]. The evolutionary trends of PEs and EUs are shown in Fig. 1. PEs varying from configurable coarse-grained EUs to DSP cores implies the return of control overhead. Although the advancements in technology make it possible to integrate more PEs, the reconfigurable architectures do not offer satisfactory performance/cost ratios to replace ASICs in high-throughput applications.

The problem partly stems from the knowledge mismatch between computing architectures and application algorithms. More attentions are given to the exploration of reconfigurable architectures in order to get the best solution in multiple-dimension design space. At the same time, the aboriginal viewpoint of disassembling applications algorithms is still held that all application systems can be ultimately divided into some basic operation units like adder, multiplier, multiplexer and logic shift. Therefore, almost all reconfigurable architectures use these basic units to process algorithms. However, the features of application systems have changed. The following are the main differences between current and former applications.

- Current innovating points mainly locate at the combinations of various algorithms.
- One single mainstream algorithm may be adopted in multiple different standards and specifications. For instance, orthogonal frequency division multiplexing technique (OFDM) is used in more than 10 communication standards and specifications.
- Mainstream algorithms are presenting more advantages, which results in just few algorithms existing in similar application systems.
- With the fast development of global economy, hundreds of million persons can benefit from one mainstream technique.

Thus it can be seen that the basic functional blocks of current application systems are much larger than those of former algorithms. Larger operation units can be extracted from common application algorithms.

We suppose that ASICs are composed of tightly coupled basic operation units like the assemblage of one multiplier, adder and register in FIR filters. Thus, the area overhead of an ASIC can be estimated according to the implementation structures of application algorithms. Based on the sampling rate of one algorithm, the minimum working frequency of one ASIC unit can then be obtained. Furthermore, the demand of an algorithm for millions instructions per second (MIPS) can be solved by the product of the sampling rate and the number of operation units. Due to the large area overhead for programming, control and communication, the area overhead of all EUs account for only a small part of the area overhead of whole reconfigurable architectures. We define the area overhead ratios of reconfigurable architectures to all EUs in the chips as *k*, if we use programmable architectures to realize the same algorithms. Thus, we can infer that for an application, programmable architectures will occupy *k* times area as that of ASIC to obtain the same MIPS, provided that the frequency is identical; similarly, programmable architectures will run at the frequency of *k* times as that of ASIC to keep the same area and performance.

The aforementioned relationship is easy to understand. Unfortunately, less attention is paid to the ratio value of *k*, which is often much larger than 10. It may be acceptable to use conventional reconfigurable architectures to implement the special applications such as prototyping development mentioned above. However, replacing ASICs in high-performance customer devices by current reconfigurable architectures on a large scale is still impossible. Therefore, it becomes the key to extend the successes of reconfigurable architectures how to reduce the ratio value of *k*. Thus, we attempt to adopt larger EUs in the PEs of a reconfigurable architecture.

3 Architecture

3.1 Array Structure

Most of real-life multimedia and baseband computations in custom devices belong to stream applications [4] and [5]. Besides computation intensity, they exhibit two other characteristics: concurrency and regularization. Fig. 2 shows the concurrency of stream applications. It includes two concepts:

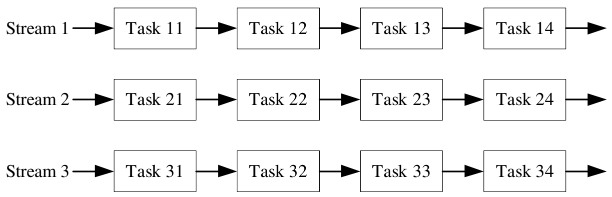

Fig. 2. The concurrency of a stream application

- Pipelined data path, in which every task can be processed simultaneously to form an efficient pipeline, similar to systolic structures [3];
- And data parallelism, where multiple data streams can be computed simultaneously.

Similarly, the regularization also has two concepts as follows:

- Producer-consumer locality, which embodies the relationship between task kernels. Each kernel receives the result data produced by its front kernel, consumes (processes) them, produces new data, and sends to its back kernels.
- Regular connection, which means there are no feedback and long-distance interconnection at unit or subsystem levels.

Based on the characteristics of current high-throughput applications, we propose an array structure shown in Fig. 3. It consists of a heterogeneous array of tens (or even hundreds) of PEs. The architecture is a template that can be instantiated for a given application. As a result, the function of every processing element depends on the mapping results of application algorithms at synthesis-time. Due to the locality of computation and communication, every PE just connect directly with its four nearest neighborhoods at the locations of the east, west, south and north. Besides direct interconnections, indirect connections can be formed by bypassing the ports of some PEs. Data-driven mechanism is employed in both interconnection network and PEs. One PE does not work until the needed data arrive. The networks of most reconfigurable architectures are synchronized by this way [8] and [9]. In our architecture, the EUs in PEs are also synchronized by data. Therefore, we call the architecture we propose as fully-data-driven reconfigurable architecture.

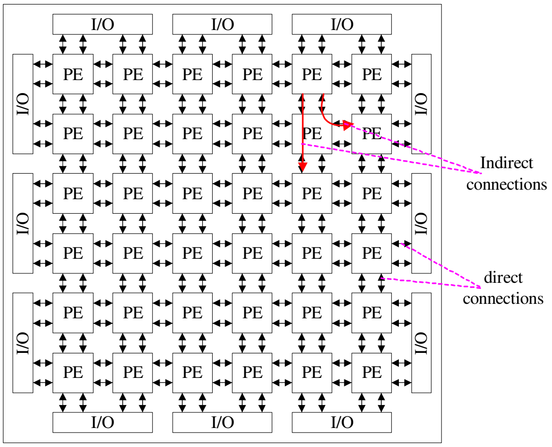

Fig. 3. The proposed heterogeneous array structure

3.2 Processing Elements

The functions of each PE are to configure a special interconnect network on the basis of a certain application, and to execute very-coarse-grained computations like multi-tap

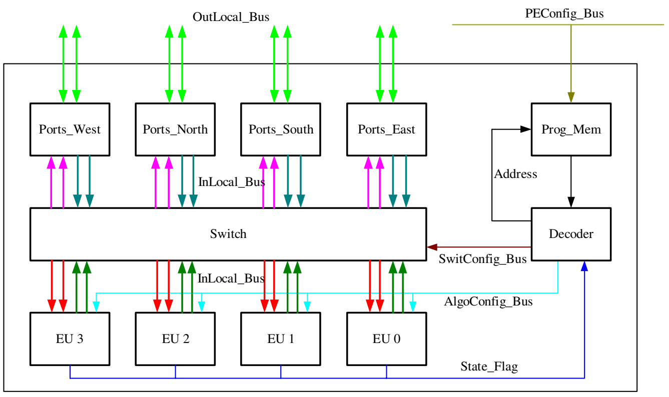

Fig. 4. The structure of the processing elements

FIR segments and FFT butterfly units. Fig. 4 presents the general structure of the PEs in proposed reconfigurable architecture, mainly including program memory (Prog_Mem), instruction decoder (Decoder), multiple ports (Ports_West, Ports_North, Ports_South and Ports_East), EU, and switch (Switch).

Instructions can be written through PE configuring bus (PEConfig_Bus) from embedded microprocessor or outside. Decoder is made use of to decode instructions, then to get the configuration information. Decoder does not work cycle by cycle, but actively fetch next instruction in term of the last instruction and the EU state flags (State_Flag).

There are four kinds of directional ports, namely Ports_West, Ports_North, Ports_South and Ports_East. Each cluster of directional ports includes two dual-direction ports connecting with other PE. The directions of dual-direction ports and the connections between the two ports and two internal single-direction input buses and two internal single-direction output buses are based on the configuration instructions. The input ports of one PE are directly connected with the output ports of its neighborhoods. The communications between these ports depend on their FIFOs states.

The configurations of EUs contain functional selection, constant coefficients, operation times, and etc. Fig. 5 shows the structure of EUs with two functions. Function select register is in charge of selecting function unit. Timer is responsible for setting the operation times of the selected function unit. Coefficient registers provide some constant coefficients to the function unit. In our architecture, function units include large-scale circuits like multi-tap FIR segments and FFT butterfly units. Small function units like adder and multiplier are also supported. The scale of function unit is based on actual applications. There is at least one FIFO in every EU. These FIFOs are used to save the input data from other PEs or input ports. The FIFOs in EUs as well as ports are very small. They can be composed of register files. The EU number of each PE may be different, which is relative to the input and output signals and the area overhead of EU.

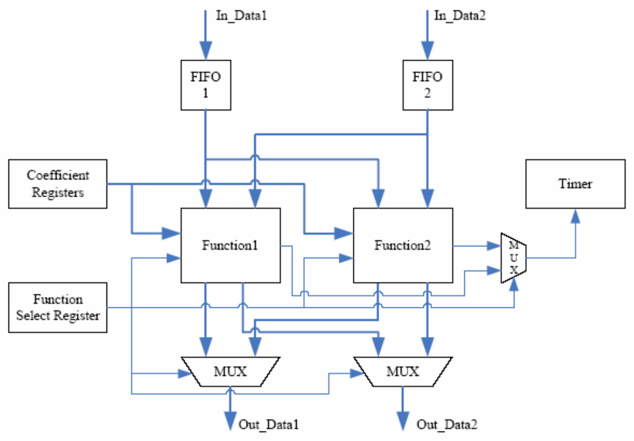

Fig. 5. The structure of execution units

Switch is the most important part that specifies the connections among all EUs and Ports, which will be introduced in the following section.

All PEs own the same Decoders, program memories, Ports, and Switches, though their roles are very different in the whole reconfigurable architecture. The only different is the execution units.

After finishing all configurations of Ports, EUs and Switch, Decoder will stay in a waiting state and continuously detect State_Flag. At this time, all the interconnection relations between Ports, EUs and Switch are static similar to an ASIC circuit driven by data. When the given number of operations has been done, EUs will give some certain signals to inform the decoder to reconfigure.

3.3 Switch

Three kinds of signals need to be routed. They are Data, Data-valid, and FIFO Near-full respectively. Data and Data-valid are always produced together. PEs or output ports receive data from other PEs or input ports on the basis of the indication of Data-valid. They can be treated with the same circuit structures. Fig. 6 shows the routing scheme of port output data. The scheme of EU input data is similar to this. The difference lies in the positions of No Connections.

When the FIFOs in EUs and output ports are close to be full, the input data signal's producers must stop to wait until the corresponding near-full signal is invalid. Otherwise, useful data will be lost. The routing of the near-full input signals is somewhat different from Fig. 6, mainly because the needed configuration signals are not the same. A translator is needed to translate the configuration information of data signals and data-valid signals to those of near-full input signals.

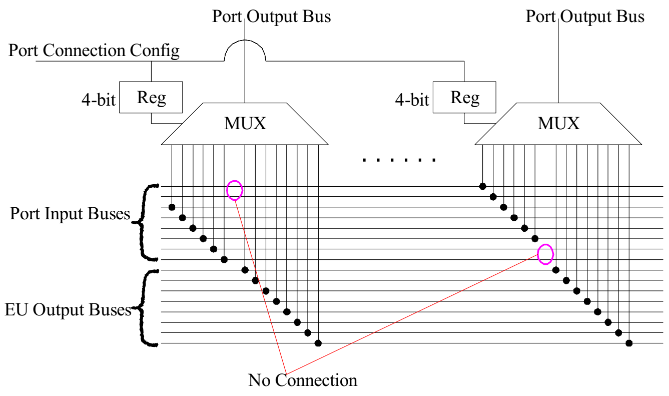

Fig. 6. The part of the Switch structure

4 Reconfiguration

Four types of instructions are distinguished as follows:

- Switch configuration instructions: Configure the connections between different EUs, between different ports, and between EUs and ports.
- Ports configuration instructions: Define the connections between ports and local interconnection buses, and the directions of the local dual-direction interconnection buses.
- EU configuration instructions: Specify the functions of EUs and the number of processing samples, and send the needed coefficients.
- Register clear instructions: Clear the data in ports FIFOs, EU registers, and EU FIFOs.
- Branch instructions: Jump to the initial address of a new application on the basis of the state flags of EUs.

Because of the simplification of PEs, the instruction operations are actually a series of sequential configurations. After configurations, the whole circuits of PE will be static. For next task, new configurations begin. According to the orderliness, we use the format of instruction packet to simplify the instruction decoders and minimize the instruction memories. The format is shown in Fig. 7. Due to the invariable sequence and format, the opcodes become implicit. Decoders can understand the meanings of codes in the packets, and send the corresponding configurations to the different parts of PEs. When a new state flag comes, Decoder begins to read next instruction packet.

By configuring the ports and switches, almost all PEs can communicate with each other. In fact, this may not be necessary due to the features of current high-throughput applications. As to the connections in each PE, the communications are multiform. For instance Ports_West can communicate with Ports_North, Ports_South or Ports_East in the one-to-one form. On the other hand, broadcasting communications can be provided. For instance, Ports_West can send data to Ports_North, Ports_South and Ports_East at the same time.

Commonly, there are multiple EUs in each PE. By different configurations, these EUs can work in parallel like Fig. 8 (a), and in pipeline like Fig. 8 (b). This multi-formity benefits from the reconfigurable switches which are able to form various connections among PEs and ports. When a task is over, PE can be reconfigured to implement a new task.

Task 1: Code Length
Ports Configuration
Switch Configuration
EU Function Selection
EU Operation Times
EU 1 Coefficient
EU 2 Coefficient
... ...
Task 2 ... ...

Fig. 7. The format of instruction packet

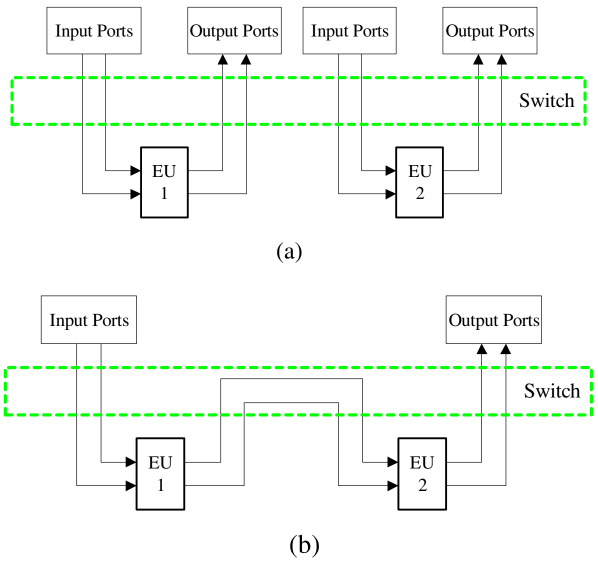

Fig. 8. The connections between EUs, (a) in parallel, and (b) in pipeline

5 Design Flow

The aim of the proposed reconfigurable architecture is not to provide a general processor, but to simply the design flow of ASICs and SOCs, for a narrow special application domain, and to lower the TTM and NRE cost. The design flow of the proposed architecture is shown in Fig. 9. The application codes in the form of MATLAB or C language are translated into the data flow graph with basic operations like multiplication and addition. Based on the area overhead demands, performance/cost ratio and programmability, some basic operations are combined together to form a function unit like FFT butterfly unit and Viterbi add-compare-select unit. Then, larger function units will be mapped as EUs onto the array architecture we proposed. The function units in different data paths can also be combined to form an execution unit with multiple functions if the application model is time-multiplexing. An optimizing process is needed in order to obtain the minimum processing elements. The results of optimization are a certain architecture and configuration instructions.

Because the interconnection network and PE configuration structure are fixed, and a set of execution units have been placed in a library, the design time of a chip will be shortened by reusing these predefined resources.

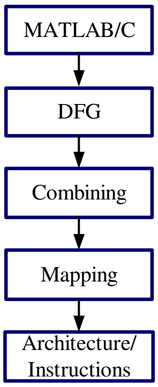

Fig. 9. The design flow of the proposed architecture

6 Performance Analysis

For illustrative purposes, let us consider the implementation of FIR digital filter on the proposed architecture. Fig. 10 gives the FIR filter structure and the possible execution units used to process the corresponding convolution tasks. We name the computation of one delay, multiplication and addition in Block A as one-tap operation, and similarly name the computation in Block B as two-tap operation. Clearly, the execution unit with the function of two-tap operation has the advantages over the execution unit with the function of one-tap operation in processing capability.

As is stated above that the reconfigurable architectures with less k can achieve higher performance/cost ratio. In order to estimate k, we design a configurable architecture with few PEs in the FPGA chip of Altera Cyclone II. Fig. 11 shows the k values under the condition of different grained EUs used in the proposed architecture. When the EU has the capability of one-tap operation, *k* is close to 10, while the value is about 2 when the EU has eight-tap operation function. With the increment of the granularity of EUs, the *k* value decrease gradually.

Several EUs can be cascaded to run a more-tap operation by configurating the Switch. Furthermore, several PEs can be cascaded to realize an ASIC-like digital FIR filter. Based on actual applications, the filter with different tap numbers and tap coefficients can be reconfigurated easily and quickly.

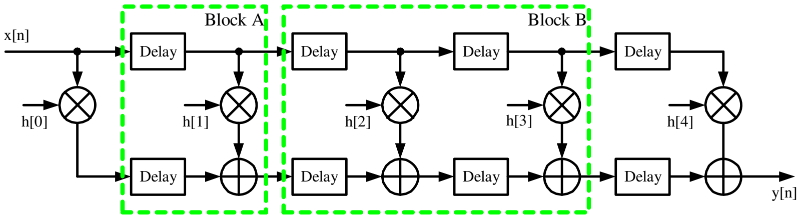

Fig. 10. A five-tap FIR filter

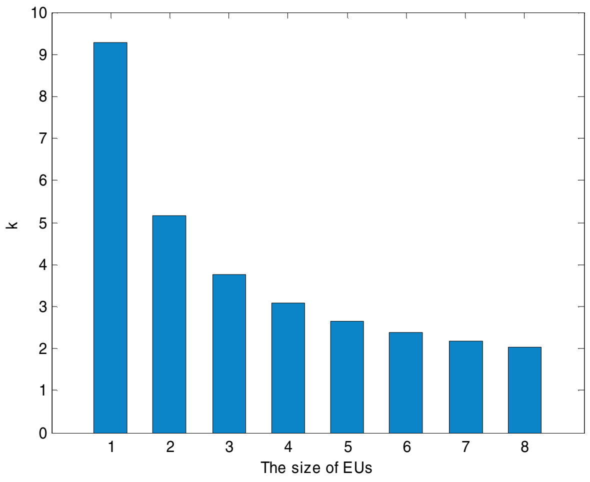

Fig. 11. The relation between *k* and the size of EUs

7 Conclusion

We have presented a reconfigurable architecture of high-throughput digital signal processing, with the characteristics of fully-data-driven mechanism and very-coarse-grained EUs. In the light of the trend of current application algorithms, a data-driven array structure or interconnection network is employed, which permits the direct connections between one PE and its four neighbors and also support the indirection connections between any two neighboring PEs by configurations. Unlike most coarse-grained reconfigurable architectures, each PE may contain very-coarse-grained EUs. By the configurations of PEs, the ASIC-like connections can be formed among EUs and ports with the data-driven mechanism to realize the mapping of special domain applications. The corresponding design flow of the reconfigurable architecture has also presented. In order to illustrate the role of very-coarse- grained EUs, we analyzed the relationship between the area overhead ratio of the whole architecture to all EUs and the granularity of the EUs. The analysis results showed that the area overhead ratio can be raised with the enhancement of EU granularity.

Future work is focused on further evaluation of the high-throughput processing capabilities of real-life applications such as WiMAX baseband processing and H.264 video system. In addition, a suitable methodology for mapping special domain applications onto the reconfigurable architecture should be explored.

References

1. Abnous, A., Rabaey, J.: Ultra-low-power domain-specific multimedia processors. In: 1996 Workshop on VLSI Signal Processing, pp. 461–470. IEEE Press, New York (1996)
2. Dennis, J.B.: Data flow supercomputers. Computer 13(11), 48–56 (1980)
3. Kung, H.T.: Why systolic architectures? Computer 15(1), 37–46 (1982)
4. Kapasi, U.J., Rixner, S., Dally, W.J., Abn, J.H., Mattson, P., Owens, J.D.: Programmable stream processors. Computer 36(8), 54–62 (2003)
5. Srikanteswara, S., Palat, R.C., Reed, J.H., Athanas, P.: An overview of configurable computing machines for software radio handsets. IEEE Communications Magazine 41(7), 134–141 (2003)
6. Hartenstein, R.: A decade of reconfigurable computing: a visionary retrospective. In: 2001 Conference on Design, automation and test in Europe, pp. 642–649. ACM Press, New York (2001)
7. Compton, K., Hauck, S.: Reconfigurable computing: a survey of systems and software. ACM Computing Surveys 34(2), 171–210 (2002)
8. Todman, T.J., Constantinides, G.A., Wilton, S.J.E., Mencer, O., Luk, W., Cheung, P.Y.K.: Reconfigurable computing: architectures and design methods. In: 2005 IEE proceedings of Computers and Digital Techniques, pp. 193–207. IEEE Press, New York (2005)
9. Abdin, Z., Svensson, B.: Evolution in architectures and programming methodologies of coarse-grained reconfigurable computing. Microprocessors and Microsystems (2008), doi:10.1016/j.micro.2008. 10.003
10. Koren, I., Mendelson, B., Peled, I., Silberman, G.M.: A data-driven VLSI Array for Arbitrary Algorithms. Computer 21(10), 30–43 (1988)
11. Chen, D.C., Rabaey, J.M.: A reconfigurable multiprocessor IC for rapid prototyping of algorithmic-specific high-speed DSP data paths. IEEE Journal of Solid-State Circuits 27(12), 1895–1904 (1992)
12. Taylor, M.B., Lee, W., Miller, J., et al.: Evaluation of the RAW microprocessor: an exposed-wire-delay architecture for ILP and streams. In: 31st Annual International Symposium on Computer Architecture, pp. 2–13. IEEE Press, New York (2004)
13. Duller, A., Towner, D., Panesar, G., Gray, A., Robbins, W.: PicoArray technology: the tool's story. In: 2005 Conference on Design, Automation and Test in Europe (DATE 2005), pp. 106–111. IEEE Press, New York (2005)
14. Yu, Z.: High performance and energy efficient multi-core systems for DSP applications. Ph.D. Thesis, University of California, Davis (2007)

15. Vejanovski, R., Stojcevski, A., Singh, J., Faulkner, M., Zayegh, A.: A highly efficient reconfigurable architecture for an UTRA-TDD mobile station receiver. In: 2003 International Symposium on Circuits and Systems (ISCAS 2003), pp. II45–II48. IEEE Press, New York (2003)
16. El-Rayis, A.O., Arslan, T., Erdogan, A.T.: Addressing future space challenges using reconfigurable instruction cell based architectures. In: 2008 NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2008), pp. 22–25. IEEE Press, New York (2008)
17. Cherepacha, D., Lewis, D.: DP-FPGA: an FPGA architecture optimized for datapaths. VLSI Design 4(4), 329–343 (1996)
18. Yeung, A.K.W., Rabaey, J.M.: A reconfigurable data-driven multiprocessor architecture for rapid prototyping of high throughput DSP algorithms. In: The Twenty-sixth Hawaii International Conference on System Sciences, pp. 169–178. IEEE Press, New York (1993)
19. Marshall, A., Stansfield, T., Kostarnov, I., Vuillemin, J., Hutchings, B.: A reconfigurable arithmetic array for multimedia applications. In: 1999 ACM/SIGDA Seventh International Symposium on Field Programmable Gate Arrays, pp. 135–143. ACM Press, New York (1999)
20. Mirsky, E., Dehon, A.: MATRIX: a reconfigurable computing architecture with configurable instruction distribution and deployable resources. In: 1996 IEEE symposium on FPGAs for custom computing machines, pp. 157–166. IEEE Press, New York (1996)
21. Singh, H., Lee, M., Lu, G., Kurdahi, F.J., Bagherzadeh, N., Chaves, F.E.M.: MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications. IEEE Transactions on Computers 49(5), 465–481 (2000)