2012 IEEE 18th International Conference on Parallel and Distributed Systems

ProDFA: Accelerating Domain Applications with a Coarse-Grained Runtime Reconfigurable Architecture

Ming Yan
Northwest Institute of Nuclear Technology
Xián, Shaanxi, China
yan.meen@gmail.com

Ziyu Yang, Lei Liu, Sikun Li
National University of Defense Technology
Changsha, Hunan, China
zyyang@nudt.edu.cn lisikun@263.net.cn

***Abstract*—With the on-chip resources largely increased, modern architectures are much different from traditional ones. The relation between temporal computing and spatial computing is getting more and more intricate. In this paper, we first analyzed and compared the abstract computing models of modern architectures. Then, a runtime reconfigurable architecture called programmable dataflow computing architecture (ProDFA) is proposed. The architecture of ProDFA is sumarized, then the process of how applications are mapped and execute are simply introduced. As a case study, a specific reconfigurable structure for symmetric ciphers is implemented. Performance of several typical symmetric ciphers are evaluated. The experimental results show high performance and efficiency of ProDFA.**[1]

***Keywords*-computing paradigm; coarse grain architecture; reconfigurable computing; spatial computing; temporal computing; run-time reconfigurable architecture**

I. Introduction

Instruction-stream-based computing architecture has lead the processor architecture for decades. With the on-chip resource largely increased, new complex architectures are proposed and some are becoming mainstream architectures such as stream architecture in desktop computer and reconfigurable architecture in embedded systems. Many novel architectures have also been proposed to solve the "memory wall" problem or the "power wall" problem of traditional instruction stream based computing architecture, such as Processor-in-Memory architecture VIRAM [1], Explicit Data Graph Execution architecture TRIPS [2], Reconfigurable Streaming Vector Processor RSVP [3], etc. With all the development, the relation between instruction stream-based computing and data stream-based computing is becoming more complicated than ever before.

Reconfigurable computing architecture (RCA) has been studied for many years. RCAs are usually classified into fine-grained reconfigurable architecture or coarse-grained architecture according to the configure granularity only. Coarse-grained reconfigurable architecture (CGRA) has been researched for decades. A lot of architectures have been proposed and many of them have been taped out for commercial applications, such as MorphSys PipeRench and PACT XPP, etc. Most of these projects use an array structure and concentrate on the specific structure and the function units. Recently, some novel architectures have been proposed and paid more attention to the programmability also known as runtime reconfigurability of RCA, such as Vector Thread Architecture [9],PPA [6], ADRES [11], RICA [12] etc. ADRES uses a tightly coupled VLIW processor with a reconfigurable function unit array. The configuration of reconfigurable array is controlled by the intruction unit of VLIW processor. The mixed instruction flow and configuration flow leads to a difficulty in the data communication and synchronization between the VLIW pipeline and the reconfigurable array. RICA gives a new approach to achieve high performance and low power consumption for domain-specific applications. The main idea of the RICA architecture is to enable parallel processing of many function units. The reconfigurable array is composed of many different instruction cells. The heterogeneous array has a Harvard-like structure and the configuration is controlled by a program execution controller which fetches the assemble code and reconfigures the structure. As the same as ADRES, the centralized control scheme leads to difficulty in the dynamic reconfiguration during computing and low resource utilize efficiency.

In this paper, a novel reconfigurable architecture named ProDFA is proposed. The main contribution of our work include:

- A new classification of modern architectures is proposed based on the control properties in section 2.
- The structure and execution model of ProDFA is summarized in section 3. The application mapping and optimization is also introduced.
- A case study on symmetric ciphers are described in section 4. The programmability of ProDFA are tested through several typical block ciphers. Experimental results of several block ciphers are analyzed.

[1]This paper work is finantially supported by National Science Foundation of China with grant number 61076020.

1521-9097/12 $26.00 © 2012 IEEE
DOI 10.1109/ICPADS.2012.136

II. Analysis and Comparison of Abstract Models

With the development of processor architecture and manufacture technology, the abstract models of modern architectures are getting more hybrided.

A. Classification According to Control Properties

The instruction-stream-based architecture is classified according to the number of instruction-stream and data-stream by Flynn et al. [4]. Since processor architecture has been developed into a multicore era, this classification is not accurate enough to distinguish different architectures. We give a new classification according to the properties of control logic.

We focused on two basic properties of control logic: the control interval and the control term. Control interval is the minimum clock cycles which is needed to change the dataflow structure of function units and their interconnection. Control interval is a temporal computing property of an architecture. For example, the control interval of scalar instruction pipeline is one cycle because the dataflow structure has to be changed according to the instruction issued every cycle; the control interval of reconfigurable computing array with a global static configure scheme is the cost of reconfiguration of the whole array which is usually tens of clock cycles. According to the control interval, architectures could be classified into fine-grained-control architecture and coarse-grained-control architecture.

The control term is the maximum clock cycles during which the dataflow structure of all function units could be kept unchanged. The control term property mostly decides the efficiency and performance of function units. It is a spatial computing property of an architecture. According to the control term, architectures could be classified into short-term-control architecture and long-term-control architecture.

With these two control properties, modern architectures could be classified as Figure 1. Obviously, for a given architecture, the control term should always be larger than or equal to the control interval. Small control interval could bring flexible programmability (for ISA) or run-time reconfigurability (for RCA), and large control term makes high efficiency computing possible for an architecture.

III. ProDFA: The Architecture Summary

The basic idea of ProDFA is to use a distributed configure scheme which brings the temporal flexibility of several function units (FUs) and maintains the high efficiency of spatial computing. With a distributed structure, the control granularity could be minimized to one cycle just as the instruction-stream-based architecture. And the long-term control ability will leads to higher performance and efficiency than the instruction pipelines.

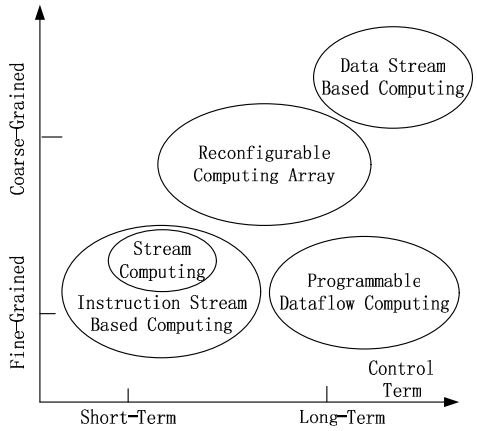


Figure 1. The classification of modern architectures

A. Overall Architecture of ProDFA

The overall structure of ProDFA is showed as Figure 2. There are four sub-figures in Figure 3. The sub-figure A of Figure 3 shows the micro-architecture of one reconfigurable processing unit (RPU) of ProDFA. The sub-figure B is the structure of a reconfigurable processing node(RPN) which has four RPUs clustered. The sub-figure C is an array structure composed of RPNs. And the sub-figure D shows the structure of the configurable application-specific function unit (CAFU). With a hierarchical structure, the ProDFA could be expanded to large scale when the application requires for more powerful performance. The programmability and efficiency is maintained through the RPU which is the basic element of ProDFA. There are mainly three parts inside a RPU showed as sub-figure A: the control and configure part showed in the left blue dots, the execution part in the center green dots, and the expandable interface part in the right red dots. The basic idea of programmable dataflow architecture is to enable each function unit and memory unit with the ability of self-control, thus they do not need to be configured every cycle. The most simple way is to register the control information in each function unit and reuse the information until reconfigured.

The control and configure part is composed of a configware memory (CRAM), a runtime control Finite-State-Mechine (FSM), a control-status-register (CSR) and an execution context buffer. Configware memory holds the configware of the application dispatched by the host processor. The runtime control FSM is just like an instruction decoder which translates the configware into configuration and interconnection information of function units to construct a dataflow structure inside the RPU. The CSR allows the host processor to check on the execution status of the RPU and to control the RPU from outside. The control part supports every-cycle-control of the execution with much simpler control logic compared to VLIW or EPIC architecture. Inside the

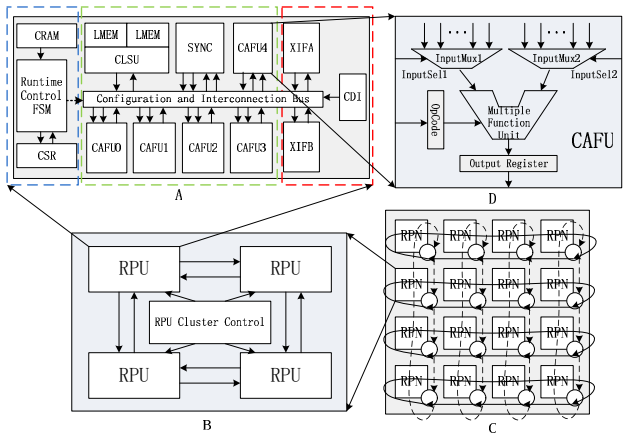


Figure 2. The overall architecture of ProDFA, A: the micro architecture of the ProDFA reconfigurable processing unit (RPU); B: the structure of a RPN composed of clustered ProDFA RPUs; C: the array structure composed of RPNs; D: the structure of the CAFU

runtime control FSM, there is an execution context buffer. The execution context buffer is composed of several wide registers which hold all the functions and interconnection information of all the CAFUs and other units inside the RPU. Each wide register holds an execution context. All the functions and interconnection could be reconfigured within one clock cycle with the support of execution context buffer.

The execution part is the execution engine of the RPU, which concludes several configurable application specific function units (CAFU), local data memory (LMEM), configuration and interconnection bus (CIB) and a synchronization unit (SYNC) used to synchronize the dataflow inside the RPU. The CAFUs are specially designed according to domain-specific applications, and each has a configure port which could register the function and the input data connection information issued from the runtime control FSM. LMEM holds the source data and results. CIB is composed of the configuration of all the operation and input selection of function units. The dataflow structure of the RPU could be reconfigured during computing through CAFU and CIB which enable the RPU with reconfigurable function and interconnection. SYNC unit is used to synchronize data transferred between function units. The SYNC unit is very important to ProDFA because there are no general register file to hold and transfer data between operations inside RPU. The structure of CAFU is showed as Fig.2 part D. The input of CAFU is selected from the output bus which concludes all the output of CAFUs in RPU. The input select information and opcode of each CAFU are issued from the runtime control unit and registered in each CAFU.

The expandable interface part is composed of three input-output interfaces: external interface A (XIFA), external interface B (XIFB) and cluster data input (CDI). The external interface A and B are two pairs of data input output channels which connect one RPU to its neighbors using a port mapping scheme. The data comes from XIF is treated as a selectable input operand to all CAFUs. With these two channels, RPUs could be connected to form a large array or cluster to get more powerful performance. And the port mapping scheme allows multiple RPUs to glue their datapath pipelines to support more complicate computation process. Cluster data input (CDI) is a special data input interface which supports data broadcasting when RPUs are clustered. The data comes from CDI is connected to one CAFUs which support dedicated input operand.

B. The Execution Context of ProDFA

Application mapping of ProDFA is based on the execution context. Compared to traditional reconfigurable arrays which usually uses a centralized configure unit to manage the execution context of all the FU array, we use a distributed execution context for each RPU of ProDFA. Once the RPU is configured, a dataflow structure is formed which could be represented as a structural dataflow graph. The functions and interconnection of all the FUs in one RPU is treated as an execution context. The execution context is supported by context buffer in order to support zero-delay context switching. There is one context buffer inside the control unit of each RPU.

With the support of execution context and CDL, applications could be written in CDL by hand or compiled to CDL descriptions. The programmer or the compiler could use the reconfigurable hardware of ProDFA efficiently with nearly zero reconfigure overhead and fully pipelined computation. The CDL description of DES algorithm showed in Figure 5 is an example of pipelined computation on a block cipher specific reconfigurable structure which will be described later. The cipher key schedule algorithm is not described in this CDL code because the key schedule is processed in the cluster control unit.

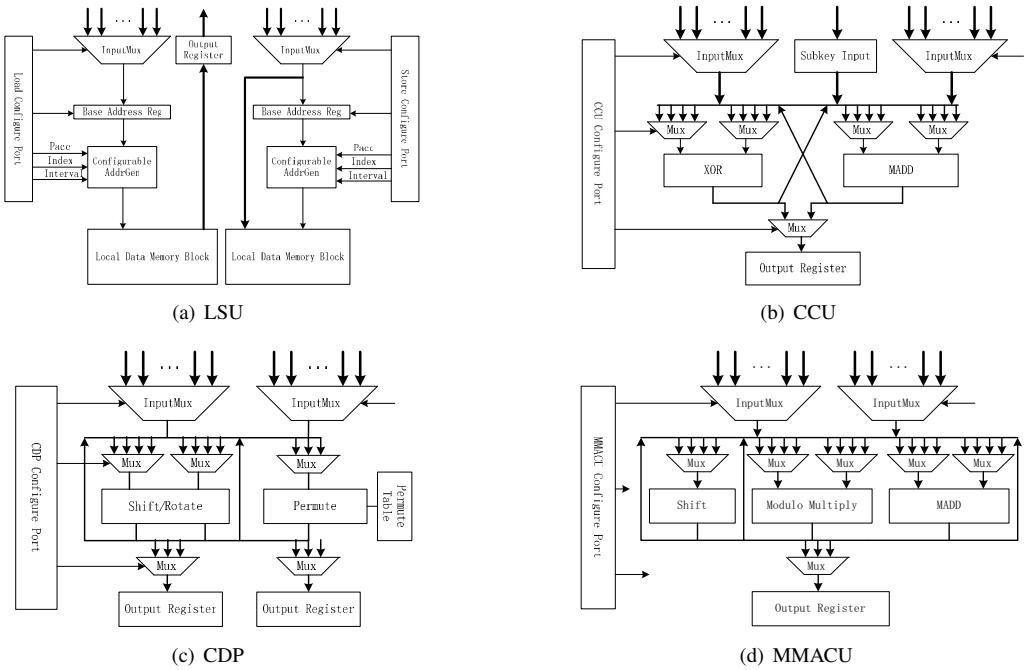
IV. Case Study: A Specific Reconfigurable Structure for Symmetric Ciphers

A. The structure of Reconfigurable SoC

In order to evaluate the performance and programmability of ProDFA, a specific reconfigurable structure for block ciphers is designed and implemented. The overall structure of the block cipher specific reconfigurable SoC is composed of a host processor system and a crypto coprocessor system. The host processor system has a OpenRISC core and its peripherals such as memory interface and general purpose IO, etc. The coprocessor has its own memory interface too.

B. Block Cipher Specific Design

The block cipher specific reconfigurable coprocessor has a structure similar to the part B of Figure 2. It consists of four crypto RPUs and one cluster control unit which is extended with subkey generation ability. The cluster control



(a) LSU (b) CCU

(c) CDP (d) MMACU

Figure 3. Structure of some symmetric cipher specific function units

and subkey generate unit has a ProDFA-based structure but with fewer function units and no external interfaces (XIF) instead of a cluster data output interface to send subkeys to crypto RPUs.

The specific functions are designed using synthesizable verilog HDL. In the crypto reconfigurable processing unit (CRPU), there are six function units specially designed for block ciphers, some of which are showed as figure 3:

***load-store unit* (LDU and STU)**: It is used to load data from local memory(LMEM) or store results to LMEM showed as subfiure a in figure 3. The configurable address generator supports three different mode: indexed mode, pacing mode and interval mode. Indexed mode is used when source data is interleaved in local memory. Pacing mode is useful when working with different data block size. Interval mode is used to configure the frequency of data load or store so as to synchronize the data flow with other function units.

***configurable computing unit*(CCU)**: CCU has two parallel datapath which supports MADD, XOR, etc.. CCU is a simple ALU showed as subfigure b in figure 3. The operand of XOR or MADD could be configured and the output could be selected and registered.

***configurable substitute unit*(CSU)**: CSU has a SBOX structure implemented with configurable multiple SRAM blocks. The SRAM blocks are grouped with each block receives a input data as an address. The SBOX could be written in the SRAM and the substitution is done in one clock cycle with a memory read operation.

***configurable data permute unit*(CDP)**: CDP supports data shuffle and select. Data shift or rotate is a common operation in symmetric ciphers. The structure of CDP is showed as subfigure c in figure 3. The permute unit shuffles input data according to the permute table which is wrote before computing started.

***modular multiple and accumulate unit*(MMACU)**: MMACU is a modulo multiply and accumulate unit that supports four input operands. The structure is showed as subfigure d in figure 3. The MMACU has three sub-units and supports a four bits opcode and operand selection configure.

***synchronization unit*(SYNC)**: SYNC unit is a group of registers which delays the input according to the interconnection. It is used to synchronize data flow between different function units inside each CRPU.

C. Implementation and Experiment Results

As a case study, the crypto specific SoC is implemented with standard cell library to estimate the area cost and timing information which is important to evaluate the performance and power consumption. We evaluated several typical symmetric ciphers on the cipher specific reconfigurable SoC(short for ProDFA), such as DES, AES, IDEA, Twofish, RC6. Performance and efficiency of ProDFA are than compared with some state-of-the-art designs.

ASIC Implementation Detail

Showed as Figure 4, the block cipher specific SoC is implemented with Artison's 0.13um CMOS standard cell library for SMIC. The area costs and timing reports of the worst case of every unit are given. The SoC is written

Unit	Area (um^2)	Timing (ns)	Unit	Area (um^2)	Timing (ns)
OpenRISC	453679	2.89	LSU	6214	1.43
Coprocessor	718592	2.43	KDU	1850	1.33
Crypto RPU	159571	2.43	MMACU	45111	2.35
ClusterControl	69508	2.21	SYNC	35214	0.95
CCU	8497	1.32	FSM	7589	2.17
CSU	39316	1.37	CSR	396	0.73
CDP	5943	2.11	SHU	5099	0.97
SPU	4900	1.12	SBU	38625	1.35
WB_Conbus	8253	2.01	SDRAM	11508	2.01

Figure 4. ASIC implementation detail of the crypto reconfigurable SoC

Ciphers	Implementation	Cycles/Block	Throughput (Mbps)	Cycles	Mode
DES	ProDFA	5.7	4491	75+19	4×4
AES-128	ProDFA(Parallel)	12.1	4231	80+17	2×4
	ProDFA(Sequential)	12.3	4163	120+27	6×2
IDEA	ProDFA(Parallel)	10.4	2461	72+11	2×4
	ProDFA(Sequential)	7.25	3531	72+15	6×2
Twofish	ProDFA(Parallel)	13.6	3764	96+13	2×4
	ProDFA(Sequential)	9.58	5344	96+19	6×2
RC6	ProDFA(Parallel)	13.4	3821	96+11	2×4
	ProDFA(Sequential)	9.38	5446	96+17	6×2

Figure 5. Performance of ProDFA evaluated with several typical Symmetric Ciphers

	Frequency	Area Cost	Cycles/Block	Throughput	Architecture
ProDFA	400MHz	0.72mm^2	12.1	4.2Gbps	Reconfigurable
CORBA[21]	102MHz	2.5Mgates	9	1.45Gbps	Reconfigurable
Cryptonite[19]	400MHz	-	70	700Mbps	ASIP
CCproc[22]	250MHz	1.46mm^2	79	810Mbps	ASIP
AES Coproc[23]	330MHz	0.79mm^2	11	3.84Gbps	ASP
Cryptomaniac[20]	360MHz	1.93mm^2	90	512Mbps	VLIW
FPGA[14]	-	-	-	11.7Gbps	ASIC[1]

Figure 6. AES performance comparison between some block cipher specific architectures

in synthesizable verilog HDL, synthesized with Design Compiler of Synoposys, and routed with NanoRoute with Cadence. The area cost of one crypto RPU which has comparable performance of ASIC designs, is much smaller than OpenRISC . The OpenRISC is integrated with a 32KB Cache and the on-chip memory concluded in each crypto RPU is 16KB. The area cost of the on-chip memory is not counted.

Showed as figure 4, compared to the OpenRISC core, the crypto coprocessor is about 1.5 times larger with four crypto RPUs and on cluster control unit inside. Note: the area cost of each unit do not conclude the area cost by memory such as SRAM in CSU and LDU, etc. The timing of coprocessor is better than OpenRISC core because of a much simple datapath logic design. The critical datapath of coprocessor resides in the MMACU which has a modulo multiply unit and huge MUX. Through the timing report with worst case condition, the performance of the crypto coprocessor could be estimated with 400MHz.

Performance of ProDFA on Typical Symmetric Ciphers

The experiment results of block ciphers is showed as Figure 5. Because there are four CRPUs inside one node, these four nodes could be working in parallel or sequential. Parallel mode means the CRPUs work independently with the same subkey. The subkey is broadcast to the four CRPUs. Sequential mode means there is data dependency between these four CRPUs, the operations of four CRPUs need to be synchronized. Subkey is transferred from cluster control unit to each CRPU independently with a subkey request from each CRPU. With different execution mode, the performance of ProDFA is veried. For example when AES is processed in parallel mode, two data blocks are processed in one CRPU and four CRPU could be working parallelly. In sequencial mode, six data blocks are processed in two CRPUs which is chained to pipeline the round computation of AES, so that the four CRPUs are divided into to independent pipelines.

When DES is mapped to the CRPU of ProDFA, 4 data blocks could be processed simultaneously inside one CRPU with a pipelined procedure. 16 round computation need 75 clock cycles. Added with time cost of configuration and key schedule overhead which is 19 clock cycles total, it costs 94 clock cycles to process 4 data blocks per CRPU. When AES is tested parallelly, two data blocks are processed simultaneously in one CRPU. The number of data blocks being processed simultaneously with IDEA is 4 data blocks, with Twofish is 2 blocks. With different ciphers, the clock cycles needed for each round computation is different. All the block ciphers are tested in non-feedback mode.

Showed as Figure 5, the performance in sequential mode is mostly bettern than parallel mode, such as in IDEA, Twofish and RC6 tests. This is mainly because sequential mode uses the configurable interconnection between CRPUs and increased the overall resource utilization. The results of throughput showed that the crypto coprocessor has very good performance with all the five tested ciphers, which proves the programmability of the coprocessor indirectly.

AES Performance Comparison and Analysis

AES is the most popular symmetric cipher which is very widely used in safety applications. The AES performance of ProDFA is compared with several cipher specific state-of-the-art designs, showed as figure 6. The Cryptonite [14] and CCproc [17] are application specific instruction-set processors (ASIP) that are optimized for AES algorithm. Cryptomaniac [15] is a crypto specific processor with a VLIW architecture. CORBA [16] is a reconfigurable array structure optimized for crypto algorithms. AES Coproce [18] is a application specific processor(ASP) which is design dedicated for AES algorithm. FPGA [10] is a fully pipelined implementation of AES which could be treated as an ASIC dedicated for AES.

The CORBA structure is implemented with 0.25um C-

MOS standard cell library. The estimated number of gates is about 2.5M which is about $12.2mm^2$ with 0.13um library. (Note: one NAND with 0.13um library costs about $4.9um^2$). Compared with CORBA, the ProDFA has much better performance and smaller area costs. Compared with ASIP structures and VLIW based crypto processor, the ProDFA has much better performance. The AES performance of ProDFA is very close to the AES Coproc. The area costs of these two design are also very similar. But the ProDFA has a much flexible programmability while other ciphers are executed such as DES, Twofish, etc.

V. Conclusion

A coarse-grained runtime reconfigurable architecture named programmable dataflow architecture (ProDFA) is proposed. The programming interface of ProDFA is described which includes the execution context model and CDL. With the support of execution context and CDL, ProDFA has a much more flexible programmability compared to traditional reconfigurable arrays. A block cipher specific reconfigurable SoC is designed and implemented. Meanwhile the performance and efficiency of ProDFA is comparable with algorithm specific designs. Several block ciphers are evaluated. The experiment results showed the high performance and efficiency of the cryptographic SoC.

References

[1] Joseph Gebis, Sam Williams, David Patterson, Christos Kozyrakis, "VIRAM1: A Media-Oriented Vector Processor with Embedded DRAM," *DAC '04: Proceedings of the 41th annual Design Automation Conference*, pp., 2004.

[2] D. Burger, S. Keckler, K. McKinley, M. Dahlin, L. John, C. Lin, C. Moore, J. Burrill, R. McDonald, and W. Yoder, "Scaling to the end of silicon with EDGE architectures," *IEEE Computer*, 37(7):44-55, July 2004.

[3] S. Ciricescu, R. Essick, B. Lucas, P. May, et al., "The Reconfigurable Streaming Vector Processor (RSVP)," *MICRO 36: Proceedings of the 36nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. , 2003.

[4] Flynn, and Michael J., " Some Computer Organizations and Their Effectiveness," *IEEE Transactions on Computers*, Vol.C-21(9):948-960, 1972.

[5] J. Gyilenhaal, B. Rau, and W. Hwu, "HMDES version 2.0 specification, IMPACT" *University of Illinois, Urbana, IL, Tech. Rep*, IMPACT-96-03, 1996.

[6] H. Park and Y. Park and S. Mahlke, "Polymorphic pipeline array: a flexible multicore accelerator with virtualized execution for mobile multimedia applications," *MICRO 42: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 370–380, 2009.

[7] L. Bauer and M. Shafique and S. Kramer and J. Henkel, "RISPP: rotating instruction set processing platform," *DAC '07: Proceedings of the 44th annual Design Automation Conference*, pp. 791–796, 2007.

[8] J.M. Moya and J. Rodríguez and J. Martín and J.C. Vallejo, et al., "SORU: A Reconfigurable Vector Unit for Adaptable Embedded Systems," *ARC '09: Proceedings of the 5th International Workshop on Reconfigurable Computing: Architectures, Tools and Applications*, pp. 255–260, 2009.

[9] R. Krashinsky and C. Batten and M. Hampton and S. Gerding, et al., "The Vector-Thread Architecture," *ISCA '99: Proceedings of the 26th annual international symposium on Computer architecture*, pp. 28–39, 1999.

[10] F. X. Standaert, G. Rouvroy, J. J. Quisquater, etc., "Efficient Implementation of Rijndael Encryption in Reconfigurable Hardware: Improvements and Design Tradeoffs," *CHES 2003: Cryptographic Hardware and Embedded Systems*, pp. 334–350, 2003.

[11] B. Mei and S. Vernalde and D. Verkest and H.D.Man and R. Lauwereins, "ADRES An Architecture with Tightly Coupled VLIW Processor and Coarse-Grained Reconfigurable Matrix," *FPL '03: 13th International Conference on. Field Programmable Logic and Applications*, pp. 61–70, 2003.

[12] S. Khawam and I. Nousias and M. Milward and Y. Yi and M. Muir and T. Arslan "The Reconfigurable Instruction Cell Array," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 75-85, VOL. 16, NO. 1, 2008.

[13] Mosanya E., Teuscher C., Restrepo H.F., Galley P., Sanchez E., " Cryptobooster: A reconfigurable and modular cryptographic coprocessor", *CHES '99: Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems*, pp.246-256, 1999.

[14] Buchty R., Heintze N., Oliva D., "Cryptonite: A Programmable Crypto Processor Architecture for High-Bandwidth Applications", *ARCS 2004: Organic and Pervasive Computing*, pp.184-198, 2004.

[15] Wu L., Weaver C., Austin T., "CryptoManiac: a fast flexible architecture for secure communication", *SIGARCH Computer Architecture News*, pp.110-119, 2001.

[16] Elbirt A. J., Paar C., "An Instruction-Level Distributed Processor for Symmetric-Key Cryptography", *IEEE Trans on Parallel and Distributed Systems*, pp.13-26, 2005.

[17] Dimitris T., Ros S. Dionisis P., "CCproc: A custom VLIW cryptography co-processor for symmetric-key ciphers", *ARC 2009: The 5th International Workshop on Applied Reconfigurable Computing*, pp.6, 2009.

[18] Alireza H., David D. H., Bocheng L., Kris T., Ingrid V., "A 3.84 Gbps AES crypto coprocessor with modes of operation in a 0.18um CMOS technology", *GLSVLSI '05: Proceedings of the 15th ACM Great Lakes symposium on VLSI*, pp.60-63, 2005.

[19] Antti Hamalainen, Matti Tommiska, and Jorma Skytt, "6.78 Gigabits per Second Implementation of the IDEA Cryptographic Algorithm", *FPL 2002: Proceedings of the 12th Conference on Field-Programmable Logic and Applications*, pp.760-769, 2002.