Energy-Efficient Application-Specific Processors

Mahendra Pratap Singh[1]
[1]Assistant Professor, Navrachana University, Vadodara, India
mahendraps@nuv.ac.in

Abstract : With the mobile computing revolution, high performance computing is redefined by energy efficiency of processors. Energy efficiency is an essential consideration while selecting a processor to be used for an application. This is more critical in mobile and embedded applications which require energy efficient operation. Application specific processors have emerged as a solution for providing low cost, energy-efficient, and high performance architectural implementations. ASIPs (Application-Specific Instruction Set Processors) improve the energy efficiency by using a hardware accelerator which is tightly coupled with the base processor. Various techniques have been proposed to improve energy efficiency of ASIPs by customizing the instruction set. This paper is an attempt to study energy-efficiency in ASIPs.

Keywords – **Energy-Efficiency, ASIP, Customization**

I. INTRODUCTION

Consumer electronics landscape has paved the way for embedded systems with ever growing functionalities. Embedded system architectures should satisfy constraints like improved flexibility, shorter time-to-market, low power designs, etc. In embedded applications, power consumption can be reduced by minimizing number of cycles through processor architecture customization and also by moving critical computations from run time to compile time. For a hardware implementation, flexibility and energy-efficiency are the major design goals. The energy-flexibility gap for various architectural implementations like DSPs (Digital Signal Processors), FPGAs (Field Programmable Gate Arrays), embedded processors and ASIPs is depicted in figure-1. For a given application, ASIPs can fill the energy-flexibility gap between programmable DSPs and dedicated hardware. ASIPs have better energy-efficiency due to custom instructions and optimized data paths.
ASIPs are the processors designed for a particular application or a set of applications. ASIP exploits special characteristics of target applications to meet performance, cost, and energy requirements. Instruction set customization process in ASIPs enables specific instructions to be added to the instruction set thereby reducing power consumption and execution time. Applications like cellular phones, digital signal processors, and automatic control systems can benefit a lot from ASIP architectural designs.

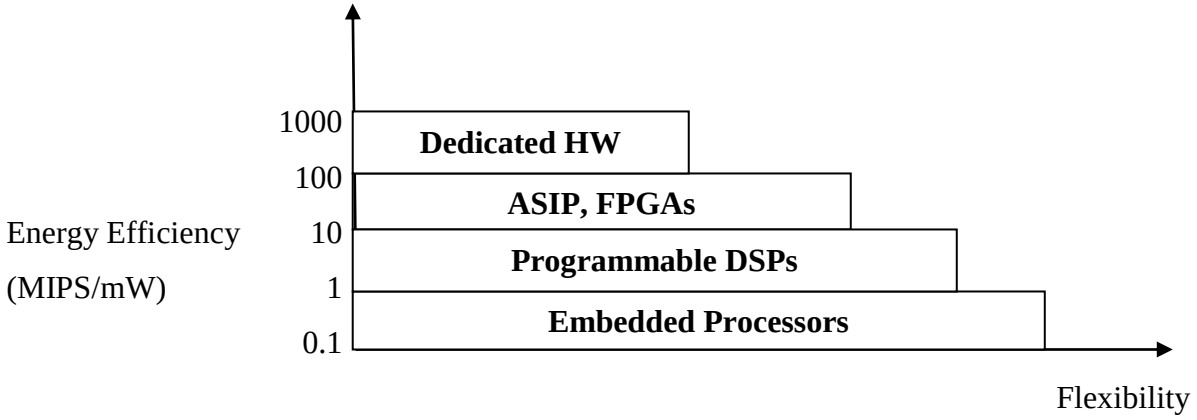

Figure-1: The Energy-Flexibility Gap

Source: Zhang et al., ISSCC 2000

II. Related Work

Lee et al. (2003) presented an approach for energy-efficient instruction set synthesis of ASIPs through optimal instruction encoding. As far as energy-efficiency is concerned, Complex Instruction Set Computer (CISC) instruction sets are more energy-efficient than reduced instruction set computer (RISC) instruction sets. With the advent of reconfigurable architectures, instruction set customization has emerged as a way for improving energy-efficiency of processors. Customization may be done regarding number of instructions, instruction bandwidth, instruction encoding, etc. in the selected processor architecture.

For synthesizing instruction sets, an application with basic instruction sets is provided as input. A retargetable compiler is used to compile the application resulting in assembly code. Complex instruction generation is the next step where a group of complex instructions are generated. In the next complex instruction selection step, most profitable complex instructions are selected to be included in the final instruction set.

Energy consumption in processors can be achieved by optimized hardware architectures along with softwares which efficiently uses the hardware. Custom instructions performing multiple operations can be used for improving energy efficiency in processors. Using scratch pad memories for temporary storage can also be used for low power operation. (Glokler and Meyr, 2004).

Goodman and Chandrakasan (2001) implemented an energy-efficient reconfigurable public-key cryptography processor. Cryptographic algorithms are classified into symmetric or secret-key algorithms and asymmetric or public key algorithms. Symmetric key algorithms have a shared secret key for encryption/ decryption of bulk data which makes its implementation compute intensive. Public-key algorithms are used for authenticating a user and utilize a set of public and private keys for this task. Public-key algorithms have complex mathematical calculations like long modular multiplication which makes its implementation a difficult task. These algorithms can be implemented by using techniques like elliptic curves (EC), integer factorization (IF), and discrete logarithms (DL). Implementing these algorithms in software leads to energy inefficient designs with slow implementations, while hardware implementations are computationally and energy efficient but are very inflexible.

Reconfigurable cryptographic processors are a mix of these two types of implementations for energy-efficient cryptographic implementation. Instruction Set Architecture (ISA) of these processors consists of various instructions grouped into six categories namely conventional arithmetic, modular integer arithmetic, GF arithmetic, elliptic curve field arithmetic over GF, register manipulation, and processor configuration. The proposed cryptographic processor provides improved performance and energy efficiency as compared to software or hardware implementations.

Lin and Fei (2012) investigated hardware extensions in ASIPS and proposed a configurable hardware structure. A novel algorithm for designing pipelined configurable hardware extension based on integer linear programming (ILP) has been devised. In this study, configurable custom functional units (CFUs) have been used for effectively sharing resources between multiple candidate custom instructions. ASIPs have emerged as a promising architectural alternative for designing embedded systems. ASIPs have the benefit of improved performance due to their customized instruction set and specialized ISEs (Instruction Set Extensions) designed for a particular application. Energy efficiency in ASIPs can be improved by optimizing components like register file, cache, instruction fetch stage, etc. This study specifically addresses static energy overhead in ASIPs caused due to integration of custom hardware. Design trade-offs between performance improvement and resource efficiency have been analyzed and demonstrated with experimental results by various researchers working in design space exploration for ASIPs.

III. CONTEMPORARY ENERGY-EFFICIENT PROCESSORS

The prominent players in microprocessors design market are Intel, ARM, AMD, MIPS, SPARC, etc. Power efficiency is a design consideration taken care by these vendors while designing architectures for specific applications. ARM processors are way ahead of its competitors in the market as far as energy-efficiency is concerned. These processors are preferred to be used for handheld and other embedded devices due to its energy efficient designs. ARM's Cortex-A53 is the most energy-efficient application processor. It is also among the world's smallest 64-bit processor. The architecture of Cortex-A53 is shown in Figure-2. ARMv8-A architecture is implemented in Cortex-A53 processor. It supports AMBA 4 ACE bus architecture or AMBA 5 CHI bus architecture. External interfaces in this architecture include Accelerator Coherency Port (ACP) which implements AXI slave interface.

Intel's Core M is the world's first 14-nanometer fan-less processor designed specifically for improving energy-efficiency in laptops and tablets. Due to the non requirement of cooling fan, these processors can be used effectively for developing electronic devices with improved portability and extended battery life. These core M processors will be included in the upcoming models of laptop from manufacturers like Dell, Lenovo, Toshiba, Acer, Hewlett-Packard, etc.

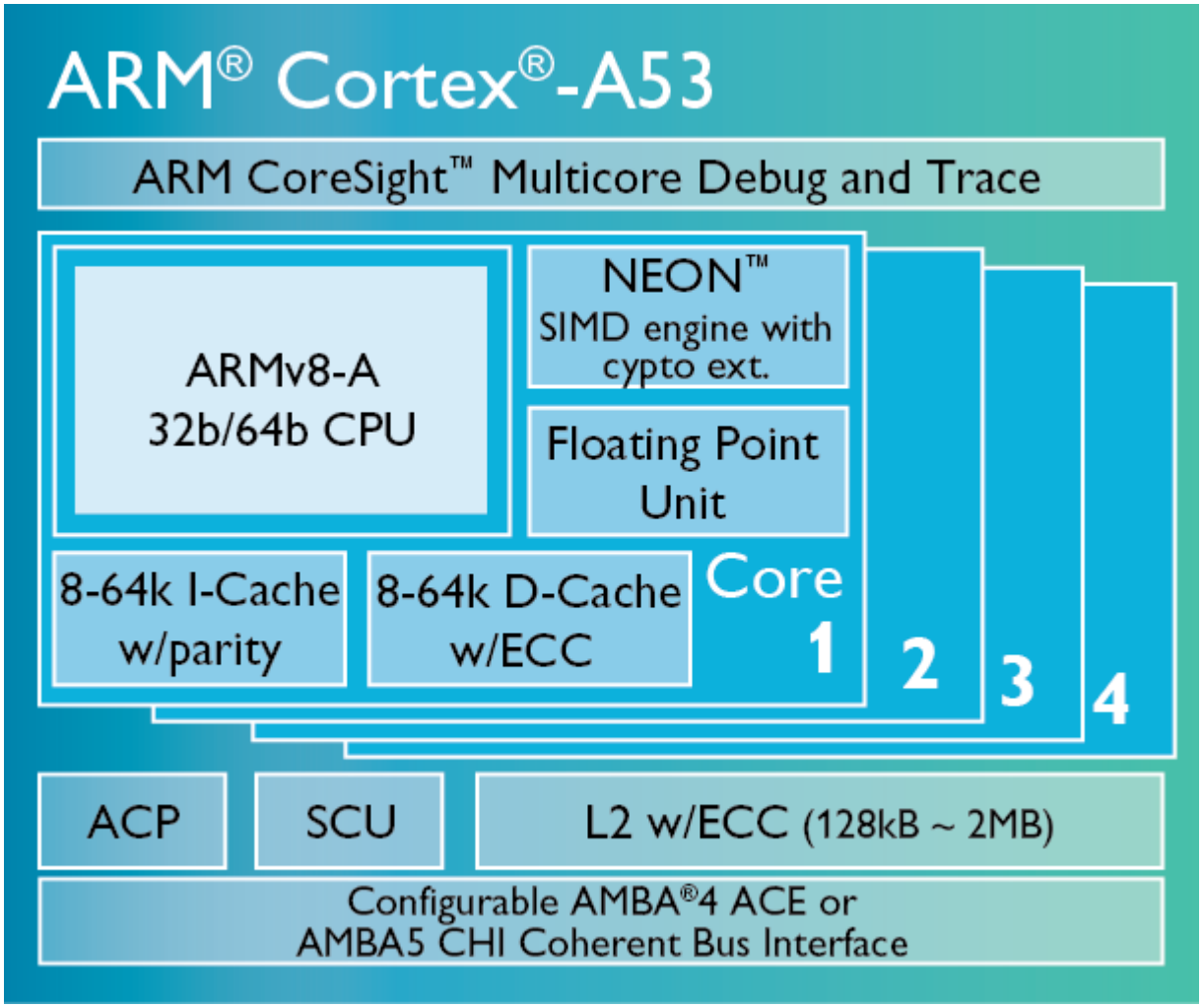

Figure-2: Cortex-A53 Processor

Source: ARM

IV. CONCLUSION

In modern embedded processors, energy efficiency is a vital design metric. ASIPs are designed to improve the performance and energy efficiency of processors for a specific application. Cache, register file, instruction fetch stage, etc. are some of the components which can be optimized to provide low power operation in ASIPs. Energy efficiency issue has been studied by various researchers but mostly the effect of dynamic energy consumption has been targeted till now. Static energy consumption also contributes significantly to the total power consumption in processors. Static energy consumption due to custom hardware extension can be reduced by improving the resource efficiency. In this paper, we have outlined the necessity for considering energy-efficiency as a vital design consideration while designing application specific processors. We have also discussed the contemporary energy efficient microprocessors that can be effectively used while designing embedded applications. Future work will include more insights into various techniques for improving energy efficiency in ASIPs.

REFERENCES

1. Lee, J., Choi, K., & Dutt, N. (2003). Energy-Efficient Instruction Set Synthesis for Application-Specific Processors. *Proc IEEE International Symposium on Low Power Electronics and Design, Seoul, South Korea*, 330-333.
2. Glokler, T., & Meyr, H. (2004). *Design of Energy-Efficient Application-Specific Instruction Set Processors (ASIPs)*. Dordretch: Springer.
3. *ARM® Cortex® -A53 MPCore Processor*. Retrieved from ARM website: http://infocenter.arm.com/help/topic/com.arm.doc.ddi0500f/DDI0500F_cortex_a53_r0p4_trm.pdfJ
4. Goodman, J., & Chandrakasan, AP. (2001). An energy-efficient reconfigurable public-key cryptography processor. *IEEE Journal of Solid-State Circuits, 36*, 1808-1820, doi: 10.1109/4.962304
5. Lin, H., & Fei, Y. (2012). Resource Sharing of Pipelined Custom Hardware Extension for Energy-efficient Application-specific Instruction Set Processor Design. *ACM Transactions on Design Automation of Electronic Systems, 17*, 158-165, doi: 10.1145/2348839.2348843