International Journal of Engineering Education Vol. 33, No. 2(A), pp. 622–628, 2017
Printed in Great Britain

0949-149X/91 $3.00+0.00

Reconfigurable Computing System: A Project-Based Course for Graduate Students*

LEIBO LIU, CHENCHEN DENG, ZHAOSHI LI, SHOUYI YIN and SHAOJUN WEI
Institute of Microelectronics, Tsinghua University, Beijing, 100084, China. E-mail: chenchendeng@tsinghua.edu.cn

With a reduction of energy budgets and increasing aggregation of different applications, reconfigurable computing system is becoming a promising platform for future system-on–a-chip (SoC) designs. This paper presents a project-based course on reconfigurable computing systems. With the aid of an education platform, students learn to design their own mixed-grained reconfigurable architectures and implement a cryptographic algorithm. Combining both fine-grained Field-Programmable Gate Array (FPGA) and coarse-grained reconfigurable architectures (CGRAs), the course provides students an integrated view of reconfigurable computing system following the up-to-date scientific development in this area. Also, students gain practical hands-on experience of designing reconfigurable computing system using hardware-software co-design methodology. More than fifty students participate a comprehensive survey after the course and twelve faculty members of the similar research interests also appraise the course. Both the survey feedback and the course assessment confirms that the objectives of the course are well achieved.

Keywords: reconfigurable computing; engineering teaching; hardware-software co-design; project-based learning

1. Introduction

Combining the performance of application specific integrated circuits (ASICs) and the flexibility of general purpose processors (GPPs), reconfigurable computing fabrics are now of broad interests [1–3]. Courses on reconfigurable architectures are taught widely in academia [4–8]. Traditionally, students learn reconfigurable computing system through programming the field programmable gate array (FPGA) with the hardware description language (HDL) based design flow [4–7]. Advanced courses for graduate students may also cover topics such as the cooperation of programmable logics and soft cores on FPGAs, the utilization of embedded modules on FPGAs [8].

In addition to traditional fine-grained FPGAs, recent development shows that academia and industry are both interested in coarse-grained reconfigurable architectures (CGRAs) [9–11]. Sacrificing the bit-level reconfigurability of FPGA in trade of lower power consumption and shorter reconfiguration time, CGRAs are viewed as a promising trend for computing. Mixed-grained architectures [12, 13] are also proposed to combine the advantage of fine-grained and coarse-grained architectures. To equip graduate students with the up-to-date development in reconfigurable computing, this course is designed with both fine- and coarse-grained reconfigurable computing fabrics. Through a series of lab projects, they can apply theoretical concepts into practice and reinforce the understanding. Project-based methodology has been widely utilized in the FPGA-related engineering courses but CGRAs have been seldom involved [14–16]. For this course, both FPGA and CGRA are both included in the projects. There are four primary objectives of this course and by completing this course students are supposed to be able to:

(1) design a reconfigurable computing system with verified performance and gain practical hands-on experience of the hardware /software (HW/SW) co-design methodology during the design process;
(2) understand the key criteria describing reconfigurable computing systems including granularity, reconfigurability, interface, depth of programmability, computation model, etc.;
(3) learn the trade-off in the reconfigurable computing system design and the impact of design decision on different metrics (performance, area, power consumption, flexibility, logical complexity);
(4) comprehend the essence of collaboration between fine-grained and coarse-grained reconfiguration fabrics.

2. Background

2.1 Context

The target audience of this course is first/second year graduate students who are planning to conduct research in reconfigurable computing or related areas. The prerequisite of students' abilities is basic understanding of computer architecture. The knowledge of processor architecture and the programming language SystemC is optional. The par-

* Accepted 20 November 2016.

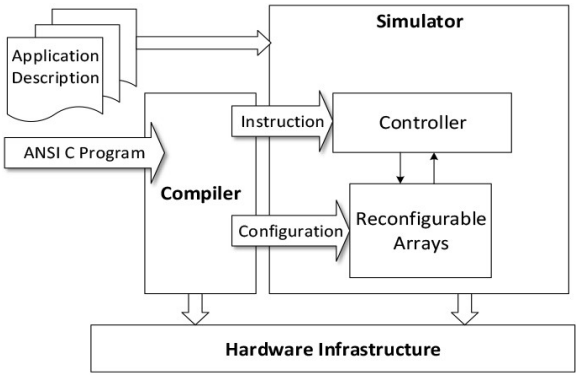

Fig.1 Block diagram of the platform for the course of reconfigurable computing systems.

ticipants of the course have various backgrounds. Some of them, majored in microelectronics, are familiar with digital circuit design. While some of them have the background of computer science but are also interested in circuit design.

2.2 Educational platform

For the lab modules, a dedicated educational platform is designed to help students learn about the up-to-date characteristics of reconfigurable computing system. As shown in Fig. 1, the platform consists of three parts: the simulator, the hardware infrastructure and the compiler.

The simulator is coded in SystemC [17] using SoC Designer [18]. With the help of the simulator, students can easily model reconfigurable architecture and quickly estimate the performance of various designs. The hierarchical architecture has three abstraction levels: Processing Element (PE), Processing Element Array (PEA) and Reconfigurable Processing Unit (RPU). From different levels, a set of architectural parameters reflecting important design decisions are extracted, and the parameter definitions are provided with the classes of each module. Students can generate an RPU simply by instantiating provided classes and defining parameters. The cycle accurate simulation mechanism and dedicated modeling techniques can give accurate estimation in performance, power and area for each design.

To fill the gap between simulation and real world implementation, a hardware platform is provided to verify students' designs. Multiple granularities are achieved by including two large-scale FPGAs on motherboard and eight coarse-grained reconfigurable arrays on daughterboard as shown in Fig. 2. With these RPUs, experiments of parallel and collaborative computing among the coarse-grained and fine-grained fabrics could be performed.

As an auxiliary tool, the C-based compiler translates ANSI C description of applications to the

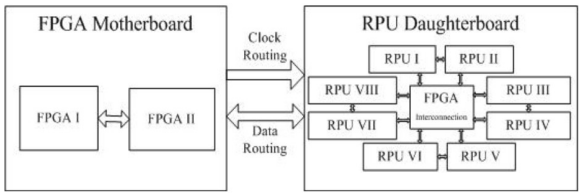

Fig. 2. Overview of the hardware infrastructure.

configuration contexts of RPU. The process of compilation is to "type and enter" in command line for simplification. By using the proposed compiler, hot codes are extracted from the C code, and then are transformed to data flow graphs (DFGs). Then the DFGs are passed to the back-end processing to generate configuration contexts for reconfigurable fabrics.

3. Project-based course

3.1 Course overview

Before the hands-on projects, four sessions of theoretical lectures are given to the students. The content of the lectures is summarized as follows:

1. Introduction of reconfigurable computing. It includes fundamental concepts and the development of reconfigurable computing.
2. High-level architecture of the reconfigurable computing system. This session involves both the hardware architecture and the compiler of reconfigurable computing systems.
3. Key techniques implemented in reconfigurable computing systems. These techniques include dynamic and partial reconfiguration, reconfigurable computing for computation-intensive and control-intensive applications, compilation techniques for reconfigurable computing systems, etc.
4. Preparation for the lab modules. The overview of the lab modules is briefly introduced and the background information of the target application is also given to students.

Since the reconfigurable computing is a new concept for most students, theoretical lectures are far from satisfying for students to fully understand. Hands-on projects are essential to help students comprehend the key features of reconfigurable computing as well as the process of designing a reconfigurable computing system. Students are divided into several groups with three or four students for each group. Students design their systems in the simulator based on the characteristics of the target application, after which the template-based compiler is used to generate the contexts for the application. Finally, the hardware platform is utilized to verify their design.

Table 1 The content of the lab modules

Modules	Topics	Design Flow
I	Overview	1. Analyze applications and decide which part of the application should be offloaded to reconfigurable fabrics. 2. Define the architectural parameters.
II	Compiler	3. Export these parameters to the compiler, write the annotated codes, and define the user-guided templates if necessary. 4. Compile the annotated code to an executable file.
III	Simulator	5. Execute the file in the simulator, and profile the statistics in simulator. 6. Analyze the performance, power and area characters to check whether the results meet the expectation, otherwise go back to step 1.
IV	Hardware	7. Verify their design on hardware infrastructure.

3.2 Lab modules

The goal of the lab modules is to design a mix-grained reconfigurable architecture and implement advanced encryption standard (AES) on the system. AES is chosen as the target application because of two primary reasons. One is that reconfigurable architectures are emerging as a promising solution for security applications by maintaining both flexibility and high energy efficiency [19–21]. Choosing a cryptographic algorithm follows one of the latest trends for reconfigurable computing. The other reason is that AES contains not only computation-intensive but also control-intensive operations. It also has its own characteristics which need to be dealt with. Different types of sub-tasks give student various choices in the design space exploration.

The project consists of four lab modules covering seven steps of design flow as shown in Table 1. The overview and correlation between these steps involving the educational platform are shown in Fig. 3. Module I acquaint students with overview of four modules to be completed. Also, the demonstration of the educational platform is also given. Students then start with the application analysis to determine which part of AES should be offloaded to the reconfigurable fabrics. Some kernels are mapped onto RPUs and the rest is executed on the host processor implemented by FPGA. Students can choose to connect RPUs to the host processor via bus or coprocessor interface by configuring the FPGA. The resulting performance of these two types of interface can be compared and therefore students get a basic grasp of interface between heterogeneous modules. Then based on the scale of computation, a set of important architectural parameters are determined. These tasks are assigned in the fourth lecture, and students are required to prepare for the lab module beforehand. By the end of this session, the lab organizer and demonstrators check the design of each student before they proceed to the next module.

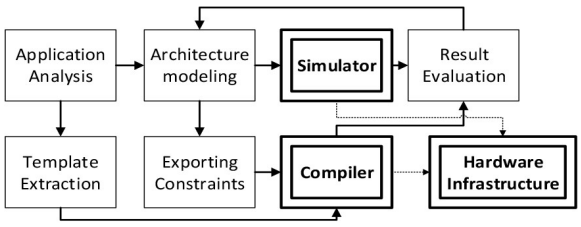

Fig. 3. Overview of the workflow of the educational platform.

Module II focus on the usage of the compiler. With the help of the compiler, convenient and accurate iteration of different designs can be implemented since more choices of architecture parameters and organizations are provided. Students start with define the parameters of the architecture in the compiler. The hot code, to be offloaded on to RPU for full exploitation of the parallelism and achieve the best performance, needs to be annotated manually. As an optional challenge, student can choose to further optimize the compiler by setting up some pre-defined template. Then the user-guided results will be compared with the automated complied results. By the end of the module, an executable file is expected to be generated by the compiler successfully. During this process, students are required to program and debug C language utilizing DFG.

Module III involves evaluating the design using the simulator. With a few instantiations and modifications on modules in SoC Designer, a prototype is set and experimental configuration contexts for RPU can be run on this system. However, in most cases, initial results are not satisfying due to lack of optimization. Therefore, after the initial trial, students analyze the key parameters that affect the performance, power and area respectively and make improvement under the guidance of the lab organizer and demonstrators. Iterations will be terminated once the requirements are met.

In module IV, students verify their design on a hardware platform. Configuration contexts are generated for those kernels to be mapped onto RPUs,

while bit-stream is generated for those tasks to be implemented on FPGA. Through a series debugging, running AES on the hardware platform is the first step. The performance is also required to be compared with the simulated results. The real-world implementation gives students a sense of accomplishment.

3.3 Course observations

During the lab modules, the lab organizer and demonstrators are there to help students with problems and check their design when necessary. Their observations in terms of common problems students encountered during the course are summarized as follows.

When design the MixColumn, some students use polynomial multiplier as a straightforward way. However, the profiled results, area in particular, are far from satisfying. After analysis, the dedicated polynomial multiplier is identified as the main problem. Students then decide to go back utilizing the basic logic and arithmetic operators in Arithmetic Logic Units (ALUs) with a few steps to implement MixColumn. Although the number of cycles is increased, the performance can be improved through pipeline. After the modification, the area and power consumption are both reduced.

The S-Box is identified as the most computation-intensive sub-algorithm in AES. Students choose to implement S-box by the 4-bit Galois Field (GF) operation since the ALUs in the reconfigurable fabrics are not as flexible as the Look-Up Table in FPGAs. This results in a 16×16 PEA with 4-bit granularity for each PE. The area of each PEA is much larger than the previous one due to the routing overhead, which can be derived from the area estimation in the simulator. To set up the reconfigurable fabric in the simulator, students have to modify the RTL code for the ALU to add support for the operations in Galois Field. By modifying the function of ALUs, students can practice RTL coding in addition to the high-level programming.

When generating the configuration contexts, they encounter another problem. Operations in Galois Field are bit-wise operations that cannot be expressed effectively in C-language, and it is even more difficult to allocate the PE resources in the compiler. Students decide to generate the configuration contexts of S-Box all by hand. Generating part of configuration contexts by hand gives students a deeper understanding of coarse-grained reconfigurable architecture.

4. Evaluation

4.1 Student Assessment

The assessment of students' performance is based on three parts: lab modules (40%), final reports (30%) and individual interview (30%). At the end of each module, students are signed off by the lab organizer or demonstrators. The results and the notes on the log book will be marked and the score for each module takes 10% of the overall marks. The final report is submitted in group, and students should state their analysis of target application, describe important decisions in design process and present their final results. In addition, several mandatory questions are given to them to answer in the report. Detailed information of these questions will be given in Section 4.2. To make sure that each student fully understands their report, an individual interview is arranged.

4.2 Course assessment

The statistics of the marks obtained by student from lab modules, final reports and interviews are categorized by the key points of learning in each course objective (Section I) as shown in Table 2. The marks, calculated on the scale from 10 to 0, reflect how students have acquired the relevant knowledge of reconfigurable computing systems through the course in a straightforward way.

As shown in Table 2, the marks for the key criteria describing the characteristics of reconfigurable computing systems are quite high in general. It confirms that the abstract definitions become explicit with the examples in the lab projects. Similar conclusion can also be made for the design tradeoff and the collaborative computing. Students are proved to comprehend most of the key points very well except for the two points mentioned in the interviews. These two points of knowledge are extended from the course which have not been fully discussed in the course. This suggests that more efforts are needed to inspire students to think on their own initiative. For the hands-one experience, most students can accomplish the tasks in each step with the help of teachers. The speed and the independence are also considered during the marking process. User-guided templates are optional for students to optimize the design. A few students find it difficult while some students have done this job very well. In summary, Table 2 shows that the characteristics of reconfigurable processor and HW/SW co-design methodology are well received by students.

After the course, a satisfaction survey is also conducted anonymously with the help of educational administration system and more than fifty participants of the course have responded to the survey. Statistics results of the survey are summarized in Table 3 and marks are given between 10 and 0 representing strongly satisfied to not satisfied at all. Table 3 shows that students are quite satisfied the

Table 2 The average and standard deviation of the marks obtained by students for each point of learning categorized by the course objectives (the number of participants: 53)

Objectives	Course Arrangement	Detailed Information	Avg.	Dev.
(1) Hands-on experience	Module I	Analysis of the characteristics of the target application.	8.38	3.01
		The hierarchical modeling strategy for reconfigurable computing.	9.17	2.3
		Identification of the hot codes (computation-intensive part).	8.92	1.74
	Module II	The architectural parameters are correctly defined in compiler.	8.62	2.05
		The codes are annotated.	10	0
		User-guided templates are defined and the explanation.	7.33	3.54
		Successful creation of executable file.	10	0
	Module III	Demonstration of profiling performance, power and area.	9.02	0.98
	Module IV	Generate bit-stream for FPGA and configurable context for RPUs	9.66	1.14
		Demonstration of running AES on the hardware platform.	10	0
		Analysis of performance, power and area compared with the simulated results.	8.69	1.92
(2) Key Criteria	Module I	The value of architectural parameters (granularity, size of the PEA, and etc.).	9.81	0.40
	Final report	The relationship between the granularity of the computing fabrics and applicable functions.	9.78	0.62
		The methods to implement different interfaces on reconfigurable computing fabrics.	8.72	1.54
		The reason that different computation models could be implemented on the same hardware for reconfigurable computing system.	8.45	1.26
	Interview	The difference between fine- and coarse- grained reconfigurable computing fabrics.	10	0
		The difference between dynamic and static reconfiguration;	9.69	0.45
		The benefits of dynamic reconfiguration;	9.33	1.07
(3) Design Trade-off	Module III	Analysis of the simulated results compared to the expected resulted.	9.29	0.99
		Description of the solution to improve the design.	9.11	0.84
	Final report	The correlation between performance, power and area.	9.21	1.37
		The fundamental cause of interconnection among them.	8.78	1.82
	Interview	Methods to balance these factors based on different requirement.	8.29	2.60
(4) Collaborative Computation	Module I	Determine the modules to be offloaded onto the reconfigurable computing fabrics.	9.09	1.79
	Final report	Explanation of the partition decision.	9.65	3.2
		The concept and the benefit of the collaborative computing.	9.42	1.99
		The mechanism of collaborative computing between the fine-and coarse-grained computing fabrics.	8.77	2.10
	Interview	The key techniques to implement parallel computing among various computing fabrics.	8.50	1.61

Table 3 Results of student satisfaction questionnaire (the number of participants: 53)

Category	Factors for satisfaction	Avg.	Dev.
During the course	Course structure arrangement	8.63	1.61
	Theoretical lectures	8.24	1.28
	Demonstration through lab projects	9.21	0.66
	Teamwork for lab modules	9.46	1.39
	User-experience of the platform	8.87	1.44
	Workload of debugging	8.23	1.70
	Methods of assessment	8.87	1.14
After the course	Knowledge consolidation	9.12	0.79
	Hands-on practical experience	9.27	0.63
	Motivations and interests	9.18	1.17
	Support for future research	9.20	0.73

Table 4 Results of teacher satisfaction questionnaire (total number of participants: 12)

Satisfaction Questions	Avg.	Dev.
1. Is the design of the course well motivated?	9.35	0.53
2. Does the content of this course follow the state-of-the-art research in reconfigurable computing?	8.98	0.42
3. Does the platform serve well as an educational platform in general?	8.83	1.20
4. Is the course assessment fair and reasonable?	9.12	0.70
5. Can the course help prepare student with solid knowledge foundation, problem solving capability and critical thinking skills for future individual scientific research?	9.28	0.62
6. Is the course helpful for research activities in other areas?	9.22	0.83

project-based form of studying and designing reconfigurable systems from the perspective of both during and after the course. Some students find the workload of debugging through the course is a bit overwhelming. The increase of debugging capability comes with long time of coding experience. Therefore, in a short term of course, it might be difficult for some inexperienced students but the course helps gain practical experience in a long term.

In addition to the survey among the course participants, twelve teachers are also invited to participate in the survey as a part of evaluation of the course. Half of them are from Tsinghua University and they help with the interview session of the assessment. Teachers from Southeast University and ShangHai JiaoTong University are also invited to evaluate the course and they are all scholars of reconfigurable computing. Some of them are the supervisor of the students who take this course for future research. The process of evaluation includes the inspection of the course structure and the educational platform. In addition, teachers are also invited to drop in some lab sessions. Table 4 lists the evaluation results given by these teachers. From the point of view of the faculty members, the course helps student learn and design the reconfigurable computing system effectively. Feedback from teachers who are the supervisors of graduate students also shows that the students who have participated in the course are making faster progress in their own research project than those who have not. Therefore, these teachers claim that they would encourage their graduate students to take the course in their first-year study.

5. Conclusions and future work

The motivation of the course is to teach students the cutting-edge knowledge in reconfigurable computing through a series of lab projects. To facilitate the understanding of the theoretical concept, the course utilizes an educational platform including simulator, hardware infrastructure and compiler to acquaint students the design flow of a reconfigurable computing system. The four lab modules can be considered as a mini research project with the guidance of the lab organizer. Participating in the course grants students with a deeper understanding of reconfigurable computing and an integrated view of HW/SW co-design flow in furtherance of their own future research.

Feedback from students and teachers is quite encouraging, but also reveals some limitations of the work. For example, students find it quite difficult to the use of compiler in the educational platform and the workload of debugging is quite heavy. Also, the theoretical lectures are a bit demanding for students to learn a novel type of computing system in a relatively short time. To improve the course, further optimizations based on the feedback are now in progress. The automation of the compiler will be further improved and therefore the prior knowledge of the applications can be less demanding. In this way, students can explore the architecture design more freely and easily. For course structure, the theoretical lectures will be arranged in the intervals of the lab modules instead of before them. This might further enhance the comprehension of the theoretical concepts if a relevant lab project is followed.

References

1. O. Atak and A. Atalar, BilRC: An Execution Triggered Coarse Grained Reconfigurable Architecture, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, **21**(7), 2013, pp. 1285–1298.
2. Y. H. Park, J. Kim, M. Kim, W. Lee and S. Lee, Programmable multimedia platform based on reconfigurable processor for 8K UHD TV, *IEEE Transactions on Consumer Electronics*, **61**(4), 2015, pp. 516–523.
3. G. Brebner and W. Jiang, High-Speed Packet Processing using Reconfigurable Computing, *IEEE Micro*, **34**(1), 2014, pp. 8–18.
4. V. Sklyarov and I. Skliarova, Teaching reconfigurable systems: methods, tools, tutorials, and projects, *IEEE Transactions on Education*, **48**(2), 2005, pp. 290–300.
5. L. Sousa, S. Antao, and J. Germano, A Lab Project on the Design and Implementation of Programmable and Configurable Embedded Systems, *IEEE Transactions on Education*, **56**(3), 2013, pp. 322–328.
6. V. Sklyarov, I. Skliarova and A. Sudnitson, Methodology and international collaboration in teaching reconfigurable systems, *IEEE Global Engineering Education Conference (EDUCON)*, 2012, pp. 1–10.
7. A. Shoufan, and S. Huss, A course on reconfigurable computing, *ACM Transactions on Computing Education*, **10**(2), 2010, pp. 7.1–7.20.
8. X. Wang, Using FPGA-based configurable processors in teaching hardware/software co-design of embedded multiprocessor systems, *Microelectronic Systems Education*

(MSE), 2011 IEEE International Conference on, San Diego, 5–6 June 2011, pp. 114–117.
9. H. Sim, H. Lee, S. Seo and J. Lee, Mapping Imperfect Loops to Coarse-Grained Reconfigurable Architectures, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **35**(7), 2016, pp. 1092–1104.
10. B. C. Schafer, Tunable Multiprocess Mapping on Coarse-Grain Reconfigurable Architectures with Dynamic Frequency Control, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, **24**(1), 2016, pp. 324–328.
11. J. Yao, M. Saito, S. Okada, K. Kobayashi and Y. Nakashima, EReLA: A Low-Power Reliable Coarse-Grained Reconfigurable Architecture Processor and Its Irradiation Tests, *IEEE Transactions on Nuclear Science*, **61**(6), 2014, pp. 3250–3257.
12. D. Rossi, F. Campi, S. Spolzino, S. Pucillo and R. Guerrieri, A Heterogeneous Digital Signal Processor for Dynamically Reconfigurable Computing, *IEEE Journal of Solid-State Circuits*, **45**(8), 2010, pp. 1615–1626.
13. I. Sourdis, D. A. Khan, A. Malek, S. Tzilis, G. Smaragdos and C. Strydis, Resilient Chip Multiprocessors with Mixed-Grained Reconfigurability, *IEEE Micro*, **36**(1), 2016, pp. 35–45.
14. J. M. P. Cardoso, Teaching strategy for developing application specific architectures for FPGAs, *International Journal of Engineering Education*, **24**(4), 2008, pp. 833–842.
15. J. Olivares, J. M. Palomares, J. M. Soto, J. C. Gamez, I. Bravo and A. Gardel, Learning FPGA Design by a Methodology Based on Projects, *International Journal of Engineering Education*, **27**(3), 2011, pp. 509–517.
16. J. de Jesus Rangel-Magdaleno, J. Rooney Rivera-Guillen, R. de Jesus Romero-Troncoso, H. Peregrina-Barreto and J. Pedro Sanchez Santana, Open core hardware description practices for DSP masters degree course, *International Journal of Engineering Education*, **28**(3), 2012, pp. 651–662.
17. IEEE Standard for Standard SystemC Language Reference Manual, *IEEE Std 1666-2011* (Revision of IEEE Std 1666-2005), 2012, pp. 1–638
18. Carbon SoC Designer Plus, http://www. carbondesign systems.com/soc-designer-plus/, Accessed 18 May 2016.
19. L. Bossuet, M. Grand, L. Gaspar, V. Fischer and G. Gogniat, Architectures of flexible symmetric key crypto engines: a survey from hardware coprocessor to multi-crypto-processor system on chip, *ACM Computing Surveys*, **45**(4), 2013, pp. 41:1–41:32.
20. W. Shan, L. Shi, X, Fu, X. Zhang, C. Tian, Z. Xu, J. Yang and J. Li. A side-channel analysis resistant reconfigurable cryptographic coprocessor supporting multiple block cipher algorithms, *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*, San Francisco, 1–5 June 2014, pp. 1–6.
21. S. Majzoub and H. Diab, MorphoSys reconfigurable hardware for cryptography: the twofish case, *Journal of Supercomputing*, **59**(1), 2012, pp. 22–41.

Leibo Liu received the B.S. degree in electronics engineering from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from the Institute of Microelectronics, Tsinghua University, in 2004. He currently serves as an Associate Professor with the Institute of Microelectronics, Tsinghua University. His research interests include reconfigurable computing, mobile computing, and very large scale integration digital signal processors.

Chenchen Deng received the B.S. degree in electronic engineering from the Beijing University of Posts and Telecommunications, China, in 2007, and the D.Phil. degree in engineering science from the University of Oxford, U.K., in 2012. She is currently with the Institute of Microelectronics, Tsinghua University, China. Her research interests include reconfigurable computing, reconfigurable network-on-chip, and System-on-Chip design.

Zhaoshi Li received the B.S degree in microelectronics from Tsinghua University, Beijing, China in 2012. He is currently pursuing the Ph.D. degree in microelectronics engineering at Tsinghua University. His current research interests include architecture design for reconfigurable computing.

Shouyi Yin received the B.S., M.S. and Ph.D. degree in Electronic Engineering from Tsinghua University, China, in 2000, 2002 and 2005 respectively. He has been with Imperial College London, London, U.K., as a Research Associate. He is currently with the Institute of Microelectronics, Tsinghua University, as an Associate Professor. His current research interests include reconfigurable computing, mobile computing, and system-on-chip design.

Shaojun Wei received the Ph.D. degree from Faculte Polytechnique de Mons, Belgium, in 1991. He became a Professor with the Institute of Microelectronics, Tsinghua University, in 1995. He is a Senior Member of the Chinese Institute of Electronics. His main research interests include very large scale integration System-on-Chip design, EDA methodology, and communication application-specific integrated circuits design.