SMARTCELL: A POWER-EFFICIENT RECONFIGURABLE ARCHITECTURE FOR DATA STREAMING APPLICATIONS

Cao Liang and Xinming Huang

Department of Electrical and Computer Engineering
Worcester Polytechnic Institute
Worcester, MA, USA
{cliang, xhuang}@wpi.edu

ABSTRACT

This paper presents SmartCell as a novel power efficient reconfigurable architecture targeted for data streaming applications. We describe the design details of the SmartCell architecture, including processing element, reconfigurable interconnection fabrics, instruction and control process and dynamic configuration scheme. The performance in terms of power efficiency and system throughput is evaluated through a set of benchmark applications, and is compared with ASIC, FPGA and RaPiD reconfigurable architecture. The results show that the SmartCell consumes about 52% and 75% less power than RaPiD and FPGA, respectively. It is demonstrated that SmartCell is a promising reconfigurable, power efficient and scalable computing architecture that can potentially bridge the gap between logic specific ASIC and configurable FPGA for data streaming applications.

Index Terms— DSP, ASIC, FPGA, CGRA, power efficiency, reconfigurability, data streaming application

1. INTRODUCTION

Reconfigurable architectures have long been proposed as a solution to achieve a balance between flexibility of processors and performance as well as power efficiency of ASICs. The field programmable gate arrays (FPGAs) are still the dominating technology in the reconfigurable computing area. The bit-level fine-grained architecture is suitable to implement a large variety of functions directly. However, this flexibility comes at a significant cost in terms of area, power consumption and speed, due to the excessive routing area overhead and timing penalty.

In recognizing these issues, several researches have been conducted in recent years to develop more coarse-grained reconfigurable operators as the basis of computing architectures for certain application domains, as in [1-3]. Because these coarse-grained reconfigurable architectures (CGRAs) have much less computing and routing overheads, they have the potential to improve the power and area efficiency and to expedite the configuration process.

This paper presents SmartCell – a novel CGRA for high performance low power signal and image processing and for data streaming applications in general. SmartCell integrates a large number of tiny processor cores (cells) onto a single chip. The cells are interconnected through three levels of programmable switching fabrics. The number of processor elements involved in computing tasks can be dynamically changed to meet different application requirements. For example, more cells can be involved to achieve higher computational performance, while fewer cells are activated when power consumption becomes more stringent. SmartCell can be configured to operate in various computing styles, such as SIMD, MIMD and systolic array fashions, targeted at applications with inherent data-parallelism and high computing and communication regularities.

The rest of paper is organized as follows. After describing the background of some existing reconfigurable computing platforms in Section 2, the design details of the SmartCell architecture is presented in Section 3. Next, we compare the SmartCell system with several other CGRAs in Section 4. Section 5 presents the implementation of a prototype SmartCell system with 64 processor elements along with its performance evaluations using benchmarks, followed by the conclusions and future work in Section 6.

2. BACKGROUND OF EXISTING COMPUTING PLATFORMS

In this section, we briefly summarize the differences and similarities between the SmartCell architecture and several other computing platforms, including FPGAs, VLIW processors, systolic arrays, and SIMD processors. A comparison with other CGRAs will be discussed in Section 4 after presenting the design details in Section 3.

Traditional FPGA architectures use the static configuration streams to control the functional units and routing resources to perform user specifications. Due to the direct mapping on the hardware resources, FPGA is able to

avoid the overhead of instruction fetching, decoding, and executing as that in a general purpose processor. Similar features can be found in the SmartCell system. However, as we mentioned previously, the fine-grained granularity and the general purpose Lookup Tables (LUTs) in FPGAs involve high routing overheads and lots of controlling requirements, which in turn degrades the performance and makes the compilation and configuration processes very slow. In realizing these problems, the commercial FPGA vendors have also introduced more coarse-grained components as the computing basics in their newly developed FPGAs. The Virtex-4 and 5 series [4] are among the latest Xilinx FPGAs, which have a mixed granularity of basic logic cells with coarse-grained DSP slices. In addition, 6-input LUTs are also introduced to the Virtex 5 FPGAs to replace the traditional 4-input LUTs. This is helpful to reduce the routing overhead and to ease the configuration process to some extent. But the SRAM-based FPGAs have some fundamental limits that hamper them becoming mainstream computing media for the data streaming applications. Traditional FPGAs do not support instruction sequencing (except for the embedded DSPs) and are thus infeasible or very costly to make any changes on the fly. The SmartCell system can easily achieve the online flexibility by pointing to a new instruction code.

SmartCell also shares some common features with VLIW processors [5]. Both architectures group multiple tiny functional units together for high performance computations, and use long instruction format for the logic and interconnection controls. The data flow scheduling and the parallelism are both exploited at compiling time. But the SmartCell distinguishes itself from the VLIW processors with some major differences. First of all, VLIW system performs a single stream of execution. SmartCell, however, has the choice to set multiple instruction streams that allow concurrent execution of independent computations. Secondly, the tiled structure of the SmartCell also provides an efficient data exchanging scheme among the processing units. Furthermore, the number of involved computing elements in the SmartCell architecture can be adjusted to meet different system requirements.

The SIMD machines, such as Intel's MMX [6], are characterized by their data level parallelism on multiple processing units controlled by the same instruction code. But the exploration of the data parallelism is not trivial in the traditional SIMD processors. The cost of data alignment among the SIMD registers is essential to their performance. In our design, the SmartCell can be configured to operate in the SIMD style, but not limited to. Specifically, the SmartCell can also be configured to operate in MIMD and systolic array fashions on a per application basis. At the same time, the SmartCell incorporates a three-level layered connection fabric that can be dynamically configured to provide more flexible data flows. This makes the task level pipeline available for many multimedia applications.

Another approach to data streaming processing is to use systolic array architectures [7], in which the data processing units (DPUs) are arranged in an array structure with local connectivity. The operations of DPUs are scheduled by regular data flows in a pipelined manner. The input/output pipeline and the concurrent data execution supports extremely high throughput. The systolic arrays also provide very high scalability due to their regular structure. SmartCell shares similar principle with systolic arrays. However, due to the fixed configurations, the cost of introducing new pipeline patterns in traditional systolic arrays is very high. In addition, the synchronization process may be very complicated in the systolic array implementations. On the other hand, the SmartCell can be easily configured to operate in 1D or 2D systolic array. New pipeline can be added by simply loading new instructions into the SmartCell instruction memories. Furthermore, the flexible control of the logic units and the routing fabrics also makes the SmartCell system suitable for other irregular data and signal processing.

3. SMARTCELL ARCHITECTURE

In a typical SmartCell architecture, a set of cell units is organized in a tiled structure. Each cell block consists of four processing elements (PEs) along with the control and data switching fabrics. A three-level layered interconnection network is designed for the intra- and inter-cell communications. A serial peripheral interface (SPI) is developed as an efficient way to load/reconfigure instruction codes into active cell units. By reconfiguring the instruction memories, the processing data flow can be changed to accommodate to different algorithms at run time. Fig. 1 depicts the high level organization of the integrated SmartCell architecture.

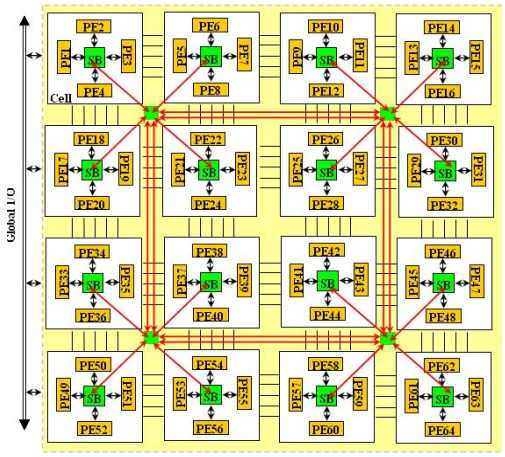

Fig. 1 Overview of the SmartCell architecture, composing of computational units and three-level hierarchical on-chip connection

3.1. Cell Design

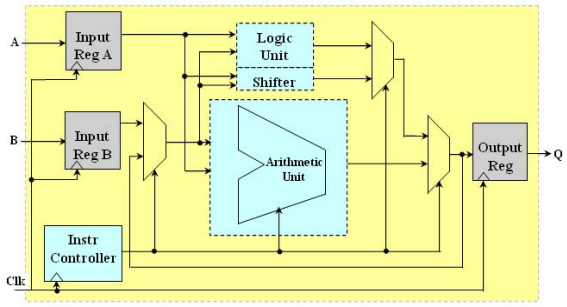

Fig. 2 Block diagram of the PE structure

Each cell consists of four PEs. Each PE is a customized ultra-low power 16-bit arithmetic logic unit (ALU) with input/output registers. The architecture of the homogeneous PE component is shown in Fig. 2. It can be configured to perform basic logic, shift and arithmetic functions. Multiple PEs can be chained together to implement more complex algorithms. An instruction controller is provided to control the datapath and functionality of each PE. The instruction code is preloaded into the instruction memory and can be dynamically changed. A desired operation is implemented by selecting relevant input/output signals and by setting corresponding operator controls. The SmartCell architecture does not perform the sequence of instruction fetching, decoding, register read/write and ALU operations as in conventional processors. Therefore, it is able to provide comparable power efficiency as an ASIC while maintaining dynamic programmability as a DSP.

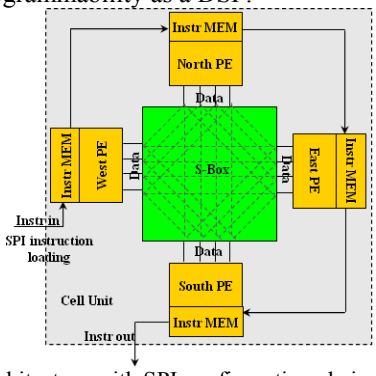

Fig. 3 Cell architecture with SPI configuration chain

For a cell unit, four PEs placed in the east, west, south and north directions form a quad structure with a fully connected cross-bar switch box (S_Box) located at the center, as shown in Fig. 3. The S_Box supports arbitrary data connections in the same cell unit. The data path controls are stored in the instruction memory attached to each PE and can be changed on a cycle by cycle basis. Each instruction code is designed in a 64-bit frame format, with a 16-bit program control section, a 31-bit datapath control section, a 10-bit operation control section and a 7-bit reserved section for future system extensions.

3.2. Reconfigurable Interconnection

As the CMOS technology scaling down, interconnection has become an increasingly important issue for integrated circuit design. A hierarchical routing structure with three-level networks is developed in our design. A fully connected switch box is developed in each cell to provide efficient intra-cell data communications. Since four PEs are placed at four edges in a cell, each PE can be directly linked to the nearest PE in adjacent cell, as shown in Fig. 1, to form a static nearest neighbor connection. Besides the local connections, a modified CMesh network is also designed to provide data exchanges among nonadjacent cells, illustrated in red in Fig. 1. The organization of the PEs involved in computing tasks can be reconfigured by the instruction code on a cycle by cycle basis.

3.3. Configuration Structure

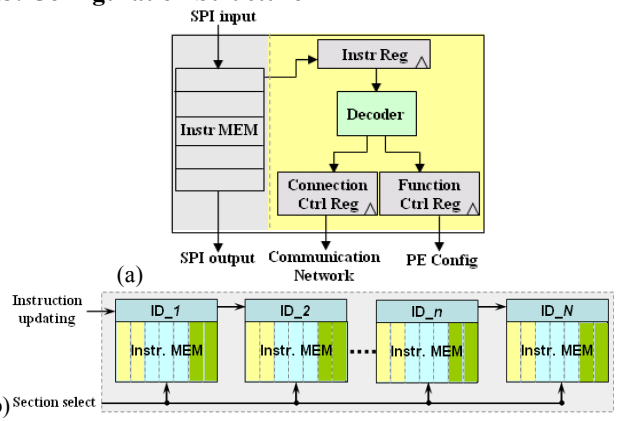

Fig. 4 SmartCell configuration structure (a) Run time controlling scheme; (b) Dynamic reconfiguration scheme.

In our design, the instruction memories are attached to PEs and are linked as a linear array through the SPI chain. During the initial configuration procedure, the instruction code is loaded into the first PE's instruction memory and is then shifted down to the second one and so on. The configuration procedure stops after the last active PE is configured. The instruction memory is partitioned into different sections. During dynamic reconfiguration, new instruction codes can be loaded into unused context sections through the same SPI chain, while the PEs maintain operating in the current contexts. Once the new contexts are fully loaded, a global select signal is used to indicate the change of operation context, as shown in Fig 4(b). This scheme can efficiently hide the configuration delay (up to 64 cycles) when swapping between different applications. Fig. 4(a) depicts the configuration controller architecture inside each PE. At run time, the configuration contexts are loaded into the instruction register one by one and are then decoded for the on-chip interconnection and functional controls. By this means, a cyclic execution of a specific dataflow is easily achieved.

4. RELATED WORK

Because the SmartCell falls into the category of coarse-grained reconfigurable architectures, it is important and meaningful to compare it with other CGRAs. A number of researches have been carried out to explore efficient CGRA designs, as summarized in [8]. In this section, we focus our comparison from the architecture viewpoint and clarify the differences between the SmartCell and some other CGRAs.

The RAW system [1] incorporates 16 simplified 32-bit MIPS processors in a 2D mesh structure to provide high parallel computing capacities. Both static and dynamic mesh networks are designed for data exchange among processors. However, the 6-stage ALU components exhibit a processor execution style of instruction fetching, decoding, and execution process that is similar to a typical multi-core processor. In addition, RAW does not provide dynamic reconfiguration. This makes it inflexible to adapt to any changes on the fly.

The RaPiD architecture [2] links hundreds of reconfigurable processing components in a 1D linear array, including ALUs, multipliers, RAMs, etc. Functionally specific hardware components can also be involved for application specific designs. Several parallel segmented buses are designed that can be dynamically changed for different data communications. Unlike the RAW processor, the RaPiD uses a combination of static and dynamic control logics. But the dynamic control is very expensive. The linear array nature limits the data parallelism in the temporal domain, and also makes it not very appropriate for 2D block based applications.

In the PipeRench system [3], several reconfigurable pipeline stripes are offered as an accelerator for data streaming applications. Each stripe consists of multiple computing units organized in a 1D array structure. Limited configurable interconnection fabrics are developed, including a local network inside a stripe, unidirectional nearest neighbor connection between stripes and some global buses. Hardware sharing is provided to map large applications onto limited computing resources. However, similar to the RaPiD system, the linear array structure makes PipeRench inefficient for 2D block applications.

Our SmartCell system integrates some of the prominent features in the previous systems. The 16-bit granularity for basic operations is efficient for data parallelism exploration, while keeping a low computing and communicational cost. The SmartCell can be configured to operate in SIMD, MIMD and systolic array styles due to the distributed contexts and rich on-chip connections. Dynamic adjustments for new system requirements are achieved by reconfiguring the context memories through the SPI structure. In combination of these features, we say that the SmartCell is a unique approach in the CGRA family.

5. SYSTEM IMPLEMENTATION AND PERFORMANCE

In this section, a 4 by 4 prototype SmartCell system with 64 PEs is designed and synthesized. Seven benchmark applications listed in Table 1 have been manually mapped onto the SmartCell, based on which the power consumption and system throughput are analyzed. These benchmarks represent a wide range of computationally intensive kernels involved in real-time applications. The same benchmarks are adopted to evaluate the power and throughput results with other computing platforms, including FPGA, RaPiD and ASIC. The Altera's Stratix II EP2S30 device [9] is selected as the FPGA benchmark platform. The main reason to choose the RaPiD CGRA system in comparison is that it shares a similar set of test benches with the SmartCell and the details on performance results are available.

Table 1. Application Domain and Evaluated Benchmarks

Application domain	Test Benches
Signal processing	64-tap FIR, 64-tap IIR
Multimedia and image processing	32-point FFT, 8*8 2D-DCT, 8 by 8 Motion Estimation (ME) in 24 by 24 searching area
Scientific computing	128 by 128 Matrix Multiplication(MMM) 64^{th}-order Polynomial Evaluation (PoE)

In our prototype SmartCell system design, a functional RTL model is firstly developed in hardware description language (HDL) and is then synthesized in Synopsys Design Compiler to generate the CMOS standard cell ASICs using TSMC .13μm technology. The area and timing reports are also generated by Design Compiler. The synthesized design is then annotated via a set of benchmarks for power consumption estimation in Synopsys Power Compiler. Some experimental setups are listed in Table 2.

Table 2. System Design and Simulation Parameters

System dimension	4 by 4
Design tools	ModelSim, Synopsys EDAs
Library	TSMC .13μm process
Voltage	1 V
Simulation frequency	100 MHz

5.1. SmartCell Synthesis Results

The chip area and power consumption of the prototype SmartCell is shown in Fig. 5. It occupies about 8.2 mm^2 that is about 1.6 million gate equivalents. The synthesis results indicate that the processing units contribute to about 45% of the total area, while the instruction memory counts for about 41% due to long instruction format and intensive controlling requirements. The hierarchical on-chip connections roughly occupy 14% of the total area. The SmartCell system can achieve a maximum frequency of about 123 MHz.

Table 7. Comparisons of Power Consumption and System Throughput @ 100 MHz

		FIR	IIR	MMM	2D DCT	ME	FFT	RC5	PoE
SmartCell	Power <mW>	152	189	143	161	145	169	140	150
	Throu.*	100 MS/s	100 MS/s	763 Matrices/s	1.56 Mblocks/s	865 Kblocks/s	58 MS/s	50 Mblocks/s	100 MS/s
RaPiD [10]	Power <mW>	203	-	428	439	235	-	-	-
	Throu.	100 MS/s	-	763 Matrices/s	1.56 Mblocks/s	865 Kblocks/s	-	-	-
FPGA	Power <mW>	725	896	445	787	573	431	553	628
	Throu.	100 MS/s	100 MS/s	763 Matrices/s	1.56 Mblocks/s	865 Kblocks/s	100 MS/s	50 Mblocks/s	100 MS/s
ASIC	Power <mW>	31	45	9	55	12	38	9	55
	Throu.	100 MS/s	100 MS/s	763 Matrices/s	1.56 Mblocks/s	865 Kblocks/s	58 MS/s	50 Mblocks/s	100 MS/s

* MS/s = Mega samples/second. The FIR results of the RaPiD implementation are based on a 16-tap FIR filter.

Full chip configuration can be completed within 12 microseconds, which is hundreds of times faster than FPGA configurations. On average, the SmartCell system consumes about 158 mW on the evaluated benchmarks. The processing units consume 41% of the total power and about 53% is consumed by the on-chip interconnection and instruction memories. An average energy efficiency of 28.5 GOPS/W (giga operations per second per watt) is achieved.

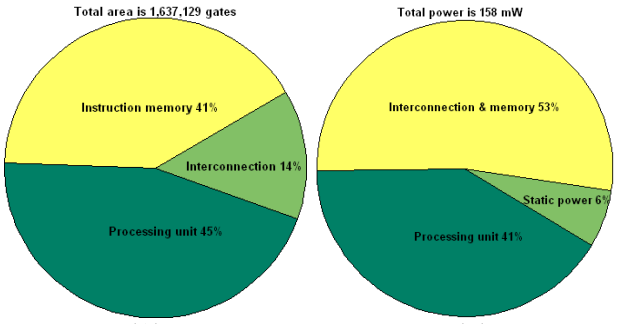

Fig. 5 Area and average power consumption of the 4 by 4 SmartCell system: (a) Area breakdown (b) Average power consumption breakdown @ 100 MHz.

5.2. Comparison with Other Computing Architectures

It is meaningful to compare the SmartCell performance with other computing platforms. Table 7 compares the power consumption and system throughput performance among the SmartCell, RaPiD, FPGA and ASIC. SmartCell and RaPiD can achieve the same system throughput due to the similar mapping structure and the same operating frequency. But the SmartCell is more flexible than RaPiD for applications involving feedback loops and intensive data communications, such as IIR and FFT. FPGA can achieve a higher throughput in the FFT test bench than the SmartCell system, due to its dedicated pipeline structure and the assistance of high volume on-chip memories. The same throughputs are observed in the SmartCell and FPGA platforms for other benchmarks.

A power comparison of the evaluated reconfigurable platforms is given in Fig. 6, which has been normalized to the power consumption of the SmartCell system. Due to different process technologies and operating voltages, power scaling is applied to the power consumption results of the RaPid platform for fair comparison. On average, the SmartCell consumes about 52% less power than the RaPiD system and about 75% less power than the FPGA. More importantly, due to the coarse granularity feature, the configuration process of the SmartCell system is much faster than the FPGA implementations. The dynamic configuration is also more efficient than that in the FPGAs.

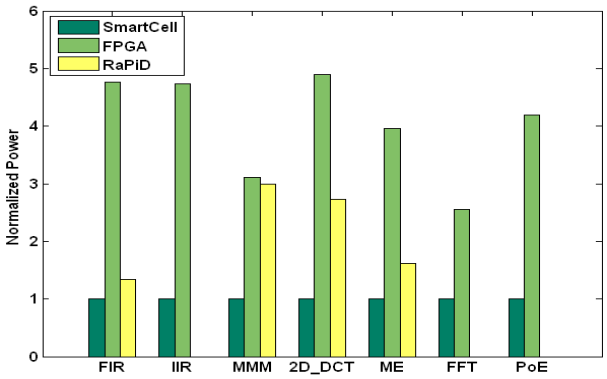

Fig. 6 Comparison of power consumption results, normalized to the SmartCell results

The ASIC implementations are also developed to provide the best system performance in comparison with other reconfigurable computing platforms. Fig. 7 compares the average energy efficiency (GOPS/W), normalized to the FPGA result. As expected, the ASIC implementation performs better than the reconfigurable systems, achieving a factor of 19.6 energy efficiency gain with respect to the FPGA result. However, this performance is achieved at a cost of non-post fabrication flexibility and high engineering design efforts. The SmartCell system achieves a factor of 4.1 better than the FPGA implementation and a factor of 4.8 less than the ASIC implementation. The results demonstrate that the SmartCell architecture is able to merge the power gap between the fine-grained FPGA and logic specific ASIC architectures.

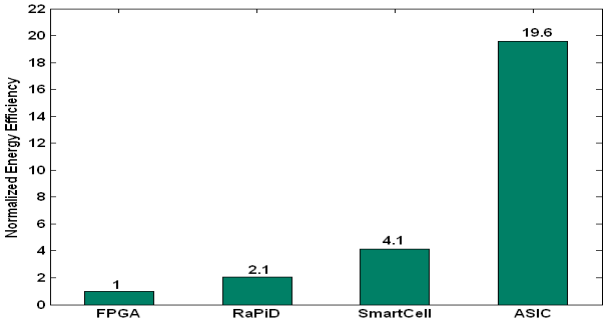

Fig. 7 Comparative energy efficiency results of different computing platforms, normalized to the FPGA result

6. CONCLUSIONS AND FUTURE WORK

This paper represents the SmartCell system as a low power and reconfigurable architecture for data streaming applications. By tiling a large number of computing units with reconfigurable communication fabrics, the SmartCell is able to bridge the gap of performance and flexibility between ASICs and general purpose processors. Its coarse granularity nature also avoids the high computing and communication overheads relative to the FPGA designs. A set of benchmark kernels has been successfully mapped and implemented onto the SmartCell system. A 4 by 4 SmartCell system is developed and evaluated in standard cell ASICs. This prototype system consists of about 1.6 million gates with an average power consumption of 158 mW. It is about 2.0 and 4.1 times more power efficient than the RaPiD and FPGA designs, respectively. The evaluation results demonstrate that the SmartCell system is able to bridge the energy efficiency gap between reconfigurable FGPAs and fixed ASIC architectures.

As part of our future efforts, the dynamic reconfiguration performance will be evaluated in terms of efficiency and timing. Also, by adding the memory connection and management supports, some large real-time applications will be developed and implemented on the SmartCell system to explore its computing capacity in real applications.

Another important aspect of our research is to develop a programming environment to automatically map applications onto the SmartCell system. Due to the controlling intensive nature, a prototype compiler is proposed to facilitate the context generation for specific application domains. A model-based algorithm description language, namely Smart-C, will be developed with annotation from standard C. In addition, a generic application mapping flow will be developed using the Smart-C environment. Firstly, profiling is performed over the input application to extract the work loads. Data dependency is also analyzed in this step. Then, the parallelism exploration and application partitioning are performed based on hardware description and system requirements. After that, the control signals are specified for each computing and communication element to form desired application datapath. Two modes are provided for both offline and online configurations. Development of the Smart-C compiler will be the next focus of our future work.

7. REFERENCES

[1] M.B. Taylor, and etc., "The Raw microprocessor: a computational fabric for software circuits and general-purpose programs," *IEEE Micro*, vol. 22, pp. 25-35, Mar./Apr. 2002.

[2] C. Fisher, and etc. "Emulator for Exploring RaPiD Configurable Computing Architectures," *In Proceedings of International Conference on Field-Programmable Logic and Applications*, Aug. 2001.

[3] H. Schmit, and etc., "PipeRench: A virtualized programmable datapath in 0.18 micron technology," *In Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 63 - 66, May 2002.

[4] http://www.xilinx.com

[5] M. Schlansker and B. Rau, "EPIC: Explicily Parallel Instruction Computing," *IEEE Computer*, vol. Vol. 33 no. 2, pp. 37 - 45, 2000.

[6] A. Peleg, S. Wilkie, and U. Weiser, "Intel MMX for Multimedia PCs," *Communications of the ACM*, 1997.

[7] M. Bayoumi and N. Ling, "Specification and Verification of Systolic Arrays," *World Scientific Publishing Singapore*, 1999.

[8] R. Hartenstein, "A decade of reconfigurable computing: a visionary retrospective," *In Proceedings of IEEE Conference and Exhibition on Design, Automation and Test in Europe*, Mar. 2001.

[9] http://www.altera.com/products/devices/stratix2/st2-index.jsp

[10] D.C. Cronquist, M. Figueroa, P. Franklin, and C. Ebeling, "Architecture design of reconfigurable pipelined datapaths," *In Proceedings of the 20th Anniversary Conference on Advanced Research in VLSI*, Mar. 1999.