A Modeling method for Reconfigurable Processor Performance Analysis

Minhui Hu[1,a], Guanwu Wang[1], Kongfei Du[1], Sikun Li[1]

[1]College of Computer, National University of Defense Technology, Changsha, 410073, China

[a]email: huminhui94@163.com

Keywords: Coarse Grained Reconfigurable Architecture; Transaction-level Dataflow Graph; Reconfigurable Processor; Performance Analysis Model

Abstract. Coarse grained reconfigurable architecture (CGRA) has become an important solution for high performance computing because of its high speed up ratio for computation intensive applications, fast configuration, good adaptability and low power consumption. However, the traditional performance analysis method of register transfer level modeling is simulating, which is still widely used. To overcome the shortage of traditional performance analysis method in flexibility and efficiency, we propose a novel method for reconfigurable processor performance analysis based on transaction-level dataflow graph (DFG). This method combined application data flow graph with architectural characteristics through constructing the transaction-level dataflow graph of reconfigurable processor and labeling hardware performance parameters corresponding to functional properties. Using the breadth first search algorithm based on hierarchical search to analysis the tree-like transaction-level dataflow graph, we can get the performance of reconfigurable processor running applications. Experimental results show that the proposed method calculates the performance of reconfigurable processor quite accurately, and benefits the optimization design of architecture and design space exploration.

Introduction

CGRA has become a hot research topic for accelerating the computation intensive applications in some specific fields, since its high speed up ratio, fast configuration, good adaptability and low power consumption. The architecture of CGRA is customized for specific domain, and it is difficult to discriminate between good and bad architecture based on experience of domains. We need to use quantitative evaluation metrics to measure the architecture. The quantitative metrics include performance (such as throughput, response time and so on) and cost (such as area, consumption and so on). And the performance index is particularly important. Usually the time performance for reconfigurable processor is indicated by its running cycles for executing the computation intensive application. In this paper, we mainly investigate the time performance.

The performance evaluation method of electronic system is divided into two categories, the simulating approach and the analysis approach. The traditional performance evaluation method of reconfigurable processor is simulating, which is still widely used. In paper [1], a co-simulator is developed to interactively simulate the system running and application mapping. The co-simulator can simulate and obtain the performance metric as well. Some commercialization simulating platforms [2][3][4] have ability of functional simulation verification and performance evaluation for CGRA, which is designed on these platforms. The simulating method is evaluating the performance during simulation, and this method has high degree of precision. However, building the simulation model and generating simulation vector is a hard work. Specially, CGRA running is controlled by the configuration dynamically. In addition, there is not a compiler tool for a new CGRA based on simulating approach. The configuration is generated manually, which is both tedious and error prone.

The analysis approach for performance evaluation of CGRA gets the performance metric by building a performance analysis model. This method has higher efficiency with lower precision than the simulating approach. In the field of performance evaluation for general purpose microprocessor, there are many research based the analysis approach [5][6] as a supplement to the simulating

method generally. Though, CGRA has dynamically reconfigurable feature, the analysis approach of general purpose microprocessor is difficult to use on CGRA directly.

In this paper, we research on the performance evaluation problem for CGRA in transaction-level. A novel performance analysis method based on transaction-level dataflow is proposed. This method combined application dataflow graph with architectural characteristics through constructing the transaction-level dataflow graph of reconfigurable processor and labeling hardware performance parameters corresponding to functional properties. Using the breadth first search algorithm based on hierarchical search to analysis the tree-like transaction-level dataflow graph, we can get the performance of reconfigurable processor running applications.

The rest of the paper is organized as follows: Sect.2 shows the structure feature of reconfigurable processor. Sect.3 introduces the generation of transaction-level dataflow. Sect.4 presents the performance analysis model and analysis algorithm, while experimental results are presented in Sect.5. Finally, conclusions are outlined in Sect.6.

Structure Feature of Reconfigurable Processor

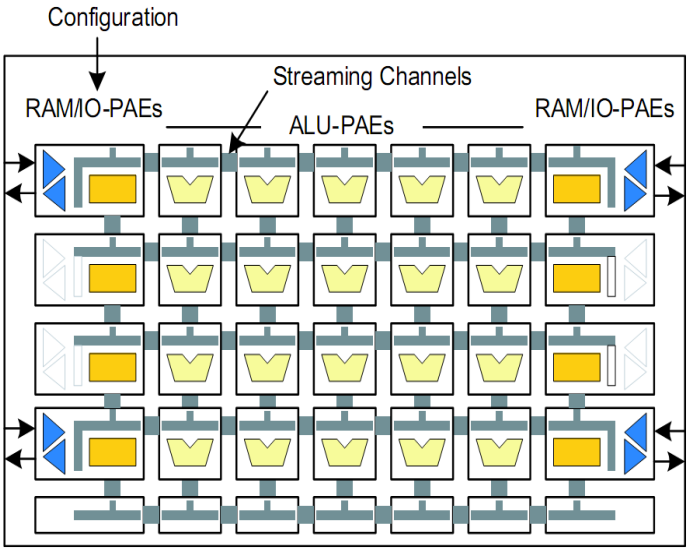

Fig.1. The XPP architecture

The major structure features of reconfigurable processor are the regular processing elements array and the dynamically reconfigurable data path. Figure1 shows a successful commercial CGRA named XPP [7]. XPP contains two types of processing element, ALU-PAE and RAM-PAE. The processing elements array is rectangle. ALU-PAEs is in the middle of array, which are responsible for calculation. And RAM-PAEs are on both sides for accessing the memory. The interconnect network of XPP forms a reconfigurable pipeline controlled by configuration. As we can see, the processing elements array is regular and the reconfigurable pipeline may change at runtime.

Figure2 illustrates the ACRP architecture designed by our research group [8] for low power consumption speech enhancement application. ACRP is mainly composed of customized processing elements (CPE), reconfigurable interconnect data channel (RDC), and data buffer memory (DBM). The CPE is customized with the operation, operation delay, operation grained and so on, based on the speech enhancement application. Some CPEs make up a reconfigurable pipeline, and all the reconfigurable pipelines form the ACRP. RDC is a cross switching network with multiplexer to exchange data between CPEs during running. DBM is a multiple functional register file (MFRF) [9] to support the data accessing from/to register file. All the CPEs are isomorphic, which means the architecture is regular, however we may custom the number of reconfigurable pipelines. In the same way, the RDC is also dynamically reconfigurable.

In this section, two CGRA are represented to explain the major structure features of reconfigurable processors, which are the regular processing elements array and the dynamically reconfigurable data path.

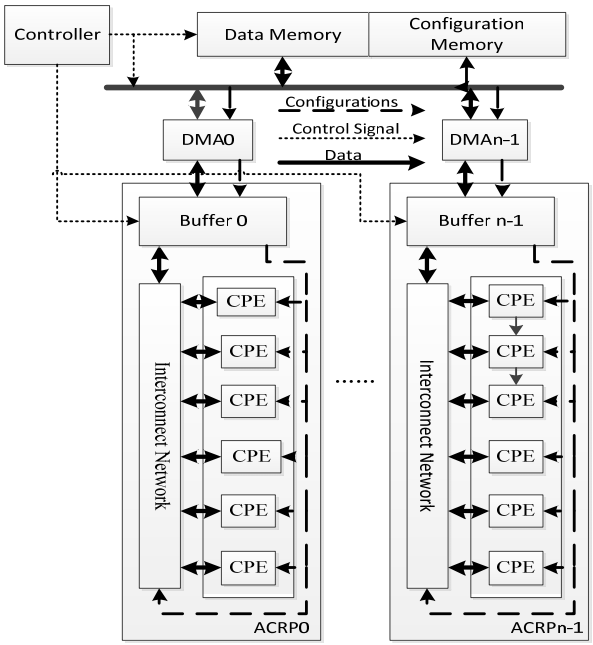

Fig.2. The ACRP architecture

Transaction-level Dataflow

- Dataflow Graph

Reconfigurable processor is usually used to accelerate the computation intensive application for specific domain. So the most important quota is the performance. The computation intensive application is represented with dataflow graph in general. Our performance analysis approach combines application data flow graph with reconfigurable architecture. Figure3 shows two typical applications in DSP, butterfly computation and first-order smooth filtering.

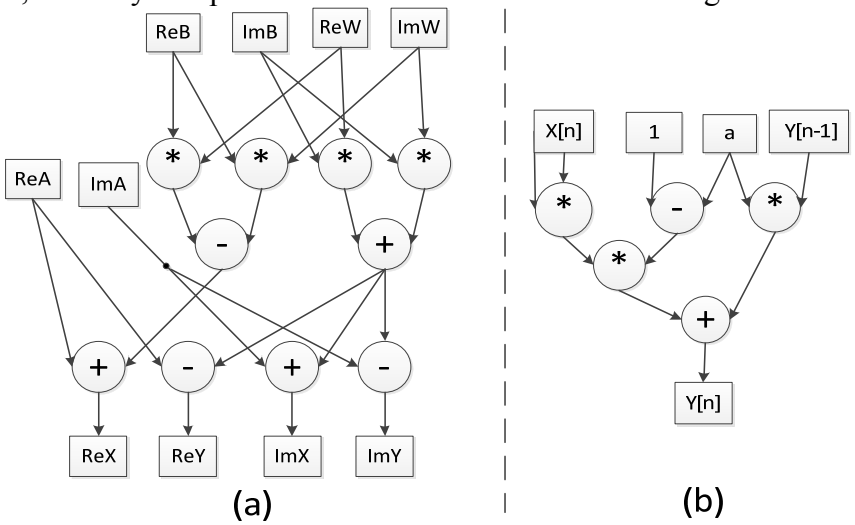

Fig.3. Dataflow graph of butterfly computation and first-order smooth filtering.

Dataflow graph is a directed acyclic graph. In DFG, a node represents an operation in application, and a directed edge is used to show the data stream between two operations. As we can see, the DFG is a tree-like structure. Although the DFG has nothing to do with the implementation of CGRA, when mapping the application onto CGRA, every node in DFG has to map a processing element. If the corresponding relationship is established between the DFG and CGRA, an estimated performance is obtained quickly using reasonable search algorithm.

- Transaction-level Dataflow Graph

To accelerate the analysis of performance of CGRA further and hide some details of DFG, firstly we introduce a transaction-level dataflow graph. In electronic system design, transaction is defined as an exchange between data communication. The kernel of transaction definition is the separation

of computation and communication.

Definition 1. Transaction-level Dataflow Graph (TDFG) is a hydra-graph, which is composed of hydra-node set (TN), hydra-edge set (TE), the input port of TN (IP) and the output port of TN (OP). In addition, the operation type and performance parameter is labeled to hydra-node.

Definition 2. Transaction-level Hydra-node(TN). Hydra-node contains not only the operation but also the input ports and output ports. An example of hydra-node is illustrated in Figure4.

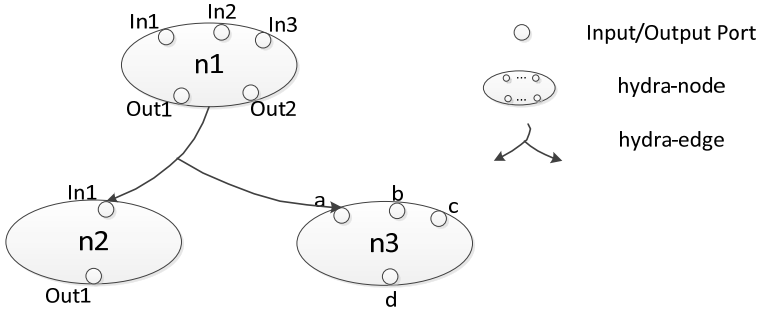

Fig.4. The example of Hydra-node

In Figure4, the hydra-node set is n1, n2 and n3. IP(n1)={n1.In1, In2, In3}, OP(n1)={n1.Out1, n1.Out2}, IP(n2)={n2.In1}, OP(n2)={n2.Out1}, IP(n3)={a, b, c}, OP(n3)={d}. There are some homonymy ports. In order to distinguish these homonymy ports, some ports is represented with its node. And we use IP(hydra-node a) to describe the input ports set of hydra-node a. In the same way, OP(hydra-node b) means the output ports set of hydra-node b.

Definition 3. Transaction-level Hydra-edge (TE)={t_e, h_e}, where t_e means the head end of hydra-edge, and h_e means the tail of hydra-edge. Hydra-edge is the connect edge between ports of hydra-nodes in TDFG. In Figure4, there are two TEs, {n1.Out2, n2.In1} and {n1.Out2, a}. Then the t_e and h_e can be figured out, $t_e \in \cup$ OP(hydra-node), $h_e \in \cup$ IP(hydra-node).

Until now, the TDFG only describes the logic relationship of application. To estimate the performance of application when mapping onto CGRA, the operation type and performance parameter must be labeled to the hydra-node.

- Generating method of TDFG

Because TDFG is refinement of DFG, the logic structure of TDFG is same as the DFG's. In essence, this refinement is an implementation of mapping the reconfigurable processor's hardware module onto the hydra-node of TDFG. For example, Figure5 shows the TDFG of butterfly computation when mapping onto ACRP reconfigurable processor.

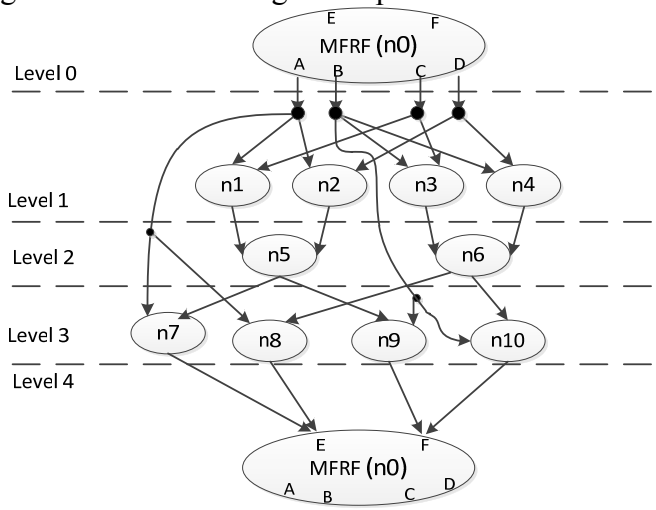

Fig.5. TDFG of butterfly computation extended from dataflow graph

In Figure5, n0 is mapped onto the storage node of MFRF in ACRP. This storage node has 4 read ports (A, B, C, D) and 2 write ports (E, F). n1~n10 is mapped onto the CPEs of ACRP.

Besides the operation and logic connection of TDFG, the performance parameters should be labeled to the TDFG. Performance parameters include:

- Operation Time, OT;
- Load Time (LT) and Store Time (ST) of the storage module;
- Configuration Time (CT). CT contains the read time of configuration and reconfiguration time of the data channel.

As shown above, different application leads to different refinement solution of ACRP. In general, the CPEs of ACRP are isomorphic, but their operation set may be customized according to the application domain. Similarly, we may choose MFRF or RAM to implement the storage unit of ACRP, and the reconfigurable interconnect network may be a crossbar network or routing network. Every implementation has different performance metrics. Table 1 illustrates these performance metrics.

Table.1. Lookup table of function and performance parameters

Module name	Function	Performance parameter
CPE0-CPE7	Addition Subtraction Multiplication ...	Add_Time Sub_Time Mul_Time ...
MFRF RAM	Read time Write time	Read_Time Write_Time
Reconfigurable crossbar network	Read configuration time	CR_Time
Reconfigurable routing network	Reconfiguration time	DR_Time

The representation in generation algorithm of the TDFG and DFG is adjacency list. As shown in Figure6.

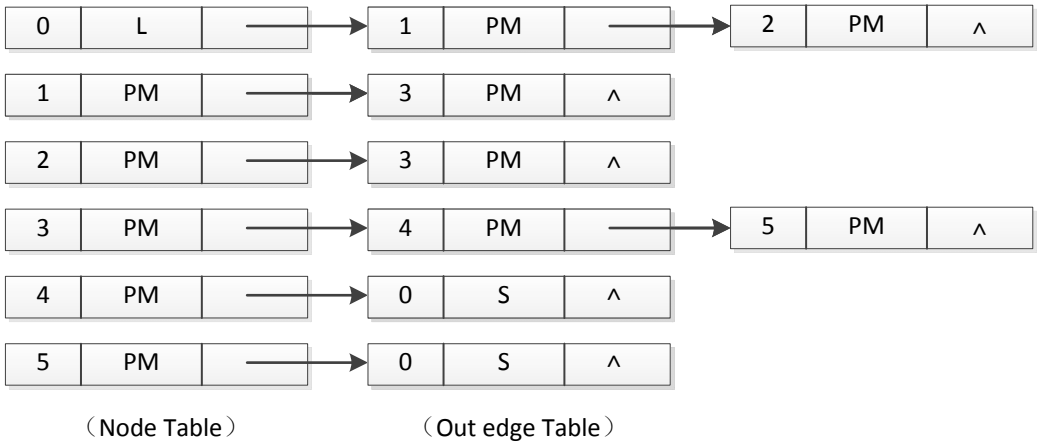

Fig.6. Adjacency list representation of transaction dataflow graph and dataflow graph

In the adjacency list representation of TDFG, every node has 3 fields, node index field, extensible performance analysis field PM and pointer filed to the next node. The node of index 0 represents the storage node, and the others are computing nodes. All the performance parameters described above, including operation style, delay and corresponding ports, are stored in the field PM.

Some compilers, for example LLVM [10], have the ability to generate DFG of application. On this basis, a pass is added to generate the TDFG, which transfers the intermediate representation of DFG to TDFG.

Performance Analysis Model and Analysis Algorithm

- Performance analysis model

On the basis of generating TDFG, merging some appropriate hydra-nodes in TDFG leads to a new performance analysis model of reconfigurable processor. Merge steps:

1, According to the data dependency of TDFG, divide all nodes into N layers. Nodes in the same layer may execute concurrently in principle.

2, Merge nodes layer by layer. If the number of CPE in ACRP was M, divide nodes in the same layer into some groups, and every group has M nodes. All nodes in group are merged into one super-node, and the performance metric are labeled with the longest delay of all nodes in this super-node.

3, Rearrange the super-nodes. Sorting super nodes, and if the hardware resource is not enough, super-nodes are sorted sequential else concurrently.

4, After sorting super-nodes, we get a performance analysis model. Figure7 illustrates the performance analysis model of butterfly computation, whose TDFG is shown in Figure5.

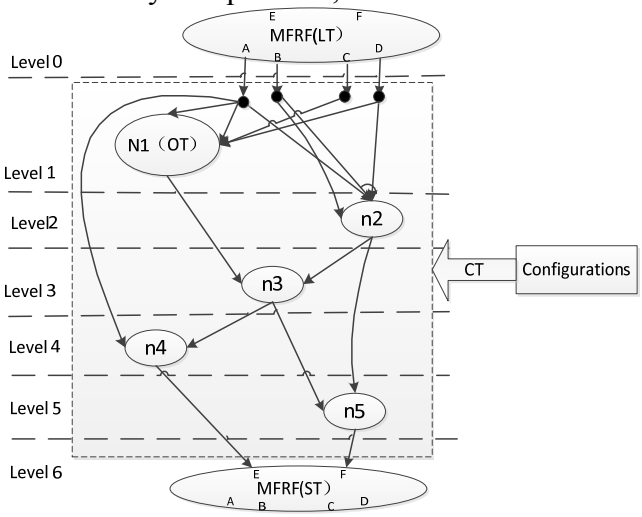

Fig.7. Performance model of butterfly computation

- Performance search and statistics algorithm based on floor traversing

According to the performance analysis model described above, we use a performance search and statistics algorithm based on floor traversing to obtain the estimated performance. Reconfigurable processor always targets computation intensive applications, which usually are the same as loop kernels. The typical loop is for-loop, whose cycle index can be computed beforehand. And as shown in Figure7, the performance analysis model is a tree-like structure, so we use breadth first search algorithm of floor traversing to analyze the performance of TDFG. The algorithm is as follows.

Performance search statistics algorithm based floor traversing of TDFG

```
Input: TDFG
Output: estimated running cycles, long

long TimeCount(TDFG *graph)
{
        Tree *tree=BuildSearchTree(TDFG *);
        Long layerTime[number of layers];
        foreach( layer i){
                layerTime[i]=max{labeled time of super-node of i layer};
        }
        index= cycle index of TDFG;
        loopTime= layerTime[0]+ layerTime[1]+... layerTime[number of layers];
        return loopTime*index;
}
```

Fig.8. Performance search statistics algorithm based on floor traversing

Experiment and Result Analysis

To verify the validity and accuracy of our method, we have implemented refinement ACRP for DSP domain with Verilog. And three computation intensive applications of DSP, points 8 FFT

based 2, first-order smooth filtering and matrix multiplication with size of 16×256 and 256×16, are mapped onto the ACRP, and their estimated performance is obtained by our method. The control group is the running time of three applications when simulating in RTL-level. The experiment software is Windows7, Visual C++6.0 and ModelSim10.1, and the experiment hardware is Intel CPU i5 3.20GHz.

The experimental ACRP contains 1 reconfigurable pipeline, and the reconfigurable pipeline has 8 CPEs. One MFRF is used to buffer data. The interconnect network is a crossbar network. When simulating three test benches in RTL-level, the configuration is generated by hand because there is not an automatic compiler and mapping tool for ACRP. This work is a big project, so we choose three simple test benches.

Firstly, we extract related performance parameters, for example, read/write delay of MFRF, operation delay, reconfiguration time and so on. All the performance parameters are illustrated in Table 2.

Table.2. Lookup table of basic performance parameters

Operation type	Running cycles
Store data	2
Load data	1
Addition	1
Subtraction	1
Multiplication	3
Read configuration	0
Reconfiguration data channel	0

On the basis of performance parameters above, the comparison of estimated running cycles using our method and the simulating running cycles in RTL-level model is shown in Table 3.

Table.3. Performance analysis result comparison: performance analysis method vs. RTL model

Applications	Estimated running cycles	Simulating running cycles	Accuracy rate（%）
First-order smooth filtering	9	9	100
8 points FFT	352	360	97.8
Matrix multiplication	7206	7424	97.1

As shown in Table 3, the accuracy rate of estimated running cycles is high compared to the simulating running cycles in RTL-level. The accuracy of First-order smooth filtering is 100%, and others are above 97%. Moreover, with the increase of computing scale, the accuracy slightly decreased.

The performance analysis method approached in this paper has not to map the application onto reconfigurable processor accurately. We estimate the performance based on TDFG and some performance parameters to avoid building the complicated RTL model and generating massive configurations by hand. This method benefits the optimization design of architecture and design space exploration at the beginning of design.

Conclusion

To overcome the shortage of traditional performance analysis method in flexibility and efficiency, we propose a novel method for reconfigurable processor performance analysis based on transaction-level dataflow graph. Experimental results show that the proposed method calculates the performance of reconfigurable processor quite accurately, and benefits the optimization design of architecture and design space exploration. The disadvantage of our work is that the labeling performance parameters should be done by hand until now. In further work we will research the

automatic labeling technology.

Acknowledgement

In this paper, the research was sponsored by National Natural Science Foundation of China No. 61133007.

References

[1] Wei Guo. Design method and Implementation of SoC[M]. Release 2.0. Electronics Industry Press. BeiJing. 2011. (In Chinese).

[2] Barreteau A, Le Nours S, Pasquier O. A Case Study of Simulation and Performance Evaluationof a SDR Baseband Architecture[J]. Journal of Signal Processing Systems, 2013, 73(3): 267-279.

[3] Gerstlauer A, Haubelt C, Pimentel A D, et al. Electronic system-level synthesis methodologies[J]. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 2009, 28(10):1517 - 1530.

[4] Haid W, Thiele L. Complex task activation schemes in system level performance analysis[C]. Codes+isss Proceedings of IEEE/ACM International Conference on Hardware/software Codesig. 2007:173 - 178.

[5] Le Nours S, Postula A, Bergmann N W. A dynamic computation method for fast and accurate performance evaluation of multi-core architectures[C]. Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014. IEEE, 2014: 1-6.

[6] Kunzli S, Poletti F, Benini L, et al. Combining Simulation and Formal Methods for System-Level Performance Analysis[J]. Proceedings of the Design Automation & Test in Europe Conference, 2006, 1:1 - 6.

[7] PACT XPP Technologies. The XPP White Paper. Release 2.0. Mar. 2002.

http://pactcorp.com/xneu/download/xpp_white_paper.pdf

[8] Wang Guanwu, Liu Lei, Li Sikun. ACRP: Application Customized Reconfigurable Pipeline[C].

Advanced Computer Architecture 10th Annual Conference(ACA2014), Shenyang, pp. 16-30. 2014.

[9] Kongfei Du, Guanwu Wang, Sikun Li. Multiple Functional Register File of Application Customized Pipeline[C]. The 2nd International Conference on Computer science and Information technology(ICCIT 2015). Shanghai. 2015. (In Chinese).

[10] LLVM: http://www.llvm.org.

[10] JinhuiXu, Guiming Wu, Yong Dou and Yazhuo Dong. Designing a Coarse-Grained Reconfigurable Architecture Using Loop Self-Pipelining.Advances in Computer Systems Architecture, Springer Berlin Heidelberg. Volume 4186:567-573, 2006

[11] MA Kel,ZHANG Longbin. A Microprocessor Performance Analysis ApproachBased on Microbench and ldeaIBound.Acta Electronica Sinica,Chinese Journal of Electronics2008 36(2). (In Chinese).