An Efficient Configuration Scheme of CGRA for Block Ciphers

Wei Jinghe¹, Wang Yufeng², Sun Jinjun³, Yang Jinjiang⁴,*

¹The 58th Research Institute of China Electronic Technology Corporation, Wuxi, China
²Wuxi Research Institute of Applied Technologies Tsinghua University, Wuxi, China
³Wuxi Research Institute of Applied Technologies Tsinghua University, Wuxi, China
⁴Wuxi Research Institute of Applied Technologies Tsinghua University, Wuxi, China
*yangjinjiang@tsinghua-wx.org

Abstract. An Efficient Configuration Scheme based on Configuration Context (ECSCC) is proposed to decrease the size of configuration memory and the time of configuration transmission. The ECSCC is applied to a coarse-grained reconfigurable architecture (CGRA) which is implemented under TSMC 40nm CMOS technology to compare with similar work. The results show that the proposed ECSCC improves the area efficiency by 21.56 X and the energy efficiency by 3.97X.

1. Introduction

The Coarse-Grained Reconfigurable Architecture adopts lots of computing resources and simplifies the interconnection among computing resources, thus CGRA has outstanding area efficiency and energy efficiency in the domain specific area [1][2][3]. However, with the increasing complexity of the application and the performance requirements of the algorithms, the capacity of the configuration becomes larger. The large capacity of configuration not only leads to the area overhead of configuration memory, but also affects the time overhead of reconfiguration transmission. Therefore, it is important to develop a feasible configuration scheduling scheme to improve the area and energy efficiency of CGRA.

The configuration which is used to configure the function of the computing resources in CGRA, is usually stored in the configuration memory. Reducing the area and time overhead used to implement the configuration needs to research the manner of configuration and the organization of configuration memory. The researchers put forward Hierarchical Configuration Context (HCC) [4], Hierarchical Context Organization (HCO) [5] and other schemes to reduce the capacity and the time of configuration. HCC scheme is based on the similarity of the kernels in the application, which reduces the capacity of the configuration by storing configuration hierarchically. HCO scheme switches the configuration row by row during the calculation, and the pipeline technology is used to reduce configuration time.

This paper proposed an Efficient Configuration Scheme based on Configuration Context. The configuration is scheduling according to the space and time characteristics of configuration context to decrease the ratio of the configuration time to calculation time. Then, the ECSCC are applied in a reconfigurable cryptographic processor, which supports several famous block ciphers. The processor is
implemented under TSMC 40nm CMOS technology, the working frequency is 350 MHz and the area is 5.12 mm$^2$. Compared with other similar work, the proposed ECSCC gains 3.32X to 21.56X incensement of area efficiency improvement and 1.19X to 3.97X advantage of energy efficiency.

2. Configuration Scheme of CGRA

2.1. Overview of reconfigurable architecture

As is shown in Fig.1, the coarse-grained reconfigurable architecture contains two parts which are configuration controller and computing array. The configuration controller, which consists of configuration memory and a scheduling module, is designed for storing and scheduling the configuration of the algorithm. The basic operations discussed above are implemented in the computing array as different Processing Elements (PEs). The PEs are organized by row and rows relate to cross-bar connection. Since the look-up table for S-box leads to much memory cost and it is not necessary to be implemented each row, every four rows share a look-up table for S-box operation. When we are executing the cryptographic algorithm, the configuration controller is used to send the configuration to the computing array step by step. Then the computing array configures the PEs to corresponding functions to perform the operations.

Configuration efficiency is the key factor of the performance of the whole reconfigurable architecture. So, this paper considers the design of the configuration controller. The purpose scheme is to reduce the configuration time and decrease the area cost of configuration memory.

Figure 1. The overview of reconfigurable architecture.

2.2. The Scheduling Scheme of Configuration

The size of the algorithm DFG is usually larger than that of reconfigurable computing array, so the dynamic reconfiguration technique is usually adopted to map the sub-DFGs to the Reconfigurable Computing Array (RCA) without stopping the calculation. Figure 2 shows the hardware structure of the configuration controller which uses the proposed organization of configuration discussed in section 2.2. As shown in the Fig.4, the top layer configuration is sent to configuration controller from outside, and it is parsed by top layer parser, T_parser. The external data transmission configuration is written to the External Data_Cfg Register in RCA directly. The index of middle layer configuration is sent to the middle layer parser, and it is parsed by M_Parser. M_parser write the internal data transmission configuration to Internal Data_Cfg Register in the RCA and send the index of the bottom configuration to the bottom layer parser, B_Parser. B_Parser reads the function configuration from bottom layer configuration memory and write it to Function Configuration Register in the RCA.
However, the configuration switching process causes a pause of RCA calculation, which will affect the computation efficiency of the RCA. The effective solution to alleviate the problem is hiding the configuration time into the calculation time as much as possible. To achieve this goal, multiple sets of configuration registers are used, such as Architecture for Dynamically Reconfigurable Embedded System (ADRES) [3]. While the current task is executing, the configuration of the next task can be written to another set of the configuration register in advance. When the current task is completed, we can switch to the next task immediately. But with the increase of RCA scale, the area cost will be extremely high. Another solution is to divide the computing resources of RCA into several blocks [5], then these blocks are configured partially.
To explain how the partial configuration scheme to hide the configuration time, we suggest that the RCA is divided into two parts, region A and region B. We can start to configure Region B/A when the Region A/B is doing the calculation task. But if Region A has more resources and Region B has less resources, the calculation time and configuration time of each region will be mismatch. In the actual case, the computing resources of the RCA are usually divided into more regions, so it is necessary to consider how to divide the entire RCA into appropriate parts carefully. An Efficient Configuration Scheme based on Configuration Context is proposed in this section. RCA is divided into appropriate parts dynamically according to the configuration context of the configuration process. The working flow of the ECSCC is shown in Fig.3. Two hardware units are implanted in the M_Parser and B_Parser respectively to monitor the context of the configuration process. As shown in Fig.3, the details of the working flow contain four possible scenarios, which are described as follow.

**Scenario 1:** If it is the first time to configure the RCA, basic configuration scheme is performed. The basic configuration scheme means it is not necessary to configure the RCA partially, because the configuration registers of RCA are empty at the first time. The RCA is configured as an entirety, and all configuration registers are updated when the basic configuration scheme is used.

**Scenario 2:** If it is not the first time to configure the RCA and the sub-DFG is the same as the last one, function registers of the RCA are not necessary to update. In order to determine the sub-DFGs consistency, the index of middle layer configuration of the last sub-DFG is stored in the M_Parser to compare with current one.

**Scenario 3:** If the sub-DFGs are not same as the last one, the RCA is dynamically divided into appropriate parts by B_Parser according to the configuration context of bottom layer configuration. If the RCA could not be mathe the mode selection, basic configuration scheme will be performed.

**Scenario 4:** If the RCA could be divided into the partition configure modes. The B_Parser should pick the best suitable mode from the four previous modes for the RCA. The four modes are described as following. 4 rows, 2 rows, 1 row and PEs of the RCA is considered as the minimum partition in mode 1, mode 2, mode 3 and mode 4 respectively. The RCA is configured partially using the mode determined by scenario 3. It should be noticed that if the configuration of the partition is the same as the last time, the registers of this part of RCA are not necessary to update.

The configuration switches row by row during the calculation, according to space characteristic of configuration context. However, the proposed ECSCC take both space and time characteristics into consideration to decrease the ratio of the configuration to calculation time, which is describe in scenario 4.

### 3. Experiment Results and Comparison

To compare the area and energy efficiency to other similar works, the cryptographic processor is implemented under TSMC 40nm CMOS technology. The RCA contains 32 rows, in which 3 PEs (arithmetic unit, logic/shift unit and permutation unit) are designed. Every four rows share a Lookup Table Unit. Results by Synopsys IC compiler are shown in Table I, and the traditional block cipher, Advanced Encryption Standard (AES), is implemented on different architectures. High area efficiency and energy efficiency are our key targets, since we should consider the performance, power and area of the cryptographic processor comprehensively.

|                        | REPROC [5] | ManyCore [6] | Cryptoraptor [7] | proposed |
|------------------------|------------|--------------|------------------|----------|
| Frequency (MHz)        | 400        | 1210         | 1000             | 350      |
| Technology (nm)        | 65         | 65           | 45               | 40       |
| Area (mm²)             | 51.36(19.44*) | 6.63(2.51*) | 6.32(4.99*)      | 5.12     |
| Throughput of AES (Gbps)| 51.2      | 1.019        | 128              | 44.8     |
| Power (W)              | 0.58       | 1.58         | 6.18             | 0.43     |
| Area Efficiency (Gbps/mm²)| 1.00 (2.63*) | 0.15 (0.41*) | 20.25 (25.63*)   | 8.75     |
| Energy Efficiency (Gbps/W) | 87.67   | 0.64         | 20.72            | 104.19   |

* Normalized under TSMC 40nm technology
As is shown in Table I, compared to a classical many-core processor [6], our proposed implementation achieves 21.56X increment of area efficiency and two orders of magnitude increment of energy efficiency. To compare with CGRAs, the proposed architecture achieves 3.32X increment of area efficiency and 1.19X increment of energy efficiency over REPROC [5]. The Cryptoraptor [7] achieves a significantly high work frequency up to 1 GHz by optimizing the timing of the memories, which makes its throughput extremely higher than other CGRAs. However, the proposed one still has 3.97X advantage of energy efficiency. It should be noticed that the area efficiency of other architectures is normalized under TSMC 40nm technology for fair comparisons.

4. Conclusion
An Efficient Configuration Scheme based on Configuration Context (ECSCC) is proposed in this paper. The time of configuration transmission are reduced a lot. It is applied to a reconfigurable processor, on which several block ciphers are implemented. Compared with other similar works, the proposed ECSCC gains 3.32X to 21.56X increasement of area efficiency improvement and 1.19X to 3.97X advantage of energy efficiency. Thus, the ECSCC is very suitable for the reconfigurable architecture, which is applied to the implementation of ciphers.

References
[1] R. Hartenstein: “A Decade of Reconfigurable Computing: A Visionary Retrospective,”, Design, Autom. Test Eur. Conf. Exhibit (2001) 642.
[2] F. Campi, A, et al: “A dynamically adaptive DSP for heterogeneous reconfigurable platforms,”, Design, Autom. Test Eur. Conf. Exhibit (2007) 1.
[3] B. Mei, B. Sutter, T. Aa, M. Wouters, A. Kanstein, S. Dupont: “Implementation of a Coarse-Grained Reconfigurable Media Processor for AVC Decoder,”, Journal of Signal Processing Systems 51 (2008) 225.
[4] W. Yansheng, L. Leibo, Y. Shouyi, Z. Min, C. Peng, Y. Jun, W. Shaojun: “ION-Chip Memory Hierarchy in One Coarse-Grained Reconfigurable Architecture to Compress Memory Space and to Reduce Reconfiguration Time and Data-Reference Time,”, EEE Trans. Very Large Scale Integr. (VLSI) Syst. 22 (2014) 983.
[5] W. Bo, L. Leibo: “Dynamically reconfigurable architecture for symmetric ciphers,”, Science China Information Sciences. 59 (2016) 42403.
[6] B. Liu, B. M. Baas: “Parallel AES Encryption Engines for Many-Core Processor Arrays,”, IEEE Trans. Computers. 62 (2013) 536
[7] G. Sayilar, et al: “Cryptoraptor: high throughput reconfigurable cryptographic processor,”, IEEE/ACM International Conference on Computer-Aided Design (2014) 154.