A Hierarchical Reconfigurable Micro-coded Multi-core Processor for IoT Applications

Ning Ma, Zhuo Zou, Zhonghai Lu, Lirong Zheng
iPack Vinn Excellence Center
School of Information and Communication Technology, KTH
Forum 120, 164 40 Stockholm-Kista, Sweden
Email: {mning, zhuo, zhonghai, lirong}@kth.se

Stefan Blixt
Imsys AB
Stockholm, Sweden
Email: stefan.blixt@imsystech.com

***Abstract*—This paper presents a micro-coded multi-core processor featuring reconfigurability and scalability with high energy efficiency for IoT domain-specific applications. By simplifying the control logic and removing the pipelines, the gate count of one core is minimized to 14 K. Meanwhile, all the hardware units are directly controlled and can be reorganized by the long microinstructions. High utilization of the hardware is thus achieved when designing the micro programs properly. Furthermore, both the ISAs for C and Java have been implemented by the micro programs to supply the general-purpose programmability. Besides, application-specific instructions can be further developed once higher performance is demanded in specific scenarios. Depending on the performance requirement, the activity and working strategies of the cores are adjustable. Moreover, several processors can be further connected to construct a network with the integrated router for even higher performance. As a case study, the AES encryption is implemented using both C and micro programs. More than 10 times of performance improvement is achieved when using micro programs on the single core, and 20 times on two cores.**

I. Introduction

In the vision of Internet of Things (IoT), objects surrounding us will be uniquely identified and connected with each other seamlessly as the “Smart object” [1], [2]. Those objects in everyday life will be equipped with the miniaturized electronic devices to possess the capability of sensing, processing and communicating. The possible applications vary from smart home, health care, retail to smart grid, smart city, environmental surveillance, etc [3]. The diversity of the objects and the application scenarios demands a flexible and reconfigurable hardware architecture to deal with the application-specific functions [4]. Meanwhile, the pervasiveness of those devices necessitates the compact design with the high area and energy efficiency.

The state-of-the-art processors for IoT applications are mainly RISC (Reduced Instruction Set Computer) processors owing to the simple hardware architecture and low power consumption [5], [6], [7]. The RISC design methodology is opposing to the CISC (Complex Instruction Set Computer) architecture that utilizes rich instructions to achieve high code density and low access frequency to the main memory. These features were important when the CISC was developed in the early years to save the costly memories. Oppositely, RISC processors manage to accelerate the execution of simplified instructions. The performance is increased with the high working clock frequency and high processing throughput resulted from the simple hardware and deep pipelines. The performance keeps increasing with the continuously shrinking of CMOS transistors. However, the deep pipelines also lead to the speculation penalty, and to avoid that, complex hardware mechanism has to be included. Moreover, the deployment of caches is inevitable to bridge the gap between the fast running core and the slow main memory.

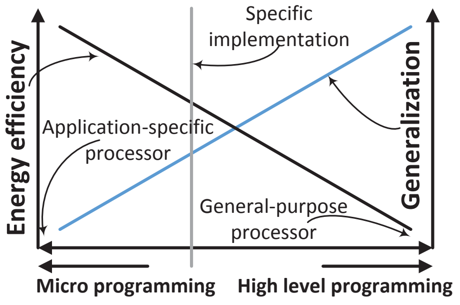

Fig. 1. Tunable working strategies of the proposed processor

In IoT applications, the high energy efficiency with proper performance is desired for the remote devices. Additionally, the memory space is constrained for the reason of the limited area and power budget. In this case, the CISC processor is a competitive solution. We thus propose the scalable and reconfigurable micro-coded multi-core processor. All the hardware units are directly controlled and can be reorganized by the long microinstructions. By designing the micro programs properly, the hardware can be utilized efficiently. Furthermore, both the ISAs for C and Java have been implemented by the micro programs to supply the general-purpose programmability. Besides, application-specific instructions can be further developed to provide higher performance and energy efficiency. As shown in Fig. 1, for different applications, the implementation by using micro programs and high-level programs can be tuned in a suitable way to satisfy the specific requirement. Moreover,

978-1-4799-5810-8/14/$31.00 ©2014 IEEE

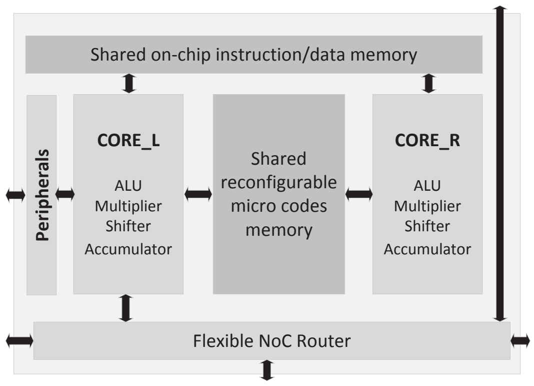

Fig. 2. System architecture

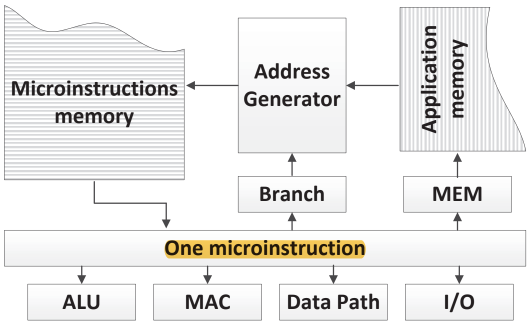

Fig. 3. Microinstruction-controlled hardware architecture

processors can be further connected to construct a network with the integrated router for even higher performance. We have also implemented the encryption algorithm using both C and micro programs to demonstrate the performance.

II. Processor Architecture

The architecture of the proposed multi-core processor is shown in Fig. 2. CORE_L and CORE_R have the same computational functionality. The ALU is used for the normal arithmetic and logic operation. Besides, a multiplication and accumulate unit (MAC) with a shifter is included to increase the computational capability. The microinstructions are stored in the shared memory, and can be fetched by each core for configuring the functional elements and data paths.

A. Reconfigurable micro-coded core

To design a suitable architecture for IoT applications, high area and energy efficiency are the important design targets. The hardware execution units and data paths should be utilized efficiently once activated. Instead of using wide data path to achieve high performance, we manage to make the 8-bit data path and ALU work efficiently to obtain the suitable performance. The micro-coded architecture is employed for each core in our system. As shown in Fig. 3, all the function elements, data paths, I/O, memory access and control logic are controlled by the long microinstructions. Being designed properly, the microinstructions can make all the hardware work in parallel to achieve very high energy efficiency. All the microinstructions are stored in the internal memory, and can be reprogrammed to supply the reconfigurability. The address of next microinstruction is calculated based on and in parallel with the execution of the current microinstruction.

Deepening the pipelines for the instructions execution is one of the most important methods to shorten the critical path and increase the system clock frequency. However, the inevitable inserted registers enlarge the area on one hand, and complicate the execution of instructions on the other hand. The result is the low area and energy efficiency. In our system, we avoid the use of pipelines and compensate the speed with the efficiency. The architecture is simplified and the speculation penalty is avoided. The gate count of one core is less than 14 K.

The instruction set architectures (ISA) designed for C and Java have also been implemented by the micro programs for the general programmability. Owing to the use of micro coding, very complex operations can be done in one instruction. The generated compact instructions lower the access frequency to the main memory and ease the requirement for external memory. Besides, the access of external memory is also controlled by the microinstructions, and is in parallel with the operation of the computational logic. As a result, caches are not necessary. The advantages are the save of the memory space as well as making the program execution more predictable.

B. Multi-core processor

Due to the simplification of the single core and efficient control of each hardware units by microinstructions, the area of the single core is small but with the high utilization. For a single core, to guarantee the sufficient execution time and make the critical path short as well, one executing cycle consists of two clock cycles. The address for the next microinstruction is calculated in the first cycle and thus the content is available in the second cycle by accessing the microinstructions memory, as shown in Fig. 4(a). Meanwhile, the execution of the microinstruction is also performed in two system clock cycles. If only one core is employed, in each clock cycle, only 0.5 microinstruction could be executed and half of the memory bandwidth is utilized.

As the shrinking of CMOS technology and the possibility of including more memories, it is the memory instead of the computational logic that occupies the most of the chip area. It is also the case in our system. To increase the utilization of the memories and thus improve the performance, the second core is integrated. The two cores share the same microinstructions memories and on-chip main memories. Since these two cores could access the memories alternatively as shown in Fig. 4(b), no arbiter is required for all the memories, which also makes the hardware simple and execution time predictable. In this case, the overall performance is one microinstruction per cycle.

The shared on-chip main memories host the instructions and data for both cores. For some IoT applications with the requirement of small storage space, the on-chip memories are large enough without the need of the external memory.

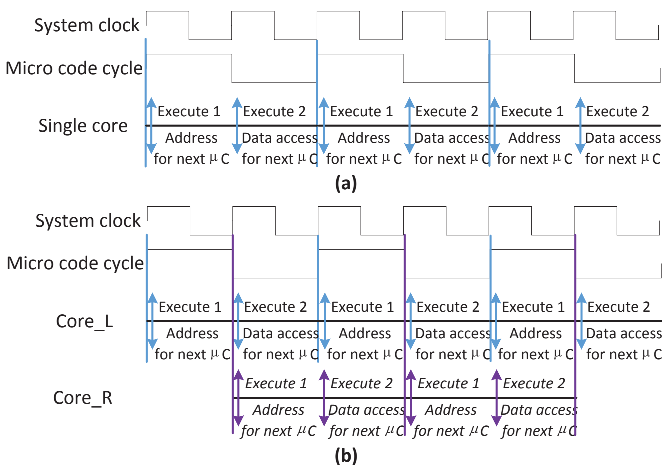

Fig. 4. Microinstruction executing cycle

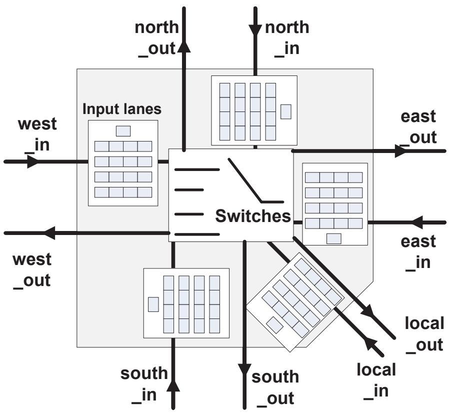

Fig. 6. Integrated router

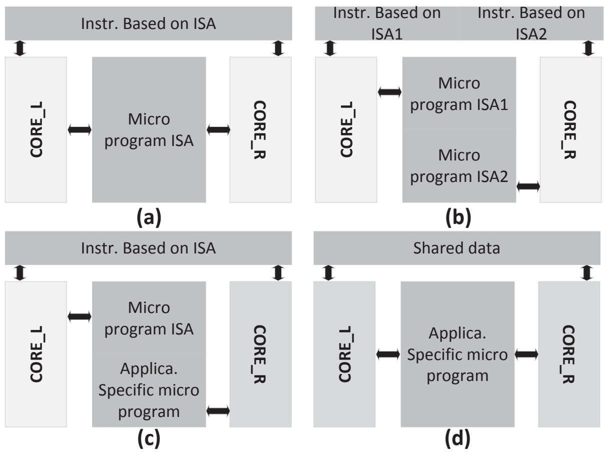

Fig. 5. Multi-core working strategies, (a) common ISA, (b) individual ISA, (c) accelerated ISA, (d) application-specific processor

Otherwise, the off-chip memories will be used, and CORE_L can access those memories directly with the microinstructions while the memory controller is not necessary. Meanwhile, the on-chip and off-chip memories share the same memory address space easing the high level design.

Since both of the cores are micro-coded processors, they can be reconfigured to different working modes as shown in Fig. 5. Two cores can work on the same ISA as a general-purpose multi-core processor. Two different ISA can also be supported individually by the two cores in Fig. 5(b). To achieve the high energy efficiency in some applications, the suitable micro programs can be tailored directly according to the requirement making the system work as an application-specific processor in Fig. 5(d). One core works as a co-processor or an accelerator for the other core in Fig. 5(c) to keep the general programmability while achieving high performance.

C. Integrated router

Fig. 6 shows the structure of the hybrid router integrated for connecting other processors. There are mainly three parts: input lanes, switches and output channels. By integrating the router, the processor can construct short-ranged off-chip network for multi-processor systems. The router will forward the data to its neighbors according to the routing algorithm if current node is not the destination, otherwise, send the data to the processor if current node is the destination. To pass the data through the network, both circuit switching and wormhole switching are supported to adapt to various data volumes and different data types. For wormhole switching, four virtual channels are deployed to supply the possibility of sharing the physical channels among different transactions. As shown in Fig. 2, the hybrid router is connected to CORE_L, so CORE_L is able to select the suitable switching mode for data transmission according to the characteristics of the data and the status of the network.

D. Hierarchical reconfigurability

To sum up, the hierarchical reconfigurability of the proposed architecture is shown in Fig. 7. In the top layer, thanks to the support of Java by the micro programs, the platform-independent IoT applications with Java implementations are able to directly run on the proposed architecture. For the various IoT applications with different performance requirement, we can make the suitable configurations for the system to obtain high energy efficiency while achieving the required performance. In the layer 5, various high-level programming languages have been supported on the system, and the user is able to choose the favorable one for the application development. Accordingly, besides the general purpose assembly codes, application-specific instructions can be further developed to increase the performance in the assembly coding layer. Since the ISA is implemented by the micro programs, various ISAs can be realized in the proposed system if necessary. The processor can thus choose the suitable ISA to execute the specific instruction in the reconfigurable ISA layer.

All the hardware units are directly controlled by the specific parts in the long microinstruction, and the reconfigurability in the microinstruction layer is presented by the capability of reorganizing all the hardware units. The compact micro programs lead to the high utilization of the hardware and thus high energy efficiency. In the hardware layer, the activity and speed

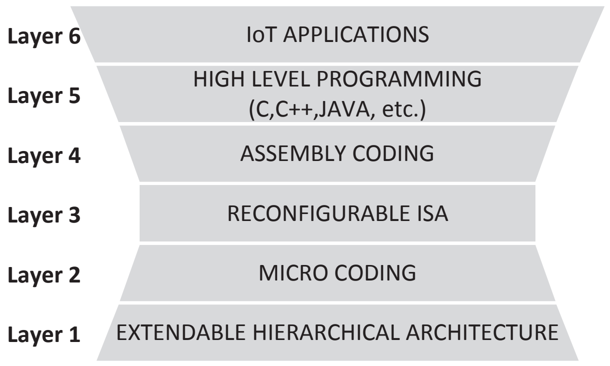

Fig. 7. Hierarchical reconfigurability

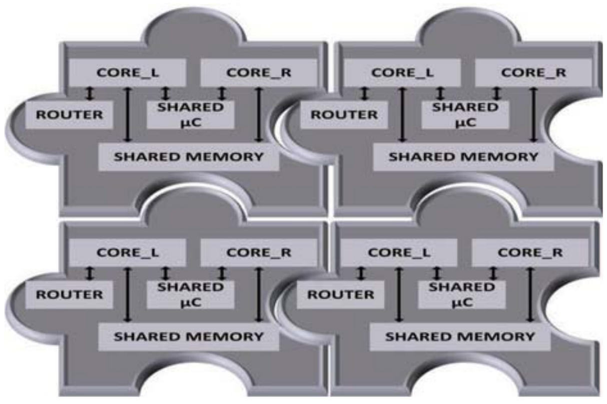

Fig. 8. Scalability of the processor

of the two cores are configurable for the specific application. Besides, when higher performance is further needed, we can connect multiple processors in one package or on one board to construct the simple network to increase the performance as shown in Fig. 8.

III. Case study

Privacy and security are the main concerns for the IoT applications. The encryption is thus the very important function. Many encryption algorithms are available, and all of them need complex computation on the plain text with a specific key. It is a challenge to implement those encryption algorithms on the IoT processors due to the limited computational capability. In the proposed system, we can make use of micro coding to reorganize the hardware units exploiting the maximum potential of the processor to realize the algorithms.

Advanced Encryption Standard (AES)[8] has been widely employed to encrypt the data in the Internet. The encryption is performed on 128 bits plain test organized in a 4x4 matrix. The length of the key can be 128, 192 or 256 bits, based on which 10, 12 or 14 rounds of multiple-stage operations will be executed for each 4x4 matrix. As shown in Fig. 9(a), the key generation and encryption process have been implemented by micro programs, and the interfaces are supplied at the same time for the high level reference. Compared with the implementation using C programming, more than 10 times

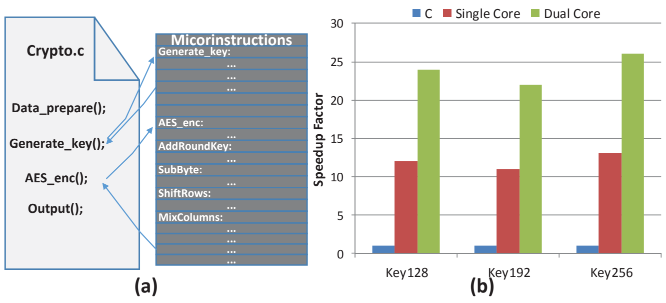

Fig. 9. Encryption with AES, (a) optimized using microinstructions, (b) performance for key sizes of 128, 192, 256 bits

performance improvement is achieved with micro-coded solution on one core and 20 times once dual cores are involved as shown in Fig. 9(b).

IV. Conclusion

A hierarchical reconfigurable micro-coded multi-core processor for IoT applications is presented in this paper. By simplifying the hardware architecture and employing long microinstructions to directly control all the hardware units, the processor can achieve the high energy and area efficiency. The proposed system can also be tuned between the general-purpose and application-specific processor with the suitable usage of micro programs. Moreover, several processors can be further connected to construct a network with the integrated router once higher performance is necessary. As a result, the processor can supply the hierarchical reconfigurability from the application layer to the extendable hardware layer. As a case study, the AES encryption has been implemented using both C and micro programs. More than 10 times of performance improvement is achieved when using micro programs on the single core, and 20 times on dual cores.

Acknowledgment

This work was financially supported by Vinnova (The Swedish Governmental Agency for Innovation Systems) through the Vinn Excellence centers program.

References

[1] H. Kopetz, “Internet of things,” in *Real-Time Systems*, ser. Real-Time Systems Series. Springer US, 2011, pp. 307–323.
[2] G. Kortuem, F. Kawsar, D. Fitton, and V. Sundramoorthy, “Smart objects as building blocks for the internet of things,” *IEEE Internet Computing*, vol. 14, no. 1, pp. 44–51, Jan 2010.
[3] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer Networks*, vol. 54, no. 15, pp. 2787 – 2805, 2010.
[4] O. Vermesan and P. Friess, Eds., *Internet of Things - Global Technological and Societal Trends*. River Publishers, 2011.
[5] A. Y. Dogan, J. Constantin, D. Atienza, A. Burg, and L. Benini, “Low-power processor architecture exploration for online biomedical signal analysis,” *IET Circuits, Devices Systems*, vol. 6, no. 5, pp. 279–286, 2012.
[6] E. Wenger and J. Grossschadl, “An 8-bit AVR-based elliptic curve cryptographic RISC processor for the Internet of Things,” in *Microarchitecture Workshops (MICROW), 2012 45th Annual IEEE/ACM International Symposium on*, Dec 2012, pp. 39–46.
[7] N. Ickes, Y. Sinangil, F. Pappalardo, E. Guidetti, and A. Chandrakasan, “A 10 pj/cycle ultra-low-voltage 32-bit microprocessor system-on-chip,” in *2011 Proceedings of ESSCIRC (ESSCIRC)*, 2011, pp. 159–162.
[8] NIST, “Advanced encryption standard,” *NIST FIPS PUB 197*, 2001.