An Energy-Efficient Heterogeneous Dual-Core Processor for Internet of Things

Zhibo Wang * Yongpan Liu * Yinan Sun * Yang Li † Daming Zhang * Huazhong Yang *

* Nanoscale Integrated Circuit & System Laboratoray
Dept. of EE Tsinghua Univ. Beijing
{wangzb13,syn08,zdm06}@mails.tsinghua.edu.cn
{ypliu,yanghz}@tsinghua.edu.cn

† Dept. of Microelectronics and Nanoelectronics
Tsinghua Univ. Beijing
yanglithu@gmail.com

***Abstract*—With the fast development of Internet of Things (IoTs) in recent years, many IoT applications, such as structure health monitoring, surveillance camera and etc, require both extensive computation for burst-mode signal processing as well as ultra low power continuous operations. However, most of conventional IoT processors focus on ultra low power consumption and cannot satisfy those demands. This paper proposes a novel energy-efficient heterogenous dual-core processor, which includes both an ultra low power near-threshold CoreL and a fast CoreH to meet those emerging requirements. Furthermore, an optimal framework is proposed to realize energy efficient task mapping and scheduling. The processor is fabricated and its energy consumption in low power mode is as low as $7.7pJ/cycle$ and outperforms related work. Detailed analysis under several real applications shows that up to $2.62\times$ energy efficiency improvements can be achieved without deadline miss compared with the high-performance-only signle core architecture.**

I. Introduction

Internet of Things are becoming more and more widely used in medical care, environment monitoring, intelligent transportation and etc. People can sense the physical world easily by deploying battery-powered sensors everywhere. To keep sensors working for enough time without frequent maintenance, most tasks are light and a low power processor with high energy efficiency is adopted in previous IoT applications. However, the development of IoTs applications, such as bridge monitoring and surveillance camera, are requiring more and more emergent tasks to be processed occasionally with a time deadline. Therefore, an energy-efficient processor satisfying those emerging requirements is highly demanded.

Dynamic voltage and frequency scaling (DVFS) [1] is a popular technique to reduce power while retaining the ability to process burst tasks with tight deadline. However, it is difficult to simultaneously optimize a processor design for both high performance and low power. [2] shows that multi-mode DVFS designs tend to have significant energy overheads when they are optimized for high performance modes. On the other hand, a core optimized for low power application is energy inefficient in the high performance mode.

BIG.LITTLE architecture [3], [4] integrates a high performance "BIG" core and a low power "LITTLE" core on a single chip. Designers can optimize the energy efficiency of two cores separately for either high performance or low power applications. However, traditional BIG.LITTLE processors are designed for general purpose computing [3] or mobile phones [4]. They can not meet the rigid energy constraints for IoT applications.

On the other hand, near-threshold computing(NTC) can reach optimal energy efficiency [5] and it is a promising technique to realize ultra low power processor for IoT. But the energy efficiency improvements come at the cost of performance loss. Conventional near-threshold processors usually violate the deadline constraints for burst mode task processing.

This paper proposes an energy-efficient heterogeneous dual-core processor CoreLH, which integrates an energy efficient near-threshold coreL and a high performance normal voltage coreH. The rationality of this architecture is that most tasks in the IoT do not need much computing ability and coreL can be used for higher energy efficiency by near-threshold computing. CoreH is used to process heavy tasks in order to meet deadline. Comparing with single power domain designs, this architecture can achieve the maximum energy efficiency point and saves much power. This paper tries to solve the following challenges: the design of the heterogeneous architecture and the energy efficient scheduling method to utilize the advantages of CoreLH.

The specific contributions of this paper are listed as the following:

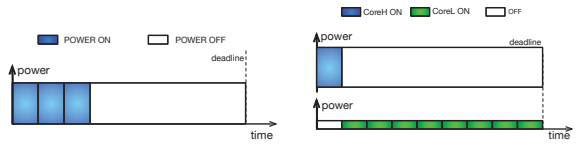
1) We propose an energy-efficient heterogenous dual-core processor architecture and implement the hardware design.
2) We develop a task scheduling framework for the proposed architecture, which improves the energy efficiency of several IoT applications, i.e., bridge monitoring, automatic controlling.
3) Experimental results show that the energy efficiency of the near-threshold core is as low as $7.7pJ/cycle$. We analyze the power savings under different task patterns and the results show that the architecture can improve energy efficiency by up to $2.62\times$ without deadline miss.

The rest of the paper is organized as follows: Section II gives the motivation and introduces the heterogenous architecture. Section III describes the hardware implementation. Section IV proposes the formulation of the task scheduling problem. Section V shows measurement results of the chip and the energy savings of the architecture on different benchmarks. Section VI concludes the paper.

II. Proposed System Design

This section firstly presents a motivational example, and then introduces the system architecture of this work.

A. Motivation



(a) High performance processor (b) CoreLH architecture

Fig. 1. A power saving example

This work was supported in part by the NSFC under grant 61271269 and High-Tech Research and Development (863) Program under contract 2013AA01320 and the Importation and Development of High-Caliber Talents Project of Beijing Municipal Institutions under contract YETP0102.

978-1-4799-8391-9/15/$31.00 ©2015 IEEE

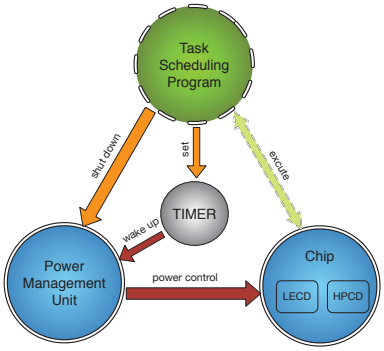


Fig. 2. System architecture

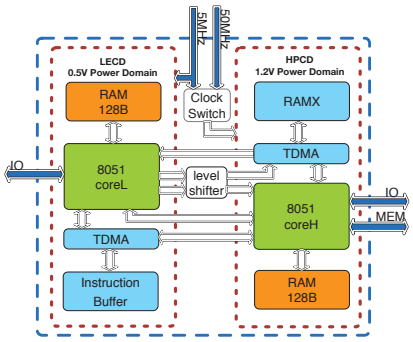


Fig. 3. Heterogeneous dual-core processor architecture

In the IoT, many tasks have relatively loose deadlines, making it possible to save power with the help of near-threshold computing. Fig. 1 gives an example. In Fig. 1(a), it takes a conventional single core processor 3 cycles to finish a task and the energy consumption is 3. There is enough time before the deadline. However, different things happen in the CoreLH architecture. CoreL is 4 times slower than CoreH but its energy efficiency is $3\times$ higher. As Fig. 1(b) shows, a proper scheduler makes CoreH to run 2 cycles and the task is then transferred to CoreL. In this way, the total energy is 1.67 and 44% energy can be saved.

B. System Architecture

The architecture of the system is shown in Fig. 2. The heterogeneous chip is used for processing different kinds of tasks, including a low energy CoreL domain (LECD) and a high performance CoreH domain (HPCD). The task scheduling program (TSP) is used for task and power management according to offline generated scheduling sequences. The power management unit (PMU) and timer help with power management of the submodules and task switching. The flow of a task scheduling procedure can be described as follows:

1) When each core finishes tasks on it, it will run TSP, and determine when to process the next task and set the timer;
2) TSP informs the PMU to power gate the corresponding cores or submodules if necessary;
3) When the time set in procedure 1) comes, the timer tells PMU to wake up the corresponding core domain or submodules;
4) The core which is waken up will process the next task, and turn to procedure 1).

The key challenges are how to implement the heterogeneous architecture, how to reduce the power of each part, how to communicate and share data between different power domains, and how to schedule tasks for lower energy consumption. The hardware and software challenges will be solved in Section III and Section IV respectively.

III. Hardware Implementation

The hardware architecture of the heterogeneous dual-core processor is shown in Fig. 3. There are 2 power domains on the chip, the $0.5V$ near-threshold power domain (or LECD) and the $1.2V$ normal voltage power domain (or HPCD). Level shifters [6] are used in order to shift $0.5V$ signals to $1.2V$, enabling the communication between the 2 power domains.

An 8051-compatible processor core with $128B$ RAM is deployed for task processing in each power domain respectively. The high performance CoreH uses an on board FLASH instruction memory, while the low energy CoreL utilizes an on chip near-threshold instruction buffer for power savings. A $4kB$ RAMX is used for data sharing between 2 cores by time-division memory access (TDMA). A clock switch which can provide non-glitch clock switching helps with synchronous communication between cores and RAMs in the 2 power domains.

Details of core and memory architecture design will be discussed in the following subsections.

A. Core design

The high performance CoreH uses normal standard cell CMOS process, while the CoreL working at $0.5V$ needs special considerations because many standard cells do not work well at near-threshold voltage. To implement the CoreL, we only use carefully selected standard cells to provide correct logic and acceptable delay. Then we evaluate the area and power efficiency of CoreL at different clock frequency constraints and choose a proper operating frequency for CoreL to get an acceptable tradeoff among area, energy and performance.

B. Sub-threshold instruction buffer

An on board FLASH memory serves as the instruction memory of the proposed processor. The FLASH can store the program of all tasks. However, tasks in the IoT are relatively simple and the instructions of a single task are short. Thus, much power can be saved if a small instruction buffer is utilized. Considering that CoreH does not work for a long time, we only implement a $1kB$ near-threshold instruction buffer for CoreL. And CoreH uses the on board FLASH memory directly.

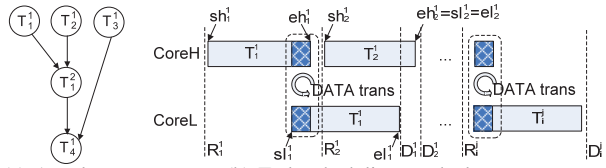
When a task is scheduled to CoreL, instructions of the task will be transferred to the instruction buffer before CoreL starts. Thus, the $1.2V$ power domain can be totally turned off if necessary and the system can work at maximal energy efficiency point. However, transferring instructions and key data of a task to the instruction buffer brings extra costs of both energy and time. The tradeoff will be evaluated in Section IV and the evaluation results will be shown in Section V.

C. Time-division memory access

The processor needs to transfer tasks and some important data between the 2 cores, so a highly efficient data sharing memory architecture is required. Conventional architecture uses bus arbitration protocols to avoid memory access conflicts, which consumes much power. In the CoreLH architecture, however, the memory access patterns are known and can be managed by software. Thus, we use a time division memory access (TDMA) method whose power is quite low and use software to control memory access.

IV. Task Scheduling Formulation

Previous works have studied task scheduling problems in different constraints. [7] has studied the scheduling between different cores



(a) A task set (b) Task scheduling on dual cores

Fig. 4. Task scheduling model

with DVFS, and [8] has considered periodic real-time tasks on multi-core processors. However, the CoreLH architecture is different from DVFS or conventional BIG.LITTLE architecture because a task has to start on CoreH and transferring a task to CoreL brings time and power costs. Thus, previous task scheduling methods cannot be used for our chip and a new task scheduling method is needed.

The new task scheduling problem will be formulated in IV-A, and the solution of the problem will be introduced in IV-B.

A. Problem formulation

We use a Direct Acyclic Graph (DAG) to present a task set. As shown in Fig. 4(a), each node T_i^j represents the j^{th} instance of $Task_i$, while an edge $e(T_i^j, T_m^n)$ indicates that T_m^n have to start after T_i^j finishes. The scheduling result can be presented as Fig. 4(b). T_i^j has to run after its release time R_i^j and finish before its deadline D_i^j. It can either run on coreH only (e.g. T_2^1) or transferred to coreL (e.g. T_1^1 and T_i^j) to save power. It should be noted that it will cost time and energy when a task is transferred from CoreH to CoreL. The formulation can be described as follows:

Given Variables:

- T_i^j: The j^{th} instance of $Task_i$,
- R_i^j: The release time of T_i^j,
- D_i^j: The deadline of T_i^j,
- W_i: The workload of $Task_i$ ($cycles$ needed),
- S_H, S_L:The computing speed of coreH and coreL ($cycles/s$),
- Ttr_i: Time needed to transfer $Task_i$ from coreH to coreL,
- Ptr: The power of both cores when transferring tasks,
- P_H, P_L: The power of each core when computing.

Variables for Optimization:

- $\{sh_i^j, eh_i^j, sl_i^j, el_i^j\}$: The start time and end time on coreH and coreL of T_i^j,
- f_i^j:The flag to show if T_i^j is transferred to coreL ($f_i^j = 1$ if transferred).

Objective:

- Minimize the total energy consumption: E_{total}

The optimization objective E_{total} can be described as:

$$E_{total} = \sum_i \sum_j [P_H \cdot (sl_i^j - sh_i^j) + P_L \cdot (el_i^j - eh_i^j) + f_i^j \cdot Ptr \cdot Ttr_i] \quad (1)$$

The sum symbols mean to add the power of all T_i^j, and 3 parts of power are power for computing on CoreH, CoreL, and power for data transfer. The goal is to determine $sh_i^j, eh_i^j, sl_i^j, el_i^j$ and f_i^j for the minimization of E_{total}.

According to Fig. 4(b), sl_i^j and eh_i^j are not independent:

$$sl_i^j = eh_i^j - f_i^j \cdot Ttr_i \quad (2)$$

The workload constraints of the CoreLH architecture can be described as:

$$S_H \cdot (sl_i^j - sh_i^j) + f_i^j \cdot S_L \cdot (el_i^j - eh_i^j) = W_i \quad (3)$$

The sequence of each time point is:

$$S_i^j \le sh_i^j \le sl_i^j \le eh_i^j \le el_i^j \le D_i^j \quad (4)$$

The time slot each task on each core should not overlap, which can be described as:

$$\begin{aligned} (sh_i^j - eh_m^n) \cdot (eh_i^j - sh_m^n) > 0 \\ (sl_i^j - el_m^n) \cdot (el_i^j - sl_m^n) > 0 \end{aligned} \quad (5)$$

From Equations (2) - (5), we can decide the optimal values of $sh_i^j, eh_i^j, sl_i^j, el_i^j$ and f_i^j for minimal E_{total}.

B. Task scheduling solution

The problem formulated in the last subsection is a mixed integer quadratic programming (MIQP) problem, which has an optimal solution. We use a MIQP server to get the scheduling results of each benchmark in Section V-A and evaluate the power savings of the CoreLH processor. It should be noted that the tasks cannot be scheduled in some cases because of tight time constraints or heavy workload, and we will relax some deadline constraints in these cases.

V. Measurement Results and Scheduling Evaluation

In this section, measurement results of the chip will be presented first. Then the power efficiency improvements of some practical benchmarks will be analyzed. Finally, the energy savings in general cases will be shown.

A. Experiment setup

The chip is fabricated with CMOS $130nm$ technology. We use an IoT application to measure the power costs of each core. It is noticed that CoreH uses an on board FLASH instruction memory, while CoreL uses an on chip near-threshold instruction buffer. To compare the 2 cores fairly, we measure the total power of each power domain. Then we get the percentage of core power by simulation, and finally we calculate the power of each core.

In order to evaluate the energy savings of the CoreLH architecture in practical applications, we use 3 typical IoT benchmarks to compare the energy of the CoreLH processor with conventional processors. A brief description of each benchmark is listed below:

1) A temperature sensing node (TempSense) which senses environment temperature and then sends it to remote server by wireless communication ($Task\,1, 2$);
2) A bridge monitoring node (BridgeMonitor) which collects and analyzes the vibration information about a bridge, and then compresses and sends valuable information to remote servers for further analysis ($Task\,3, 4$);
3) A realtime automatic controller (AutoControl) which can control an industry machine hand, and interact with humans ($Task\,5-7$).

The power of each core (P_H, P_L, Ptr) is from measurement results, and the key parameters of each task are listed in Table I. It should be noted that the workload of each task has been normalized to the running time on coreH.

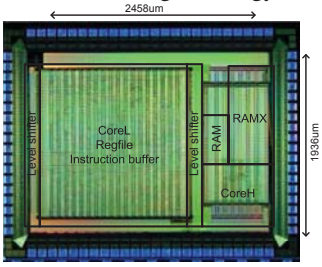
TABLE I
Parameters of the Tasks

	Function	Norm workload	Coming frequency	deadline	$\mathbf{Ttr_i}$
$\mathbf{Task_1}$	Sense temperature	$10us$	$100ms$	$100ms$	$800us$
$\mathbf{Task_2}$	Send data to server	$6ms$	$1min$	$50ms$	$1.4ms$
$\mathbf{Task_3}$	Sense vibration and analyze	$600us$	$10ms$	$10ms$	$900us$
$\mathbf{Task_4}$	Compress and send data	$7.5ms$	$1s$	$10ms$	$1.4ms$
$\mathbf{Task_5}$	Get information and calculate	$1ms$	$10ms$	$10ms^*$	$1ms$
$\mathbf{Task_6}$	Control machine	$1ms$	$10ms$	$10ms^*$	$900us$
$\mathbf{Task_7}$	Update GUI	20ms	$1s$	$1s$	$1ms$

* $Task_5$ and $Task_6$ have same release time and deadline.

B. Measurement results

The micrograph and specification summary of the chip is shown in Fig. 5. The measurement is performed at $26°C$ and the deviation among 3 measured chips is negligible. Measurement results and comparison with related works are shown in table II. In high performance mode, the energy of core is $21.3pJ/cycle$. In low power mode, the energy of the core is $7.7pJ/cycle$, which is similar to work [1], [9]. However, the maximal frequency of this work in low power mode is quite higher than the others' because we did not lower the voltage that much considering performance. And the CoreLH processor has the potential to achieve higher energy efficiency if the voltage is lower.



Chip Summary	
Process Technology	CMOS $130nm$
RAM Size	$128B$
RAMX Size	$4KB$
CoreH Frequency	$50MHz$ @$1.2V$
Regfile Size	$128B$
Instruction Buffer Size	$1KB$
CoreL Frequency	$5MHz$ @$0.5V$

Fig. 5. Chip micrograph and specification summary

TABLE II
COMPARISON WITH STATE-OF-THE-ART PROCESSORS

		This Work	[9]†	[1]	[10]
Process		$130nm$	$180nm$	$65nm$	$180mn$
CPU Architecture		MCS51	MCS51	32b RISC	RISC BSP
Voltage(V)		$0.5-1.2$	$0.23-1.8$	$0.2-1.2$	$0.5-1.8$
High Efficiency Mode	Core Energy/Cycle	$7.7pJ$	$7.9pJ$	$9.94pJ$‡	–
	Frequency	$5MHz$	$10kHz$	$133kHz$	–
High Performance Mode	Core Energy/Cycle	$21.3pJ$	$20.5pJ$	$110.2pJ$‡	$20.4pJ$
	Frequency	$50MHz$*	$43.5MHz$	$94.3MHz$	$10MHz$

† The chip is not fabricated, and the data is simulation result;
‡ RAM energy is included.
* Max frequency is simulation result because the flash used for measurement is not fast enough;

C. Practical benchmarks evaluation

Simulation results of real case benchmarks are shown in Table III. The results show that the CoreLH architecture can save much power compared with the single high performance processor. To our surprise, the CoreLH chip only saves 4.4% for $TempSense$ whose workload is quite light. The reason is that $Task_1$ in $TempSense$ is too short to be transferred to CoreL for higher energy efficiency, because a task transfer will consume power and time which is not worth for short tasks. Another interesting result is that $BridgeMonitor$ wastes more power for task transfer than $AutoControl$, but its relative power saving (42%) is a lot more than that of $AutoControl$ (21%) because the slightly heavy workload of $AutoControl$ can make full use of the benefits of near-threshold computing while suffering from similar task transfer energy cost. But the tight time constraints make it impossible to transfer all tasks to CoreL, so the total power of $AutoControl$ rises.

TABLE III
SIMULATION RESULT

	TempSense	BridgeMonitor	AutoControl
Norm workload (per second)	$1.1ms$	$67.5ms$	$220ms$
Power on single high performance processor	$1.17uW$	$72uW$	$234uW$
Power on the CoreLH chip	$1.12uW$	$41uW$	$186uW$
Power for task transfer	$2.65nW$	$10uW$	$8uW$
CoreH utilization (computing)	0.102%	0.75%	13.7%
CoreL utilization (computing)	0.789%	60%	83.1%
Time for task transfer (percent)	0.002%	9.09%	7.3%
Power saving	4.4%	43%	21%

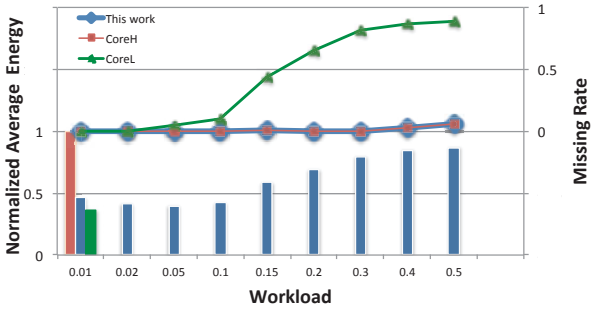


Fig. 6. Normalized average energy and miss rate

D. General cases evaluation

We evaluate the relationship between energy consumption and workload, and the result is shown in Fig. 6. When $W = 0.05$, the energy consumption of the CoreLH chip equals to 8.13pJ/cycle without missing deadline. It achieves $2.62\times$ better energy efficiency than the high-performance-only single core processor. On the other hand, a pure near-threshold processor will miss deadline when the workload is high and cannot be used in many IoT applicatons. The result also shows that when the average work load is between 0.01 and 0.1, which is common in the IoT, the power saving of the CoreLH chip is significant.

VI. CONCLUSIONS

This paper proposes an energy efficient heterogeneous dual-core processor for the IoT and evaluates the energy savings of this architecture by running multiple kinds of tasks. The processor is fabricated and its power is only $7.7pJ/cycle$ in low power mode. According to simulation under several real applications, tasks with slightly high workload and loose time constraints will contribute most to improving the power efficiency. Detailed analysis shows that up to $2.62\times$ energy improvements can be achieved without deadline miss compared with the high-performance-only single core architecture.

REFERENCES

[1] S. Luetkemeier, T. Jungeblut, M. Porrmann, and U. Rueckert, "A 200mV 32b subthreshold processor with adaptive supply voltage control," in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*. IEEE, 2012, pp. 484–486.

[2] A. B. Kahng, S. Kang, R. Kumar, and J. Sartori, "Enhancing the efficiency of energy-constrained dvfs designs," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 21, no. 10, pp. 1769–1782, 2013.

[3] R. Kumar, K. I. Farkas, N. P. Jouppi, P. Ranganathan, and D. M. Tullsen, "Single-ISA heterogeneous multi-core architectures: The potential for processor power reduction," in *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*. IEEE, 2003, pp. 81–92.

[4] P. Greenhalgh, "Big.LITTLE processing with ARM cortex-a15 & cortex-a7," *ARM White Paper*, 2011.

[5] B. Zhai, D. Blaauw, D. Sylvester, and K. Flautner, "Theoretical and practical limits of dynamic voltage scaling," in *Proceedings of the 41st annual Design Automation Conference*. ACM, 2004, pp. 868–873.

[6] B. Zhai, L. Nazhandali, J. Olson, A. Reeves, M. Minuth, R. Helfand, S. Pant, D. Blaauw, and T. Austin, "A 2.60 pJ/inst subthreshold sensor processor for optimal energy efficiency," in *VLSI Circuits, 2006. Digest of Technical Papers. 2006 Symposium on*. IEEE, 2006, pp. 154–155.

[7] Y. Zhang, X. S. Hu, and D. Z. Chen, "Task scheduling and voltage selection for energy minimization," in *Proceedings of the 39th annual Design Automation Conference*. ACM, 2002, pp. 183–188.

[8] W. Y. Lee, "Energy-saving dvfs scheduling of multiple periodic real-time tasks on multi-core processors," in *Proceedings of the 2009 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications*. IEEE Computer Society, 2009, pp. 216–223.

[9] W. Jin, S. Lu, W. He, and Z. Mao, "A 230mV 8-bit sub-threshold microprocessor for wireless sensor network," in *VLSI and System-on-Chip (VLSI-SoC), 2011 IEEE/IFIP 19th International Conference on*. IEEE, 2011, pp. 126–129.

[10] X. Liu, J. Zhou, X. Liao, C. Wang, J. Luo, M. Madihian, and M. Je, "Ultra-low-energy near-threshold biomedical signal processor for versatile wireless health monitoring," in *Solid State Circuits Conference (A-SSCC), 2012 IEEE Asian*. IEEE, 2012.