International Journal of Advanced Research in IT and Engineering

ISSN: 2278-6244
Impact Factor: 6.111

ENERGY MINIMIZATION TECHNIQUES OVER MULTICORE PROCESSING SYSTEM: A REVIEW

K. Nagalakshmi , Department of Computer Science and Engineering, Hindusthan Institute of Technology, Coimbatore, Tamilnadu, India

N. Gomathi, Department of Computer Science and Engineering, Vel Tech Dr. RR & Dr. SR Technical University, Chennai, Tamilnadu, India

Abstract: *Energy efficiency and processor performance have become key metrics in the designing of multicore computational systems. Due to breaking down of Moore's law, increasing energy savings without compromising raw performance is considered as a major limiting factor in multicore architecture. Recent technological advances in energy minimizing methods of multicore system substantially meet the contradictory demands of low power, low cost, small area and outstanding performance. This paper aims at ascertaining more competent energy-minimizing techniques for managing energy consumption of multicore processor through investigations. We highlight the necessity of the energy savings techniques and study several novel technologies to focus their pros and cons. This paper is intended to serve the researchers and architects of multicore processors in accumulating ideas about the energy savings techniques and to incorporate it in near future for more effective fabrications.*

Keywords*: Clock Gating; DVFS; Energy Efficient Design; Multicore Processor; Task Scheduling;*

I. INTRODUCTION

At present, multicore architectures (MCA) are becoming dominant design paradigm which assimilates two or more processing elements (cores) in a single die for higher performance computing. Proliferation of heavy computational requirements of real time applications in MCAs leads severe energy efficiency and performance constraint to provide quality of service (QoS) to the users. Thus, designing the MCAs to resolve power-performance tradeoff is a challenging endeavor. The research and design community have invested significant efforts in exploring several energy efficient technologies to scale their performance and make sure reliability, prolonged existence and acceptance in wide range of applications.

As stated in the Moore's law, the number of transistors fabricated in a single chip approximately doubles every 18 to 24 months [1], resulting in an exponential increase in transistor density. This indicates that the speed (clock frequency) of the processor will also double in every 18 months. However we cannot enjoy this exponential growth continuously due to its increasing power density on chip which prevents all the cores to be switched on simultaneously. This utilization barrier is called as Dark silicon, is driving the emergence of heterogeneous MCAs [2]. Multicore processors can be categorized into three types: Homogeneous [3, 6, 8], Heterogeneous [10, 11] and Dynamic reconfigurable processors [12]. Conventionally, most of the general purpose MCAs are built with identical cores. All these cores consist of same micro-architectural innovations (i.e., cache memory, out-of-order execution, speculation, pipeline, branch prediction configuration, etc.) and are able to operate under same instruction set architecture (ISA). This type of architecture is called as a homogeneous or symmetric multicore architecture (SMP). It is easy to design and implement as we just need to duplicate the core.

INTEL CORE i7 [3], AMD PHENOM [4] and SUN NIAGARA [5] multicore architectures are general purpose SMPs with large cache memories. These processors are designed for general purpose desktop and server applications where energy efficiency is not a primary concern. In contrast, the homogeneous architectures XMOS-XS1 [6] and ARM-CORTEX [7] are specially designed for mobile devices, where energy efficiency is an increasing concern. Some of the MCAs are developed for high performance computing. Therefore, they employ larger number of cores. For example, AMD RADEON 700 GPU [8] contains 160 cores while NVIDIA G200 [9] contains 240 cores. Though homogeneous cores are simple to design, easy to implement and provide regular software environments; they cannot deliver required performance and energy efficiency for different real time applications. Real time applications with different QoS encourage the computer architects and software designers to exploit architecture innovations and design heterogeneous multicore processors.

Asymmetric Multicore Processor (AMP) or Heterogeneous processor implements mixture of non-identical processing elements that are asymmetric in their underlying principles and performance. The cores are varying in size and complexity, but they are designed to cooperate with each other to increase the performance of the system. These designs provide an efficient solution for dark silicon era and also increase reliability, performance

and energy efficiency of the applications. A typical AMP will integrate small (slow) cores to process simple tasks in an energy efficient way, and complex (fast) cores to provide higher performance.

ARM big.LITTLE [10] and Cell BE [11] are renowned examples for AMP architectures. Cell BE is extensively used in gaming devices and computing platforms aiming high performance computing. ARM big.LITTLE is designed for mobile platforms where complex, performance driven, quad core Cortex A15 assembly is combined with simple, power-optimized, quad core Cortex A7 assembly to provide peak performance.

Dynamic heterogeneous multicore architectures are able to reconfigure itself at run time to regulate their performance, speed and complexity level based on application requirements. It has the ability to resolve the power-performance tradeoff by integrating efficient hardware with flexible software. By introducing adaptability and hardware flexibility, this dynamic architectures can achieve high performance within the energy budget and therefore to meet the QoS requirements of real time applications. Flexible heterogeneous Multicore processor (FMC) is an eminent example of dynamic reconfigurable architecture that can deliver both an increased throughput for uniformly distributed parallel workloads and outstanding performance for fluctuating real time tasks [12]. Depending upon the application requirements, the FMC can scale up and down its computing resources such as memory engines (ME), functional units, and pipelines that are anticipated to improved performance.

The evolution of multicore designs opens up a new space of research called energy saving techniques. Reducing the peak and average energy consumption could have a positive impact on performance of multicore processors, starting from circuit level to system level. In the last decade, several researches have been keen to explore various energy saving techniques in multicore regime. The remaining part of our survey is structured as follows: Section II presents the basics of power dissipation in multicore domain and also highlights the necessity of energy saving techniques. Section III provides an overview and taxonomy of classic power management techniques and explores some of these techniques in detail. Finally, Section IV is arranged to provide the conclusions of our investigation.

II. BACKGROUND

A. Basics of Power dissipation

Nowadays, multicore is ubiquitous both in general purpose and application specific computing systems starting from smartphones to commercial servers. Power dissipation is an important constraint because it increases the temperature and cooling costs, reduces reliability, and degrades performance. Power consumed by CMOS devices can be resolved into three parts [13] as shown in equation (1).

$$P_T = P_{st} + P_{dy} + P_{sc} \quad (1)$$

Where P_T is total power dissipation, P_{st} is static or leakage power due to leakage of a transistor's bias currents. P_{dy} is dynamic power due to switching of transistors. P_{sc} is power dissipation related to concurrent conduction of p type and n type transistors.

$$P_{st} = V.I_L \quad (2)$$

Where V denotes source voltage, I_L is the transistor leakage current. Usually, leakage power contributes 20 to 40 % of the total power dissipation [14]. Dynamic power is a prevalent factor as compared to other two components in equation (1). It can be denoted as follows

$$P_{dy} = \beta C_L . V^2.f \quad (3)$$

Where β is activity factor, C_L indicates the effective load capacitance and f is the switching speed. To simplify equation (3), it is assumed that the clock speed is linearly proportional to the source voltage. If we apply this notion to the above equation (3) then, the reduction in source voltage and switching speed lowers dynamic power cubically.

$$P_{dy} \alpha V^3 \quad (4)$$

Short circuit power is calculated by the following equation

$$P_{sc} = V.I_{sc} \quad (5)$$

Here I_{sc} represents the short circuit current flowing from supply to ground. Short circuit power is comparatively trivial for static CMOS circuits.

B. Failure of Dennard scaling

For almost 30 years, the computing community has realized a stable performance evolution in uniprocessor, motivated by Moore's Law [1] and Classical (Dennard) scaling [15]. But now this curve is slowed down and came to halt due to memory wall, power wall and Instruction Level Parallelism (ILP) wall [16]. Under Dennard scaling, the power requirements per unit

space can remain constant across semiconductor generations. According to this principle [15], with a linear dimension scaling ratio of 0.707, the transistor count could doubles (Moore's Law), frequency increases by 40%, but the power consumed per transistor is reduced by half keeping the total chip power constant in every two years [17]. From equation (3), the power density in a chip area A is measured as follows

$$\text{Dynamic Power density} = (\beta C_{Load} \cdot V^2 \cdot f)/A \qquad (6)$$

As we move towards the next generation of IC manufacturing technology, the linear size of an IC gets scaled by 0.707. The same scaling ratio is applied for load capacitance and supply voltage while clock speed is scaled by 1⁄0.707. So the area of the chip is now 0.707^2A. If we calculate a new power per unit space, we have $0.707CX0.707V^2Xf/(0.707^3XA)$. Hence, the power per unit area becomes unchanged. But unfortunately, in 65nm technology and below, this law ceased because of exponential growth in leakage current and reduction in supply voltage decreases the speed of the processor. Nonetheless, the high performance demand is continued and this stimulates a shift from the single core to a multicore paradigm.

C. Need for Energy Saving Techniques

Today's innovations in semiconductor technology lead to not only an increase in the number of cores on a die, but also an increase in power density and concomitant heat dissipation. Increasing power dissipation leads many negative impacts on power delivery, performance/watt (PPW) ratio, packaging and cooling costs, reliability, availability and overall performance of the processors. So energy consumption issues occasionally more important than speed of the processor.

Energy efficiency is essential in mobile electronics where devices are battery powered. For last few decades, processor performance has been accelerating at a rate faster than the evolutions in battery technologies. This has led to a considerable drop of the battery life in mobile devices. At the same time, modern computational intensive applications demand very high performance. These two conflicting requirements, the need to conserve energy and the demand to deliver outstanding performance lead new approaches to resolve it. Existing researches to achieve optimal energy budget have two significant guidelines. First is in what way to increase the processor's efficiency within a specified energy limit. Second is

in what way to decrease the energy consumption of computing devices without compromising processor performance.

III. OVERVIEW OF CLASSIC ENERGY SAVING TECHNIQUES

In this section, we delve into the up-to-date techniques in energy saving of MCAs. As of now, several hardware and software approaches have been adopted for alleviating power and energy costs. Through this investigation, we aim to demonstrate how the research community is trying to achieve an outstanding performance and energy efficiency. We can classify the power management techniques into three broad categories: Hardware techniques, Hardware-enabled middleware techniques and Software techniques.

A. Hardware techniques

Several energy saving techniques with dedicated controllers are embedded into the modern processor architectures to provide energy efficiency. Applications running in a multicore domain need a carefully tailored computing architecture to meet their QoS within the power budget. The architectural innovations in designing of core, memory and interconnection networks improve the energy efficiency significantly.

1) Core Layout

Puttaswamy et al. propose a 3D microarchitecture with Thermal Herding techniques, which provides outstanding PPW ratio [18]. Compared to a conventional planar processor the proposed architecture can achieve 15% to 30% of active power reduction depending on the characteristics of the application. But 3D architecture incurs augmented power per unit area and associated temperature issues. This problem is resolved by Fazal Hameed et al. They present a thermal-aware 3D microarchitecture that effectively integrates the potential gains of dynamic architectural adaptation, fail-safe DVFS, and global migration [19]. Research shows that thermal-aware 3D architecture can achieve significant power reduction over 3D multicore processors [18] because it can reduce the active and the leakage powers simultaneously.

Kontorinis et al. introduce an adaptive processor with peak power guarantees which can reduce peak power by table-driven reconfiguration [20]. Most of the functional units (e.g. ALU, L1 cache, register files, load-store units and so forth) of the processor are dynamically organized for power conservation and maximum performance whereas peak power

constraints are assured. Adaptive processor can reduce peak power about 25% with a smaller amount of performance cost.

Rodrigues et al. propose Dynamic Core Morphing (DCM) architecture for heterogeneous multicores [21]. The resources of the cores are morphed at runtime based on varying performance requirements. Depending on the computational need of the current workload, two cores may swap the execution units to maximize the PPW ratio.

The Scalable stochastic processor developed by Narayanan et al. has been demonstrated as an auspicious way to tackle power dissipation problem for error-tolerant applications such as audio or video. Improved scalability is realized by substituting or augmenting conventional computational units by gracefully degrading functional units [22]. The scalability leads power savings range between 20% and 60% in the well-known H.264 video encoder.

2) Memory Design

Many researches are carried out to bring innovations in memory organization in order to minimize the power dissipation. Smart caching [23] emphases on power saving computing techniques and implements way-predicting caches with reduced leakage designing techniques. Flaunter et al. [24] explore the use of instruction pre-fetch algorithms combined with the drowsy caches, where cache lines are periodically put into a low power mode without considering their access histories. Implementation of a drowsy cache in a 0.07µm CMOS process can reduce 50% to 75% of the total energy consumption in the caches. Cai and Lu present a joint venture for saving energy in system memory and hard disk unanimously [25]. This technique periodically reconfigures the size of physical memory by adding or freeing up the allotted memory pages and uses a timeout policy for shutting down the hard disk. The suitable memory size and timeout are selected according to their proportionality with the average power consumption. This technique achieves energy savings higher than 50% over a fixed-timeout scheme.

3) Interconnection network

Several researches show that the design choices for interconnect fabric have significant impact on the power budget [20, 23]. The interconnect network itself is a power consuming resource. The power consumption of the interconnection network for a 16 core processor is more than the combined power consumption of two cores. Rakesh Kumar et al. [26]

demonstrates the need for careful co-design of interconnect network and memory hierarchy. The power consumption of the core increases super linearly with the number of connected units and average length of wire. So a power-optimized architectural design needs compact length of interconnection wire segments and appropriate routing algorithms [23].

B. Hardware-Enabled Middleware techniques

The following techniques are employed as middleware and partially implemented in hardware. The hardware enables middleware to shut down or slow down the functional units according to the operating temperature. Hardware-enabled middleware techniques including Stop-and-Go [27], Dynamic Voltage and Frequency Scaling [29], Advanced Configuration and Power Interface [50] and different Gating Techniques [44,46] have attracted a great deal of attention.

1) Stop-and-Go

The stop-and-go is the simplest form of dynamic power management (DPM) technique [79]. The DPM techniques reduce the power consumption by shutting down or lowering the performance of idle cores. Stop-and-go can be realized on both global and local scale. In global approach, if one of the cores reaches its specified threshold temperature, this scheme shut down the whole chip until its non-critical level has been recovered. If stop-and-go is realized locally, only the overheating core will be halted until it has cooled down. Global stop-and-go mechanism provides a smaller amount of control and less efficiency as a particular overheating core leads to unwanted delaying of all other non-critical cores.

Donald et al. [27] implement 12 combinations of local and global stop-and-go policies with other power management techniques (i.e. DVFS and Task migration) for managing the temperature of multicore processors. They investigate the pros and cons of each combination by comparing their performance. Whenever peak temperature of the processor reaches 84.2° C the stop-and-go controller shuts down the cores for 30msec to allow the cores to cool down. Their implementation results show that local stop-and-go can outperform global schemes. Chaparro et al. [28] propose a stop-and-go mechanism with clock gating mechanism. Whenever the core reaches its critical temperature, this combined technique halts the core, stores its current state information, and then shuts the core off completely. There is no dynamic as well as leakage power consumption in this technique. So

it allows the overheating core to cool down quicker. As the current state of the core is saved before shutting down, this method would be the ideal choice of the options.

2) Dynamic voltage and frequency scaling (DVFS) [29]

DVFS is the prevailing and powerful DPM technique, used to regulate the power consumption of the processor by dynamically scaling the level of supply voltage and clock frequency [29, 30, 31, 32]. The sub-threshold leakage current and gate-oxide tunneling leakage can be reduced by reducing supply voltage [33]. Reducing the clock frequency reduces the supply voltage linearly and decreases power consumption quadratically [34]. DVFS is widely used for memory bound workloads and can be employed in two ways:

1. The Local (per-core) DVFS allow us to scale the voltage of individual cores, so that the overheating core can cool down faster [28, 35, 36].
2. The Global DVFS allow us to adjust the voltages and frequencies of all cores uniformly and simultaneously. Similar to stop-and-go, a single hotspot on one of the cores could result to unnecessary performance penalty on all cores [28, 35, 37].

Using Local DVFS introduces more flexibility as each core can select its own voltage–frequency pair individually. However, that suffers from a large number of expensive inherent voltage regulators. The global DVFS can solve the thermal issues faster but the efficiency of DVFS is affected by limited flexibility to determining a single optimal voltage to all cores.

Weiser et al. present the first paper to suggest an interval based DVFS for reducing the power dissipation in computing devices. Their work focuses on three scheduling algorithms: Unbounded-delay perfect-future (OPT), Bounded-delay limited-future (FUTURE), and Bounded-delay limited-past (PAST) [38]. The deployment of each algorithm controls the clock frequency and makes the scheduling decisions simultaneously. The PAST scheduling algorithm with a 50msec adjustment interval can achieve power conservation of 50% to 70% based on circuit conditions.

Wonyoung et al. develop a fast, per-core DVFS mechanism with on-chip integrated voltage regulators [32]. This mechanism uses the potential benefit of both per-core voltage regulation and very fine-grained voltage switching. The in-built regulators can increase the energy efficiency opportunities of DVFS and result in 21% of energy savings over conventional global DVFS with off-chip regulators.

Many researchers have unified DVFS technique with thread migration polices to reduce energy consumption. Cai et al. develop a new thread shuffling algorithm, which integrates thread migration and DVFS techniques on a MCA supporting simultaneous multithreading (SMT) [39]. Thread shuffling dynamically migrates slower threads with same criticality degrees to a particular core and implements DVFS for other cores having fast threads. The proposed scheme realizes energy savings around 56% with no performance degradation. Quan Chen et al. propose an Energy-Efficient Workload Aware (EEWA) task scheduler, that consists of a work load aware frequency adjuster and a preference-based task-stealing scheduler [40]. With the help of DVFS, the workload-aware frequency adjuster can accurately configure the frequencies of the cores in an efficient fashion based on the profiled workload statistics. The preference-based task-stealing scheduler can successfully distribute the tasks across various cores at runtime according to the preference list. The EEWA can save energy about 28.6% with only 0.9% of performance loss.

All the previous works cited above, simply fail to consider the static power that has turn into a substantial portion of the total power consumption, unfortunately. LeSueur and Heiser [41] assess the factors influencing the efficiency of DVFS on AMD Opteron processors, using an extremely memory-bound benchmark. They illustrate that the ability of DVFS is retreating in modern digital systems due to escalating leakage power. Furthermore, their investigation reveals that switching-off idle cores will facilitate greater energy savings. To reduce leakage power, Awan et al. [42] propose an enhanced race-to-halt (ERTH) approach. By integrating DVFS and slack management policies, ERTH can improve energy efficiency considerably.

When global DVFS is realized in MCA, determination of optimal voltage that satisfies all cores is a challenging endeavor; some applications will suffer from performance penalty or overheads. This issue exacerbates as the running applications and number of cores in next generation processors. From a hardware implementation point of view, local DVFS is more expensive than global DVFS, because of its costly inherent voltage regulators and phase-locked loops. However, the per-core DVFS provides better tradeoff between performance and power.

3) Gating Techniques

Clock Gating [44] and Power Gating [46] are very useful methods for decreasing dynamic and static power correspondingly [43]. Gating techniques are realized by insertion of an

additional logic between the clock source and clock input of the processor's circuitry. It diminishes power consumption by logically turning off (gating) the power to the portions of core that are not useful to the current workload.

a) Clock Gating (CG) Techniques

The clock gating techniques employed in the Hexagon™ Digital Signal Processors (DSP) are analyzed by Bassett et al. [44]. The proposed four levels of clock gating and spine-based clock distribution allow switching off the power to the different regions, from single logic cell to entire chip. Further power reduction is achieved through a structured clock tree by distributing the clock signal across the chip with low skew and delay. This technique provides reduction in power consumption by 8% for active mode and over 35% for sleep mode.

Hai et al. describe a deterministic clock gating (DCG) technique, which hinge on the advance knowledge about at what time the functional block will be idle in the upcoming cycles [45]. With this advance information DCG can switch-off the idle blocks that maximize the energy efficiency. By exploiting DCG to various functional units, the proposed technique achieves 19.9% of average diminution in power without any performance cost. However, for all these techniques, the effectiveness of gating is restricted by the granularity of components that can be gated, the failure to change the overall size and complexity of the processor. Also, these designs are still vulnerable to leakage inefficiencies.

b) Power Gating (PG) Techniques

Power Gating (PG) is a circuit-level technique to reduce leakage power consumption by effectively turning off the source voltage to the idle elements. PG can be applied either at the core-level [46] or at the unit-level of the processor such as cache banks, ALUs, pipeline branches etc. [47, 48]. Recently, Intel Core i7 processors use power gating transistors to turn off its idle cores [49].

Hu et al. develop a parameterized model based on analytical equations, which decides the breakeven point used for proper gating. They evaluate the dynamic power gating ability of the *fpu* (floating-point units) and *fxu* (fixed-point units) of POWER4 processor by three techniques namely ideal, time-based, and branch-misprediction-guided [47]. The implementation of these techniques in various execution units shows that a considerable decrease in static power consumption can be realized through power gating.

Lungu et al. propose a Success Monitor Switch (SMS) and a Token Counting Guard Mechanisms (TCGM) for applying predictive power gating technique in POWER6 processor [48]. By employing SMS, the control logic enables or disables the PG depends on the success of policy. By implementing work for TCGM, this predictive power gating achieves a guarantee on the worst case execution of the policy. Leverich et al. [46] propose a Per-Core Power Gating (PCPG) technique with DVFS for datacenter workloads. This combined technique can save up to almost 60% of energy consumption.

4) Advanced Configuration and Power Interface (ACPI) [50]

ACPI is an industry standard for efficient handling of power management in computing devices. It is developed by the collaborative effort of Intel, Hewlett-Packard, Phoenix, Microsoft, and Toshiba [50]. ACPI provides platform-independent interfaces for power management and monitoring. These interfaces have the potential to work with existing DPM techniques [50]. ACPI is relay on operating system-directed configuration and Power Management (OSPM), which defines four switchable C-states (CPU idle states) C_0, C_1, C_2, and C_3 and n P-states (CPU-performance states) P_0 to P_n for active power management. ACPI allows the processor to achieve fine tuning of the power consumption by moving idle devices into lower power states (sleeping state). Bircher and John [51] point out the implicit and explicit performance impacts of various CPU-idle states and Performance states of AMD quad-core processors. They verify their results for both compute-bound and memory-bound applications with fixed and OS scheduling. They develop an enhanced hardware and operating system configurations that decreases average active power by 30% with 3% of performance loss.

C. Software techniques

The performance per watt ratio of a MCA is depends on efficient built-in hardware and the ability of software to effectively control the hardware. Many up-to-date processors exploit software level power management techniques for energy efficiency. Recently, researchers have paid greater attention to the software power management policies because it can gain the power disparity statistics of processing threads on the fly with low cost. Software techniques can achieve predictable performance through transferring or scheduling tasks to minimize thermal gradients and hot spots. Software-based approaches include data forwarding [52, 53] and task scheduling [54].

1) Data forwarding

Most modern processors use large size of on-chip L1 caches with multiple ports. Such a cache consumes a substantial part of the overall power owing to its larger size and high frequency access rate. Researches reveal that L1 data cache contributes 15% of the overall energy consumption of the processor [52]. Thus it is essential to develop tactics for precluding large power consumption in cache. Data forwarding is one of the appropriate solutions to reduce the energy consumption of L1 data cache.

Carazo et al. [53] propose a data cache filtering technique with forwarding predictor to reduce the power consumption of L1 data cache (DL1). This mechanism exploits an effective utilization of load-store queue (LSQ), which is responsible to provide the right data to load instructions by data forwarding method. Their experiments exhibit that the proposed cache filtering technique can achieve an average power savings up to 36% with a 0.1% of performance degradation. To reduce the access rate of DL1, Nicolaesu et al. propose a cached LSQ (CLSQ) to maintain load and store instructions after their execution [52]. Hitting in the CLSQ is faster and wastes not as much of energy as a DL1 access. Thus the significant savings in the frequency of accesses leads to 40% of energy reduction without any additional hardware complexity and performance penalty.

2) Task scheduling

Task scheduling is another breakthrough technique in power management arises from software approach. These algorithms are designed to solve temperature issues by distributing tasks among different cores. There have been wide ranges of literature put out on scheduling algorithms to achieve more processor utilization, better power conservation and more uniform power density without degrading the processor throughout. These algorithms schedule the tasks across cores based on predetermined temperature threshold.

Work proposed by Hsin-Hao Chu and Yu-Chon Kao is a perfect example of how an adaptive thermal-aware multi-core task scheduling algorithm with multiple run-time controllers can mitigate the inter-core thermal costs and dynamic variations of task execution [54]. Implementations of run-time controllers increase the system complexity. To resolve this problem, the temperature-aware task scheduling algorithm, called Low Thermal Early Deadline First (LTEDF) is suggested by Wu et al. The LTEDF allocates tasks based on a novel History Coolest Neighborhood First allocation algorithm [55]. Simulation of the LTEDF

algorithm demonstrates that it can satisfy the timing constraints for soft real time tasks and minimize the thermal consequences simultaneously.

Power aware task scheduling algorithms for MCAs can be classified into three types [56]: the global (dynamic binding) approaches [57], the partitioned (static binding) approaches [58] and the semi partitioned approaches [59, 60]. In the global approach, any core in the multicore system may execute any task. Global scheduling saves tasks in single priority-ordered queue, shared by all processors [57]. At every moment, the global scheduler chooses the highest-priority task for operation and the tasks are permitted to migrate between the processors. There are three types of priority assignment schemes for MCA: fixed-priority, [61, 58], dynamic priority [64] and Proportionate Fair (PFair) priority [65].

Fisher et al. develop a thermal-aware global scheduling algorithm for sporadic real-time tasks based on two priority assignment schemes, namely the global earliest-deadline-first (EDF) and the global deadline-monotonic (DM) [62]. The suggested schemes can substantially lessen the peak temperature around 30°C to 70°C as compared to load-balancing strategies.

Wang et al. develop a scheduling approach for hard real-time systems. They execute delay analysis for generic task arrivals using First-In-First-Out (FIFO) scheduling and Static-Priority (SP) scheduling with reactive speed control techniques [62]. But Andersson and Baruah prove that the fixed priority scheduling algorithms cannot achieve a utilization bound greater than 50% [58]. Some researchers handle this deficiency by PFair priority assignment. Baker addresses the aforementioned problem and demonstrates a schedulability test for preemptive deadline scheduling of periodic or sporadic real-time tasks [64]. Baruah and Shun-Shii Lin propose a new Pinfair algorithm that is very efficient in terms of runtime complexity and has a superior density threshold for a very large subclass of generalized pinwheel task systems [65]. Levin et al. propose a deadline partitioning algorithm, called DP-WRAP algorithm to handle sporadic task sets with arbitrary deadlines [66].

In the partitioned approaches, task set is partitioned and statically allocated to a designated processor. These task set are executed by existing scheduling algorithms and migration across core is not permitted. Fan et al. present a Partitioned Scheduling algorithm with Enhanced RBound (PSER) that exploits a flexible task set scaling technique and enhanced utilization bound for fixed-priority periodic real-time tasks [67]. This algorithm effectively

improves the schedulability of the system. They combine PSER with Harmonic Aware Partition Scheduling (HAPS) in [68], which converts the complete task set into harmonic set and takes the benefit of harmonic relationship between tasks to achieve increased utilization bound up to 100% [69].

Andersson demonstrates global PFair and partitioned static-priority scheduling on multiprocessors [70]. Guan et al. develop two separate fixed-priority scheduling algorithms for light tasks and heavy tasks. The algorithm RM-TS/light (Rate monotonic-task set for light loads) can execute light task sets with sustainable parametric utilization bound and the RM-TS algorithm can perform any task set, whereas the utilization bound is lower than a specified limit [71].

Recently, a significant portion of semi-partitioned approaches [59, 60, 72, 73, 74, 75, 76] have been proposed to minimize energy expenditure in multicores. Semi- partitioned algorithms allocate most of the tasks to one particular processor. But, limited tasks (i.e. less than number of cores – 1) are partitioned into many subtasks and are allocated to various cores under some constraints.

Lakshmanan et al. introduce a Highest-priority task-splitting (HPTS) algorithm to enhance the utilization bound of partitioned deadline-monotonic scheduling algorithms (PDMS) from 50% to 60% on implicit deadline task sets [75]. They can obtain 88% of average utilization with very low migration overhead for randomly generated implicit-deadline task sets by extend this algorithm, which assigns the tasks in the decreasing order in terms of their size. Kato et al. [74] and Andersson et al [60] present real-time scheduling algorithms with high schedulability. Similar to partitioned scheduling, the proposed algorithms assign each task to a specific processor but can divide a task into two processors if there is not sufficient capacity remaining on a processor. The semi-partitioned approaches are more efficient as compared to the conventional global and partitioned approaches theoretically [75, 76, 77] and also suitable for practical implementations [73]. Zhang et al. show that the implementation complexity of semi-partitioned scheduling algorithm is relatively low. They investigate semi-partitioned approaches in the Linux OS and demonstrate their results on an Intel Core-i7 processor [78].

IV. CONCLUSIONS

Current innovations in semiconductor technology lead to not only an increase in the number of cores on a die, but also an increase in power density and concomitant heat dissipation. This adversely affects the system reliability and availability. Achieving high performance with low power consumption is imposing a new challenge on IC fabrication technology. This article presents various effective techniques for alleviating power dissipation of MCA and its classification based on their attributes. We accept as true that our review will help the researchers and architects to acquire ideas into the next-generation multicore processors and encourage them to endorse new energy efficient elucidations for fabricating competent architectures.

REFERENCES

1. Gordon E. Moore. Cramming more components onto integrated circuits. vol. 86(1); pp. 82 – 85, IEEE (1998)
2. Christian Martin.: Post-Dennard Scaling and the final years of Moore's Law consequences for the evolution of multicore-architectures (2014)
3. Intel Corp.: Intel core i7-940 processor. Intel Product Information, 2009 [Online]. Available: http://ark.intel.com/cpu.aspx?groupId=37148
4. Advanced Micro Devices Inc.: Key architectural features—AMD Phenom II processors. AMD Product Information, 2008 [Online]. Available: http://www.amd.com/usen/Processors/ProductInformation/ 0,301181533115917% 5E159 19, 00.html
5. Johnson, T., Nawathe, U.: An 8-core, 64-thread, 64-bit power efficient SPARC SoC (niagara2). In Proceedings of 2007 International Symposium on Physical Design (ISPD '07), pp. 2 – 2, ACM, (2007)
6. May, D.: XMOS XS1 architecture. Micro IEEE 2012; vol. 32(6); pp. 28 – 37.
7. ARM Ltd.: The ARM Cortex-A9 Processors. ARM Ltd. White Paper, Sept.2007 [Online] Available: http://www.arm.com/pdfs/ARM CortexA-9Processors .pdf
8. Advanced Micro Devices Inc.: ATI Radeon HD 4850 & ATI Radeon HD 4870—GPU specifications. AMD Product Information, 2008. [Online]Available: http://ati.amd.com/products/radeonhd4800/ specs3. html

9. NVIDIA Corp.: NVIDIA CUDA: Compute unified device architecture. NVidia CUDA Documentation, June2008. [Online] Available: http://developer.download.nvidia.com/compute/cuda/2_0/docs/NVIDIA_CUDA_Programming_ Guide_2.0.pdf
10. ARM Ltd.: ARM Development Tools, 2011. [Online] Available: http://www.arm.com/products/tools/development-boards/versatileexpress/index.php
11. Gschwind, M., Hofstee, H.P., Flachs, B., Hopkin, M., Watanabe, Y., Yamazaki, T.: Synergistic Processing in Cell's Multicore Architecture, vol. 26(2), pp.10 – 24, IEEE (2006)
12. Pericas, M., Cristal, A., Cazorla, F.J., Gonzalez, R., Jimenez, D.A., Valero, M.: A flexible heterogeneous multi-core architecture. In Proceedings of 16th International Conference on Parallel Architecture and Compilation Techniques, pp.13 – 24, IEEE (2007)
13. Zhuravlev, S., Saez, J.C., Blagodurov, S., Fedorova, A., Prieto, M.: Survey of Energy-Cognizant Scheduling Techniques, vol. 24(7), pp.1447 – 1464, IEEE (2013)
14. David Chinnery, Kurt Keutzer.: Overview of the Factors Affecting the Power Consumption. In proceeding of Tools and Techniques for Low Power Design, pp.11 – 53, Springer, (2007)
15. Dennard, R.H., Gaensslen, F. H., Yu, H., Rideout, V.L., Bassous , E., LeBlanc, A.R.: Design of ion-implanted MOSFET's with very small physical dimensions. In proceedings of IEEE Solid-State Circuits, vol. 12(1); pp. 38 – 50, IEEE (2007)
16. Hennessy, J.L., Pattersson, D.A.: Computer Architecture - A Quantitative Approach. Morgan Kaufmann, second edition (1996)
17. Hadi Esmaeilzadeh, Emily Blem, Renee St. Amant, Karthikeyan Sankaralingam, Doug Burger.: Dark silicon and the end of multicore scaling. In proceeding of 38th International Symposium on Computer Architecture (ISCA), pp. 365 – 376, IEEE (2011)
18. Puttaswamy, K., Loh.: Thermal Herding: Microarchitecture Techniques for controlling hotspots in high-performance 3D integrated processors. In proceeding of 13th

International Symposium on High Performance Computer Architecture, pp.193 – 204, IEEE (2007)

19. Fazal Hameed, Mohammad Abdullah Al Faruque, Jorg Henkel.: Dynamic thermal management in 3D multi-core architecture through run-time adaptation. In proceeding of Design, Automation & Test in Europe Conference & Exhibition, pp.1–6, IEEE (2011)
20. Kontorinis, V., Shayan, A., Tullsen, D., Kumar, R.: Reducing Peak Power with a Table-Driven Adaptive Processor Core. In Proceedings of IEEE/ACM 42nd Annual International Symposium on Microarchitecture (MICRO-42), pp. 189 – 200, IEEE (2009)
21. Rodrigues, R., Annamalai, A., Koren, I., Kundu, S., Khan, O.: Performance per Watt Benefits of Dynamic Core Morphing in Asymmetric Multicores. In Proceedings of International Conference on Parallel Architectures and Compilation Techniques, pp. 121–13, ACM (2011)
22. Narayanan, S., Sartori, J., Kumar, R., Jones, D.: Scalable Stochastic Processors. In Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), pp.335 – 338, IEEE (2010)
23. Kornaros G.: Multi-core Embedded Systems. Taylor and Francis Group, CRC Press, (2010)
24. Flautner, K., Kim, N., Martin, S., Blaauw, D., Mudge, T.: Drowsy Caches: Simple Techniques for Reducing Leakage Power. In Proceedings of 29th Annual International Symposium on Computer Architecture, pp. 148 – 157, IEEE (2002)
25. Le Cai, Yung-Hsiang Lu.: Joint Power Management of Memory and Disk. In Proceedings of the conference on Design, Automation and Test in Europe, pp. 86 – 91, IEEE Computer Society (2005)
26. Kumar, R., Victor Zyuban, Dean M. Tullsen.: Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In Proceedings of 32nd International Symposium on Computer Architecture (ISCA), pp.408 – 419, IEEE (2005)

27. Donald, D., Martonosi, M.: Techniques for Multicore Thermal Management: Classification and New Exploration. In Proceedings of 33rd International symposium on Computer Architecture, pp.78 – 88, IEEE (2006)
28. Chaparro, P., Gonzalez, J., Magklis, G., Cai, Q. Gonzalez, A.: Understanding the Thermal Implications of Multicore Archtectures. IEEE Transactions on Parallel and Distributed Systems, vol.18 (8), pp. 1055 – 1065, IEEE (2007)
29. Isci, C., Buyuktosunoglu, A., C.-Y. Chen, Bose, P., Martonosi, M.: An Analysis of Efficient Multi-Core Global Power Management Policies: Maximizing Performance for a Given Power Budget. In Proceedings of 39th Annual IEEE/ACM International Symposium on MICRO-39, pp. 347 – 358, IEEE (2006)
30. Hanumaiah, V., Vrudhula, S.: Energy-efficient Operation of Multicore Processors by DVFS, Task Migration and Active Cooling. Computers, vol. 63, no. 2, pp. 349 – 360, IEEE (2012)
31. B. de Abreu Silva, Bonato V.: Power/performance optimization in FPGA-based asymmetric multi-core systems. In Proceedings of 22nd International Conference on Field Programmable Logic and Applications (FPL), pp. 473 – 474, IEEE (2012)
32. Wonyoung Kim, Gupta M S, Gu-Yeon Wei, Brooks D.: System level analysis of fast, per-core DVFS using on-chip switching regulators. In Proceedings of 14th International Symposium on High Performance Computer Architecture, pp.123 – 134, IEEE (2008)
33. Dongwoo Lee, Wesley Kwong, David Blaauw, Dennis Sylvester.: Simultaneous Subthreshold and Gate-Oxide Tunneling Leakage Current Analysis in Nanometer CMOS Design. In Proceedings of 4th International Symposium on Quality Electronic Design, pp. 287 – 292, IEEE (2003)
34. Burd, T., Pering, T., Stratakos, A., Brodersen, R.: A dynamic voltage scaled microprocessor system. In Proceedings of International Solid-State Circuits Conference, pp. 294 – 295, IEEE (2000)
35. Jayaseelan, R., Mitra, T.: A Hybrid Local-global Approach for Multi-core Thermal Management. In Proceedings of 2009 IEEE/ACM International Conference on Computer-Aided Design, pp. 314 – 320, IEEE (2009)

36. Pruhs, K., Van Stee, R., Uthaisombut, P.: Speed scaling of tasks with precedence constraints in approximation and online algorithms. In Proceedings of 3rd International conference on approximation and online algorithms, pp.307– 319, Springer (2006)
37. March, J.L., Sahuquillo, J., Hassan, H., Petit, S., Duato, J.: A new energy-aware dynamic task set partitioning algorithm for soft and hard embedded real-time systems. vol. 54, no. 8, pp. 1282 – 1294, The Computer Journal (2011)
38. Weiser, M., Welch, B., Demers, A., Shenker, S.: Scheduling for reduced CPU energy. In Proceedings of 1st USENIX conference on OSDI, pp. 13 – 23, ACM (1994)
39. Cai Qiong, Gonzalez, J., Magklis, G., Chaparro, P., Gonzalez, A.Q.:Thread shuffling: Combining DVFS and thread migration to reduce energy consumptions for multi-core systems. In Proceedings of 2011 International Symposium on Low Power Electronics and Design, pp. 379 – 384, IEEE (2011)
40. Quan Chen, Long Zheng, Minyi Guo, Zhiyi Huang.: EEWA: Energy-Efficient Workload-Aware Task Scheduling in Multi-core Architectures. In Proceedings of Parallel & Distributed Processing Symposium Workshops (IPDPSW), pp.642 – 651, IEEE (2014)
41. Le Sueur E., Heiser G.: Dynamic voltage and frequency scaling: The laws of diminishing return. In Proceedings of Hot Power: Workshop on Power aware computing and systems, pp. 1– 8 (2010)
42. Awan, M.A, Petters, S.M.: Enhanced Race-To-Halt: A Leakage-Aware Energy Management Approach for Dynamic Priority Systems. In Proceedings of 23rd EUROMICRO Conference on Real-Time Systems (ECRTS), pp.92 – 101, IEEE (2011)
43. Li Li, Ken Choi, Haiqing Nan.: Activity-driven fine-grained clock gating and run time power gating integration, vol. 21(8); pp. 1540 – 1544 IEEE (2013)
44. Bassett, P., Saint-Laurent M.: Energy efficient design techniques for a digital signal processor. In Proceedings of International Conference on IC Design & Technology, pp.1–4, IEEE (2012)
45. Hai Li, Swarup Bhunia, Yiran Chen, Vijaykumar T N, Roy K.: Deterministic Clock Gating for Microprocessor Power Reduction. In Proceedings of 9th International Symposium on High-Performance Computer Architecture, pp.113 – 122, IEEE (2003)

46. Leverich, J., Monchiero, M., Talwar, V., Ranganathan, P.: Power management of data center workloads using per-core power gating. vol. 8(2); pp. 48 – 51, IEEE (2009)
47. Hu, Z., Buyuktosunoglu, A., Srinivasan, V., Zyuban, V., Jacobson Bose, P.: Micro architectural Techniques for Power Gating of Execution Units. In Proceedings of 2004 International Symposium on Low Power Electronics and Design, pp.32 – 37, IEEE (2004)
48. Lungu, A., Bose, P., Buyuktosunoglu, A., Sorin, D.: Dynamic Power Gating with Quality Guarantees. In Proceedings of International Symposium on Low Power Electronics and Design (ISLPED), pp. 377– 382, IEEE (2009)
49. Rajesh Kumar, Glenn Hinton.: A Family of 45nm IA Processors. In Proceedings of IEEE International Solid-State Circuits Conference, pp. 58 – 59, IEEE (2009)
50. Hewlett-Packard, Intel, Microsoft, Phoenix Technologies, Toshiba: Advanced Configuration and Power Interface Specification, Revision 4.0a. April 2010, [Online]. Available: http://www.acpi.info/spec.html.
51. Lloyd Bircher, Lizy, W., John, K.: Analysis of Dynamic Power Management on Multi-Core Processors. In Proceedings of 22nd Annual International Conference on Supercomputing, pp. 327–338, ACM (2008)
52. Dan Nicolaescu, Alex Veidenbaum, Alex Nicolau: Reducing Data Cache Energy Consumption via Cached Load/Store Queue. In Proceedings of 2003 International Symposium on Low Power Electronics and Design, pp. 252 – 257, IEEE (2003)
53. Carazo, P., Apolloni, R., Castro, F., Chaver, D., Pinuel, L., Tirado F.: L1 Data cache power reduction using a forwarding predictor, vol. 6448; pp.116–125, Springer (2011)
54. Hsin-Hao Chu, Yu-Chon Kao, Ya-Shu Chen.: Adaptive thermal-aware task scheduling for multi-core systems. vol. 99; pp.155–174, ELSEVIER (2015)
55. Wu, G., Xu, Z., Xia, Q., Ren, J., Xia, F.: Task allocation and migration algorithm for temperature-constrained real-time multi-core systems. In Proceedings of 2010 IEEE/ACM International Conference on Green Computing and Communications, pp.189–196, IEEE (2010)

56. Carpenter, J., Funk, S., Holman, P., Srinivasan, A., Anderson, J., Baruah, S.: A Categorization of Real-time Multiprocessor Scheduling Problems and Algorithms. In Handbook on Scheduling Algorithms, Methods, and Models, pp. 30.1– 30.19 (2006)
57. Andersson, B.: Global Static-Priority Preemptive Multiprocessor Scheduling with Utilization Bound 38%. In Proceedings of ACM International Conference on Principles of Distributed Systems (OPODIS), vol. 5401; pp.73–88, ACM (2008)
58. Andersson, B., Baruah, S., Jonsson, J.: Static-Priority Scheduling on Multiprocessors. In Proceedings of 2nd Real-Time Systems Symposium, pp.193 – 202, IEEE (2001)
59. Kato, S., Yamasaki, N.: Semi-partitioned fixed-priority scheduling on multiprocessors. In Proceedings of 15th Real-Time and Embedded Technology and Applications Symposium, pp.23 – 32, IEEE (2009)
60. Andersson, B., Bletsas, K, Baruah, S.: Scheduling Arbitrary Deadline Sporadic Task Systems on Multiprocessors. In Proceedings of Real-Time Systems Symposium, pp. 385 – 394, IEEE (2008)
61. Lundberg, L.: Analyzing fixed-priority global multiprocessor scheduling. In Proceedings of 8th Real-Time and Embedded Technology Symposium, pp.145–153, IEEE (2002)
62. Fisher, N., J.-J. Chen, Wang, S. Thiele, L.: Thermal aware global real-time scheduling on multicore systems. In Proceedings of Real-Time and Embedded Technology and Applications Symposium, pp. 131–140, IEEE (2009)
63. Wang, S., Bettati, R.: Delay analysis in temperature constrained hard real-time systems with general task arrivals. In Proceedings of 27th IEEE International Real-Time Systems Symposium, pp. 323–334, IEEE (2006)
64. Baker, T.P.: An analysis of EDF schedulability on a multiprocessor. vol. 18(8); pp. 760 – 768, IEEE (2005)
65. Baruah, S.K, Shun-Shii Lin: Pfair scheduling of generalized pinwheel task systems. Transactions on Computers, vol.47 (7), pp. 812– 816, IEEE (1998)
66. Levin, G., Funk, S., Sadowski, C., Pye, I., Brandt, S.: DP-fair: A simple model for understanding optimal multiprocessor scheduling. In Proceedings of 22nd EUROMICRO Conference, pp. 313, IEEE (2010)

67. Ming Fan, Qiushi Han, Gang Quan, Shangping Ren: Multi-core partitioned scheduling for fixed-priority periodic real-time tasks with enhanced RBound. In Proceedings of 15th International Symposium on Quality Electronic Design, pp.284 – 291, IEEE (2014)
68. Ming Fan, Qiushi Han, Shuo Liu, Shaolei Ren, Gang Quan, Shangping Ren: Enhanced fixed-priority real-time scheduling on multi-core platforms by exploiting task period relationship. pp.85– 96, Elsevier (2014)
69. Liu, J.W.S.: Real-time systems. Prentice Hall (2000)
70. Andersson, B., Jonsson, J.: The utilization bounds of partitioned and pfair static priority scheduling on multiprocessors are 50%. In Proceedings of 15th EUROMICRO Conference on Real-time Systems, pp.33 – 40, IEEE (2003)
71. Guan, N., Martin Stigge, Wang Yi, Ge Yu: Parametric Utilization Bounds for Fixed-Priority Multiprocessor Scheduling. In Proceedings of 26th International Parallel and Distributed Processing Symposium, pp.261-272, IEEE (2012)
72. Anderson, J.H., Bud, V., Devi, U.C.: An EDF-Based Scheduling Algorithm for Multiprocessor Soft Real-Time Systems. In Proceedings of EUROMICRO Conference on Real-Time Systems (ECRTS), pp.199– 208, IEEE (2005)
73. Bastoni, A., Brandenburg, B.B, Anderson, J.H.: Is Semi-partitioned scheduling practical? In Proceedings of 23rd Conference on Real-Time Systems, pp. 125 – 135, IEEE, (2011)
74. Kato, S., Yamasaki, N.: Semi-partitioned fixed-priority scheduling on multiprocessors. In Proceedings of 15th Real-Time and Embedded Technology and Applications Symposium, pp. 23 – 32, IEEE (2009)
75. Lakshmanan, K., Rajkumar, R., Lehoczky, J.P.: Partitioned fixed priority preemptive scheduling for multi-core processors. In Proceedings of 21st EUROMICRO Conference on Real-Time Systems, pp. 239 –248, IEEE (2009)
76. Guan, N., Stigge, M., Yi, W., Yu G.: Fixed-Priority Multiprocessor Scheduling with Liu and Layland's Utilization Bound. In Proceedings of IEEE Real Time and Embedded Technology and Applications Symposium (RTAS), pp. 165 – 174, IEEE (2010)

77. Guan, N., Martin Stigge, Wang Yi, Ge Yu: Fixed-Priority Multiprocessor Scheduling: Beyond Liu and Layland's Utilization Bound. In Proceedings of WiP Real-Time Systems Symposium (RTSS), pp. 1594– 1601, IEEE, (2010)
78. Zhang, Y., Guan, N., Yi. W.: Towards the Implementation and Evaluation of Semi-Partitioned Multi-Core Scheduling. In Bringing Theory to Practice: Predictability and Performance in Embedded Systems. vol. 18; pp. 42– 46, In Open Access Series in Informatics (2011)
79. Young-Si Hwang, Ki-Seok Chung: Dynamic Power Management Technique for Multicore Based Embedded Mobile Devices. IEEE Transactions on Industrial Informatics, vol. 9(3), pp. 1601 – 1612 (2013)