Comparison between the Simulator and Scheduler based approach of Design Space Exploration for Application Specific Instruction set Processor

M. K. Jain
Associate Professor
Dept. of CS
Mohan Lal Sukhadia University, India

Deepak Gour
Assistant Professor
Dept. of CSE
Sir Padampat Singhania University, India

ABSTRACT
An Application Specific Instruction set Processor (ASIP) is a processor designed for a particular application or for a set of applications. An ASIP exploits special characteristics of application(s) to meet the desired performance, cost and power requirements. ASIP Design Space Exploration can be carried out by one of the two popular available techniques as Simulator or Scheduler based approach. This paper is an attempt to survey and compare both the mentioned techniques of design space exploration of ASIP. A list of explored design space parameters using simulator based approach is included in this paper. This paper also highlights few potential parameters which can be explored using the scheduler based approach.

General Terms
Application Specific Instruction set Processor (ASIP), Design Space Exploration (DSE)

Keywords
Simulator based approach of Design Space Exploration, Scheduler based approach of Design Space Exploration

1. INTRODUCTION
An Application Specific Instruction set Processor (ASIP) is a processor designed for a particular application or for a set of applications. An ASIP exploits special characteristics of application(s) to meet the desired performance, cost and power requirements. According to Liem et al [1], ASIPs are a balance between two extremes: ASICs (Application Specific Integrated Circuit) and GPP (General Programmable Processors). Since an ASIC is specially designed for one behavior, it is difficult to make any changes at a later stage. In such a situation, the ASIPs offer the required flexibility at lower cost than GPP.

ASIP can be easily used in many embedded systems such as automotive control, household appliances, cellular phones, avionics etc. GPP are designed for general use. Many times it happens that specific applications need a certain mix which does not match the GPP resource mix. If we plan to design an ASIC to meet the given performance, power and area constraints for the given application, deign becomes rigid. In the ASIP design, it is important to search for a processor architecture that matches target application. To achieve this goal, it is essential to estimate design quality of various candidate architecture in terms of area, performance, and power consumption. Table 1 shows the comparison among GPP, ASIP and ASIC.

Table 1. Comparison among GPP, ASIP and ASIC

	GPP	ASIP	ASIC
Performance	Low	High	Very High
Flexibility	Excellent	Good	Poor
HW design effort	Nil	Large	Very Large
SW design effort	Small	Large	Nil
Power	Large	Medium	Small
Reuse	Excellent	Good	Poor
Markets	Very large	Relatively large	Small

2. STEPS IN ASIP SYNTHESIS
Gloria et al [2] defined some main requirements of the design of application-specific architectures. Important among these are as follows:

- Design starts with the application behavior.
- Evaluate several architectural options.
- Identify hardware functionalities to speed up the application.
- Introduce hardware resources for frequently used operations only if it can be supported during compilation.

ASIP fits in between these two and provides flexibility at lower cost than general programmable processors. According to MK Jain et al [3, 4, 5, 6, 7] design of ASIP can be typically divided in five steps as mentioned below:

- Application Analysis
- Architecture design space Exploration.
- Instruction-set generation
- Code synthesis
- Hardware synthesis

3. ARCHITECTURE DESIGN SPACE EXPLORATION

It involves in identifying the broad architectural features of the ASIP. Figure 1 explains the block diagram of architecture explorer. First of all, the architectural space to be explored is defined, keeping in view the parameters extracted during application analysis and the input constraints. Architecture is

defined using some standard Architecture Definition Language (ADL) as EXPRESSION [8] and LISA [9, 10, 11].

Performance estimation which drives the design space exploration is either **simulation based** (e.g. Gloria et al [2], Kienhuis et al [12], Imai et al [13], Binh et al [14]) or **scheduler based** (e.g. Gupta et al [15]).

The architectural design space is to be explored usually defined in terms of a parameterized architectural model.

The main focus points are as follows:

- The parameterized architectural model suggested by all the researchers includes the number of functional units of different types.
- Architectures considered by different researchers also differing in terms of the instruction level parallelism they support.
- Most of these approaches consider only flat memory.

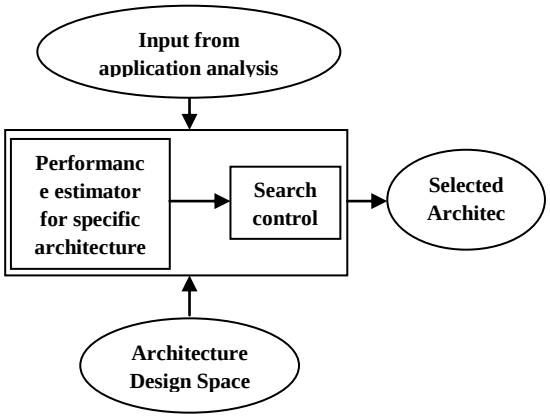

Fig 1: Block Diagram of an Architecture Explorer

4. PERFORMANCE ESTIMATOR TECHNIQUES

4.1 Simulator based approach

The most popular approach for ASIP design space exploration is simulator based approach. In the simulator based approach, a simulation model of architecture based on the selected features is generated and the application is simulated on this model to compute the performance. Figure 2 explains the functioning of simulator based approach.

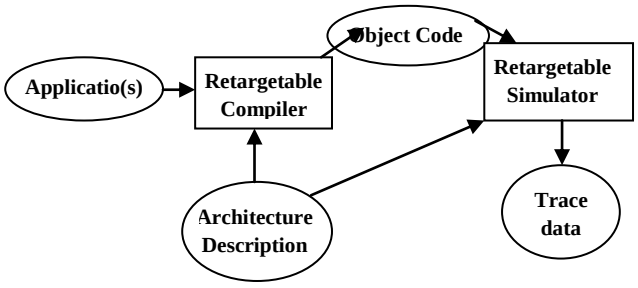

Fig 2: Architecture exploring using simulator based approach

4.2 Scheduler based approach

In scheduler based approach the problem is formulated as a resource constrained scheduling problem with the selected architecture components as the resources and the application is scheduled to generate an estimate of the cycle count. Profile data is used to obtain the frequency of each operation. Figure 3 explains the functioning of simulator based approach.

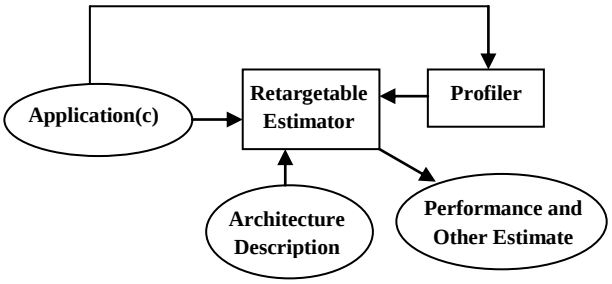

Fig.3: Architecture exploring using scheduler based approach

5. RELATED WORK AND PARAMETERS EXPLORED IN DESIGN SPACE EXPLORATION USING SIMULATOR BASED APPROACH

In the recent past the major work carried out in Design Space Exploration is by using Simulator based approach. The major contributions are as follows:

Swarnalatha Radhakrishnan et al [16] explores the DSE on heterogeneous multiple pipelines. She proposed Application Specific Instruction Set Processors with heterogeneous multiple pipelines to efficiently exploit the available parallelism at instruction level. They have developed a design system based on the Thumb processor architecture. Given an application specified in C language, the design system can generate a processor with a number of pipelines specifically suitable to the application, and the parallel code associated with the processor. Each pipeline in such a processor is customized, and implements its own special instruction set so that the instructions can be executed in parallel with low hardware overhead.

Ascia, Vincenz Catania, Palesi et al [17] explores the DSE using genetic algorithms on parameterized SOC platforms. The basic idea is to avoid designing a chip from scratch. They proposed an approach based on genetic algorithms for exploring the design space of parameterized system-on-a-chip (SOC) platforms. The strategy focuses on exploration of the architectural parameters of the processor, memory subsystem and bus, making up the hardware kernel of a parameterized SOC platform for the design of embedded systems with strict power consumption and performance constraints. The approach has been validated on two different parameterized architectures: one based on a RISC processor and another based on a parameterized very long instruction word architecture.

Kwon, Lee, Kim, Ha et al [18] explores cache misses and memory arschitecture issues using Y-Chart approach to DSE. Y chart consists of two loops as 1) Co-synthesis loop for component selection and mapping of the function blocks to

the processing components and 2) Communication DSE loop for communication architecture optimization.

Lilian Gogniat, Phillipe et al [19] explores DSE using special tool called Design Trotter. This tool allow for the exploration of their design space to choose the best architecture characteristics. They proposed an original approach based on a high-level representation of the application and on a hierarchical functional model for the architecture. This approach targets fine-grain, coarse-grain, and heterogeneous architectures.

Kyeong, Mooney et al [20] explore the DSE on issues related to Bus Architecture where they propose Bus Synthesis tool to generate the five different bus systems. This paper presents a methodology to generate a custom bus system for a multiprocessor System-on-a-Chip (SoC). Our bus synthesis tool (BusSyn) uses this methodology to generate five different bus systems as examples: Bi-FIFO Bus Architecture (BFBA), Global Bus Architecture Version I (GBAVI), Global Bus Architecture Version III (GBAVIII), Hybrid bus architecture (Hybrid) and Split Bus Architecture (SplitBA). They verified and evaluate the performance of each bus system in the context of two applications: an Orthogonal Frequency Division Multiplexing (OFDM) wireless transmitter and an MPEG2 decoder. This methodology gives the designer a great benefit in fast design space exploration of bus architectures across a variety of performance impacting factors such as bus types, processor types and software programming style.

Kim, Keimh, Choi et al [21] explores the DSE on the issues of Area, Critical path delays. The optimization is based on pipelining and sharing of functional resources in the PE of the array. They proposed design space exploration flow with two optimization techniques. The optimization is based on pipelining and sharing of functional resources in the processing elements of the array. For fast architecture exploration, optimization techniques are applied to SystemC model. They estimated entire performance at early stage by transaction level simulation and this feature enables early detection of optimal architecture specification.

Kunzil, Thiele et al [22] explores the DSE on the issues like # of cache lines, block size and replacement strategy. A generic approach is described based on multi-objective decision making, black-box optimization and randomized search strategies. The interface between problem-specific and generic parts of the exploration framework is made explicit by defining an interface called PISA. This specification and implementation interface, and the availability of a wide range of randomized multi-objective search methods, makes the proposed framework accessible to a wide range of exploration problems.

Ascia, Catania et al [23] explores the DSE on the issues related on Register File size (GPR, FPR, PR, CR, BTR) and L1 and L2 caches. They presented EPIC-Explorer, a framework for the simulation of a parameterized SOC platform based on a VLIW processor. The main use the platform has been designed for is to provide a powerful, flexible simulation and estimation framework that can be used to develop design space exploration algorithms.

Pasricha, Dutta et al [24] explores the DSE on the issues related to the Bus architecture. They proposed an automated application specific co-synthesis framework for memory and communication architectures (COSMECA) in MPSoC designs. The primary objective is to design a communication architecture having the least number of busses, which satisfies performance and memory area constraints, while the secondary objective is to reduce the memory area cost.

Gour Deepak, Jain M K et al [25] derived the parameters of design space exploration using the simulator based approach. Table 2 presents the same explored parameters.

Table 2. Parameters of Design Space Exploration using Simulator based approach

Sr. No.	Explored Design Space Exploration Parameters using Simulator based approach
1	Instruction cache size [Kunzil, Thiele et al [22]]
2	Data cache size [Kunzil, Thiele et al [22]]
3	Processor to address bus encoding [Pasricha, Dutta al [24]]
4	Processor to data bus width [Pasricha, Dutta et al [24]]
5	Processor to data bus encoding [Pasricha, Dutta et al [24]]
6	Processor to address bus width [Pasricha, Dutta et al [24]]
7	Cache to memory address bus width [Pasricha, Dutta et al [24]]
8	Cache to memory address bus encoding [Pasricha, Dutta et al [24]]
9	Cache to memory data bus width [Pasricha, Dutta et al [24]]
10	Cache to memory data bus encoding [Pasricha, Dutta et al [24]]
11	GPR (General Purpose Register) File size [Ascia, Catania et al [23]]
12	FPR (Floating Point Register) File size [Ascia, Catania et al [23]]
13	PR (Predicate Register) File size [Ascia, Catania et al [23]]
14	CR (Control Register) File size [Ascia, Catania et al [23]]

15	BR (Branch Register) File size [Ascia, Catania et al [23]]
16	# of IU (Integer Units) [Kim, Keimh, Choi et al [21]]
17	# of FPU (Floating Point Units) [Kim, Keimh, Choi et al [21]]
18	# of MU (Memory Units) [Kim, Keimh, Choi et al [22]]
19	# of cache lines [Kunzil, Thiele et al [22]]
20	Block size [Kunzil, Thiele et al [22]]
21	Associativity [Kunzil, Thiele et al [22]]
22	Replacement strategy (LRU / FIFO) [Kunzil, Thiele et al [22]]
23	Bus speed [Kunzil, Thiele et al [22]]
24	Arbitration Speed [Kunzil, Thiele et al [18]]
25	OO Buffer size [Kwon, Lee, Kim, Ha et al [18]]
26	Memory Mapping [Kwon, Lee, Kim, Ha et al [18]]
27	Pipelined function [Swarnalatha Radhakrishnan et al [16]]
28	Latency of functional units [Kim, Keimh, Choi et al [21]]
29	Number of operational slots [Kim, Keimh, Choi et al [21]]

6. SCHEDULER BASED APPROACH FOR DESIGN SPACE EXPLORATION

The second approach for exploration of DSE is Scheduler based approach. Where most of the researcher worked on the simulator based approach but few worked on scheduler based approach as Gupta, Balakrishnan et al [15] and MK Jain, Kumar Anshul, Balakrishnan et al [3,4,5,6,7] are the main contributors of the scheduler based approach.

The major disadvantage in the simulator based approach is to map the large / growing design space. As the parameter of the design space to be changed, every time one needs to construct the retargetable simulator for an architectural template to accommodate the newly introduced design space parameter. For each architecture instance, a specific simulator is derived in three steps. The architecture instance is constructed, an execution model is added and the executable architecture implemented with metric collectors to obtain the performance numbers.

Whereas in the Scheduler based approach one need to develop a retargetable estimator for performance estimation required for architectural exploration using a better architecture model. The optimization considered include: optimized multi operation pattern matching, adding mode optimization, loop optimization, loop unrolling etc.

As the Design Space is growing exponentially, it is virtually very difficult to develop a simulator which meets the growing size of the Design Space. Here the proposed scheduler based approach comes into the picture which is an ideal solution in the current scenario.

The main parameters explored in the scheduler based approach are on the combinations of Functional Units and partially of Memory architectures. For e.g. MK Jain et al [3, 4, 5, 6, 7] explores the Design Space Exploration (DSE) on register file size, register window and on-chip data cache. There are very less work carried out on the parameters like Storage Exploration, Pipeline Structure, and Bus Architecture which was extensively explored by simulator based approach.

MK Jain, Anshul Kumar, Balakrishnan et al [3, 4, 5, 6, 7] explores the register file size but took only the simple plain register file size i.e. GPR (General Purpose Register) and didn't consider the related parameters as FPR, BR, CR and PR in his study of DSE. Apart from that a very limited study carried out in the estimation of the execution time where the effect of instruction word length on execution is ignored. Also there is no consideration on the effect of pipeline structures in the study as well. Table 3 presents the potential parameters which one can explore using the scheduler based approach.

Table 3. Potential parameters which can be explored using the scheduler based approach

Sr. No.	**Potential parameters can be explored using the scheduler based approach**
1	**General Purpose Register (GPR)**
1.1	GPR Heterogeneous register files
1.2	GPR Multiple register files
2	**Impact of special purpose register**
2.1	FPR (Floating Point Register) File size
2.2	PR (Predicate Register) File size
2.3	CR (Control Register) File size
2.4	BR (Branch Register) File size
3	**Effect of instruction word length on execution**
4	**Multi level memory architecture**

7. CONCLUSION

This paper is an attempt to throws the light on the major disadvantage in the simulator based approach. The simulator based approach is not an ideal approach when one needs to

map the large / growing design space. As the parameter of the design space to be changed, every time one needs to construct the retargetable simulator for an architectural template to accommodate the newly introduced design space parameter.

In this scenario, the scheduler based approach is one of the best available approaches to map the growing / large design space. This approach gives the facility to tap the large and fast growing design space with considerable lesser amount of time and complexity. Table 3 shows the potential parameters which can be further explored using the scheduler based approach.

8. REFERENCES

[1] Liem, C.; May, T.; Paulin, P., "Instruction-set matching and selection for DSP and ASIP code generation.", In Proc. EURODAC-94, 28 Feb.-3 March 1994, pp. 31-37.

[2] Gloria A. D.; Faraboschi, P., "An evaluation system for application specific architectures.", In Proc. Micro-23, 27-29 Nov. 1990, pp. 80-89.

[3] Jain M.K.,Balakrishnan M., and Kumar A., "ASIP Design Methodologies: Survey and Issues", In Proceedings of the IEEE / ACM International Conference on VLSI Design. (VLSI 2001), pages 76–81, January 2001.

[4] Jain M. K.,Wehmeyer L., Steinke S., Marwedel P., and Balakrishnan M., "Evaluating Register File Size in ASIP Design", In Proceedings of the Ninth International Symposium on Hardware/ Software Co-design,(CODES 2001), pages 109–114, April 2001.

[5] Jain M. K., Balakrishnan M. and Kumar A., "An Efficient Technique for Exploring Register File Size in ASIP Design", In Proceedings of the Fifthth International Conference on Compilers, Architecture and Synthesis for Embedded Systems, (CASES 2002).

[6] Jain M. K., Wehmeyer L., Marwedel P., Balakrishnan M., "Register File Synthesis in ASIP Design", Technical Report #746, 07.12.2000, Lehrstuhl Informatik XII, University of Dortmund, Germany.

[7] Jain M. K., Balakrishnan M. and Kumar A., "Exploring Storage Organization in ASIP Synthesis", In Digital System Design, 2003. Proceedings. Euromicro Symposium on
Volume , Issue , 1-6 Sept. 2003 Page(s): 120 – 127.

[8] Halambi A., Grun P., Khare A., Ganesh V., Dutt N., Nicolau A., EXPRESSION: A Language for Architecture Exploration through Compiler/Simulator Retargetability, In Proceedings of the Design Automation and Test in Europe (DATE), pages 485–490, March 1999.

[9] Pees S., Zivojnovic V., Mey H., LISA- Machine Description Language for Cycle Accurate Models of Programmable DSP Architectures, In Proceedings of the Design Automation Conference (DAC), pages 933–938, June 1999.

[10] Hoffmann A., Kogel T., Nohl A., Braun G., Schliebusch O., Wahlen O., Wieferink A., Meyr H., A Novel Methodology for the Design of Application-Specific Instruction-Set Processors (ASIPs) Using a Machine Description Language, In IEEE Transactions on Computer Added Design of Integrated Circuits and Systems, 20(11) pages 1338–1354, November 2001.

[11] Schliebusch O., Hoffmann O., Nohl A., Braun G., Meyr H., Architecture Implementation Using the Machine Description Language LISA, In Proceedings of the IEEE / ACM International Conference on VLSI Design and ASP Design Automation Conference. (VLSI/ ASPDAC 2002), pages 239–244, January 2002.

[12] Kienhuis B., Deprettere E., Vissers K., The Construction of a Retargetable simulator for an architecture template In Hardware/Software Codesign, 1998. (CODES/CASHE apos;98) Proceedings of the Sixth International Workshop on Volume, Issue, 15-18 pages 125 – 129, March 1998.

[13] Sato J., Imai M., Hakata T., Alomary A. Y. ,Hikichi N., An integrated design environment for application specific integrated processor, In Proc. ICCD-91, pages 414-417, October 1991.

[14] Binh N. N., Imai M., Shiomi A., A new HW/SW partitioning algorithm for synthesizing the highest performance pipelined ASIPs with multiple identical FUs, In Proc. DAC-96, pages 126-131, September 1996.

[15] Gupta T. V. K., Sharma P., Balakrishnan M., Malik S., Processor evaluation in an embedded systems design environment, In Proc. VLSI Design 2000, pages 98-103, January 2000.

[16] Radhakrishnan Swarnalatha: "Customization of application specific heterogeneous multi pipeline processors", In Proc. EDAA 2006, pp. 746 – 751.

[17] Giuseppe Ascia, Vincenzo Catania, and Maurizio Palesi, "A GA-Based Design Space Exploration Framework for Parameterized System-On-A-Chip Platforms", In IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 8, NO. 4, AUGUST 2004, pp. 329 – 346.

[18] Seongnam Kwon, Choonseung Lee, Sungchan Kim, Youngmin Yi, Soonhoi Ha, "Fast Design Space Exploration Framework with an Efficient Performance Estimation Technique", In Embedded Systems for Real-Time Multimedia, 2004. ESTImedia 2004. 2nd Workshop on
Volume , Issue , 6-7 Sept. 2004 Page(s): 27 – 32.

[19] Lilian Bossuet, Guy Gogniat, and Jean-Luc Philippe, "Communication-Oriented Design Space Exploration for Reconfigurable Architectures", In EURASIP Journal on Embedded Systems, Volume 2007, Article ID 23496, 20 pages.

[20] Kyeong Keol Ryu and Vincent J. Mooney III, "Automated Bus Design Space Exploration for Multiprocessor SoC", In Design, Automation and Test in Europe Conference and Exhibition, 2003 Volume , Issue , 2003 Page(s): 282 – 287.

[21] Yoonjin Kim, Mary Kiemb, Kiyoung Choi, "Efficient Design Space Exploration for Domain-Specific Optimization of Coarse-Grained Reconfigurable Architecture", In Design, Automation and Test in Europe, 2005. Proceedings Volume , Issue , 7-11 March 2005 Page(s): 12 - 17 Vol. 1.

[22] Kunzli S., Thiele L. and Zitzler E., "Modular design space exploration framework for embedded systems", In Computers and Digital Techniques, IEE Proceedings - Volume 152, Issue 2, Mar 2005 Page(s): 183 – 192.

[23] Giuseppe Ascia, Vincenzo Catania, Maurizio Palesi and David Patti "EPIC Explorer: A parameterized VLIW based Platform Framework for Design Space Exploration", In First workshop on Embedded Systems for Real time Multimedia (ESTIMedia), Newport Beach, California, USA, Oct. 3-4, 2003.

[24] Pasricha S. and Dutt N., "A Framework for Memory and Communication Architecture Co-synthesis in MPSoCs", In Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on Volume 26, Issue 3, March 2007 Page(s):408 – 420.

[25] Gour Deepak, Jain M. K., "ASIP Design Space Exploration: Survey and Issues", International Journal of Computer Science and Information Security, Volume 9, No. 3, 2011 Page(s): 141 – 145.