MEMS-based Runtime Idle Energy Minimization for Bursty Workloads in Heterogeneous Many-Core Systems

Ali Aalsaud[1,2], Haider Alrudainy[3], Rishad Shafik[1], Fei Xia[1] and Alex Yakovlev[1]
[1] School of EEE, University of Newcastle, Newcastle upon Tyne, NE1 7RU, England, UK
[2] School of Engineering, Al-Mustansiriya University, Baghdad, Iraq
[3] Basra Engineering Technical College, Southern Technical University, Iraq
Email[1]: { A.m.m.aalsaud, Rishad.Shafik, Fei.Xia, Alex.yakovlev }@ncl.ac.uk. Email[3]: h.m.a.alrudainy@stu.edu.iq

***Abstract*—Heterogeneous many-core systems are increasingly being employed in modern embedded applications for high throughput at low energy cost considerations. These applications exhibit bursty workloads that provide with opportunities to minimize system energy. Traditionally, CMOS-based power gating circuitry, consisting of sleep transistors, is used for idle energy reduction in such applications. However, these transistors contribute high leakage current when driving large capacitive loads, making effective energy minimization challenging.**

In this paper, we propose a novel MEMS-based runtime energy minimization approach. Core to our approach is an integrated sleep mode management based on the performance-energy states and bursty workloads indicated by the performance counters. For effective energy minimization we use a systematic optimization of the controller design parameters by adopting finite element analysis (FEA) in multiphysics COMSOL tool. A number of PARSEC benchmark applications are used as case studies of bursty workloads, including CPU- and memory-intensive ones. These applications are exercised on an Exynos 5422 heterogeneous many-core platform showing up to 50$\%$ energy savings when compared with ondemand governor. Furthermore, we provide all extensive trade-off analysis to demonstrate the comparative advantages of MEMS-based controller, including zero-leakage current and non-invasive implementations suitable for commercial off-the-shelf systems.

I. Introduction

The impetus of high throughput at low energy cost is at the core of design and implementation of many-core embedded systems. To manage the trade-offs between throughput and energy an effective technique is to allocate heterogeneous computing resources on these systems. Exynos 5422 big.LITTLE octa-core platform, which includes 4 big (ARM A15), and 4 LITTLE (ARM A7) cores, is a typical example [1].

Over the years significant research has been carried out to address energy minimization in heterogeneous embedded systems [2]. Such works typically control the core allocation, coupled with dynamic voltage/frequency scaling (DVFS) decisions to react to workload variations [3]. When higher workload is encountered more number of cores are allocated with suitably determined DVFS. Conversely, when the workload is lower, fewer cores are executed with reduced voltage/frequency levels. These allocation are managed by a runtime system that interact with the application for workload-based optimizations.

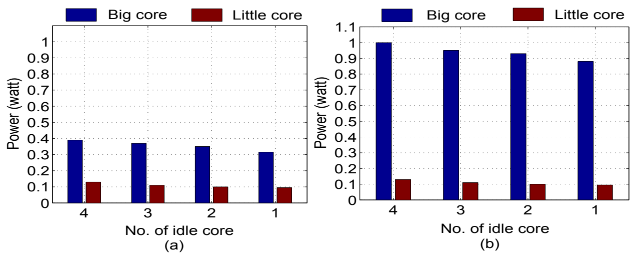


Figure 1: Experimental measurements of idle power by adopting Odroid-XU3 big.LITTLE platform (a) 1400MHz big.LITTLE; (b) 2000MHz big, 1400MHz LITTLE.

From a core-level viewpoint, continuous runtime controls render bursty workloads, which is characterized by frequent switching between high activity followed by no activity. The period of inactivity leads to idle energy consumption as the clock and supply voltage remain operational. Figure1 depicts the idle power measurements on the Odroid-XU3 big.LITTLE platform for different core allocations and frequencies. The following two observations can be made. Firstly, with increasing number of inactive cores (big or LITTLE) the idle power consumption increases. As an example, the idle power of 4 big inactive cores at 2000 MHz is 1 Watt, which drops to 0.8 Watt when only 1 big core is inactive. Secondly, the idle power is also dependant on the operating frequency. For instance, when parallel threads are allocated to LITTLE cores only, the idle power dissipation of 4 big inactive cores rises from 0.39 Watt at 1400 MHz to 1 Watt at 2000 MHz [4].

Idle power contributes to unuseful energy consumption, essentially reducing the battery operational lifetime. To reduce the idle power, the traditional approach is to use power gating. The basic principle is to adopt a number of sleep transistors to disconnect the supply voltage rail for shutting down the inactive cores. Table I summarizes contributions of the existing power gating approaches. A hardware-based stateless load balancing for homogeneous multi-core scheme is evaluated in terms of power and thermal behaviour in [5]. In this approach, a power reduction is achieved by switching off the idle cores. In [6], a sub-clock power gating technique is proposed to reduce static power during the sub-clock cycle of ARM Cortex-M0. This technique uses intrusive redesigning of the power gating paradigm.

Among others, Charles et al. [7] implemented per core power gating in mainstream homogeneous processor (Intel Core i7). They showed that extra power headroom from power gating idle cores can be diverted to the active cores to increase their voltage and frequency without violating the power and thermal envelop. Similarly, diverting the saved power of idle cores into active cores was investigated in [8] by adopting a homogeneous many-core AMD Opteron 6168 processor. The experimental results of this paper are based on manually tuned dynamic voltage scaling (DVS) coupled with power gating.

Minimizing idle power using the existing CMOS-based approaches (see Table I) still remains largely challenging. With increasing capacitive loads, consisting of many cores, the gate dimensions of sleep transistors are becoming wider, posing challenges on device geometry for effective leakage power minimization. MEMS-based power gating solution is attractive for the following two reasons: firstly, MEMS-based controller itself contributes zero-leakage current unlike CMOS-based power gating. Secondly, such controller can be integrated by using back-end metallization layers with no penalty to the overall die area leading to low-cost commercial off-the-shelf implementation [9] , a MEMS-based approach has been demonstrated in , highlighting simulation results that show potential energy reduction benefits over CMOS counterpart (for off-periods $>$ 1 ms). Further, others illustrated methods of forming electromechanical power switch on top of IC device for controlling idle energy consumption of CPU/GPU, I/O interface, and memory controller [10].

Despite its promises, the full-scale implementation for MEMS-based power gating remains unresolved due to engineering challenges such as parametric optimization and interaction with hardware/software platforms. In this paper, we propose a novel MEMS-based runtime and non-invasive idle energy controller for bursty workloads exercised on Odroid-XU3 heterogeneous platform. In our proposed approach, we make the following main *contributions*:

- propose a MEMS-based non-invasive runtime power gating controller to support bursty workloads,
- core to the controller is an integrated sleep mode management based on the performance-energy states modelled using the performance counters feedback,
- show a novel systematic optimization of MEMS parameters using finite element analysis (FEA) in multiphysics COMSOL tool, and
- validate using a number of real application benchmarks to demonstrate the comparative advantages and trade-offs of our controller.

To the best of our knowledge, this is the first work that investigates (a) a systematic optimization of MEMS-based relay through parametric sweep in COMSOL tool, and (b) runtime power gating control using non-invasive MEMS-based solution for heterogeneous many-core systems. The rest of this paper is organized as follows: Section II shows the background of MEMS devices. Section III shows modelling approach of MEMS, and energy-latency optimization by using finite element analysis (FEA) method. The proposed system approach is described in Section IV. Experimental results are presented in Section V. Section VI concludes the paper.

Table I: Features and limitations of the existing approaches.

Approach	Architecture	Validation	Design abstraction	Key method
[5]	Homogeneous	Hardware	System	Task mapping, power gating
[6]	Homogeneous	Hardware	Micro-architecture	Power gating, ARM Cortex-M0
[7]	Homogeneous	Hardware	System	Power gating, (Nehalem)
[8]	Homogeneous	Hardware	System	Power gating, AMD Opt. 6168, DVS (manually)
Proposed	Heterogeneous	Hardware+ simulation	System	MEMS power gating+DVFS

II. Background of MEM relays

Relays can be classified based on the method of actuation into electrostatic, electrothermal, magnetostatic, and piezoelectric. However, they could also be classified either according to the axis of deflection (lateral, vertical) or to the contact interface (ohmic, capacitive). Based on the actuation method, each relay has different characteristics of bias voltage, bias current, on-resistance, delay time, current handling, and endurance, as illustrated in [11].

Among these relays, micro electrostatic actuated switch (MEMS) has recently received considerable attention for digital logic applications due to its low active power consumption, scalability, and ease of manufacture using conventional planar processing techniques. As a consequence, numerous implementations of MEM switches have recently been proposed that indicate potential of an order of magnitude power savings than CMOS in low frequency applications [12].

Their principle of operation, in general, can be summarised as in Figure 2 (a)-(b): when the gate-body voltage increases above the "pull-in voltage" ($|V_{gb}| \geq V_{pi}$), a contact dimple touches the source and drain terminal, causing the current to flow. The electrical contact is broken when the gate-body voltage decreases below the "pull-out voltage" ($|V_{gb}| \leq V_{po}$).

The number of on/off switch cycles of MEM switch without demonstrating any operating failure is reported in [12] [11] to be $1{\times}10^9$ and $2.1{\times}10^9$, respectively. As an example, if these relays switch once/second then, they roughly can last for 67 years without experiencing any failure in operation. Consequently, MEM switches can be an attractive candidate for runtime power gating of bursty behavioural systems.

Experimental results of fabricated MEMS revealed that contact resistance of theses switches ranges from 20Ω to 1KΩ as reported in [12] [11], respectively. This means, fewer MEMS than CMOS counterparts are required in the power switch network (PSN) to mitigate any performance degradation due to voltage drop.

III. MEM relay modelling

Finite element analysis (FEA) is a numerical analysis method used to solve large numbers of partial differential equations (PDEs) for any design. This method is capable of

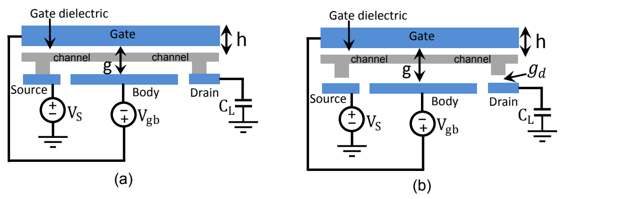


Figure 2: Cross-section in the (a) on state ($V_{gb} \geq V_{pi}$); (b) off-state ($V_{gb} < V_{pi}$) [12].

handling multiphysics phenomena and accurately simulating static and dynamic behaviour. To model and capture the physical behavior of MEMS accurately, COMSOL multiphysics tool has been used in our work. Figure 3(a) shows the simulated pull-in voltage by using FEA, while Figure 3(b) depicts the adopted MEMS in our analysis.

An extensive parametric sweep simulation is performed to estimate the range of electo-mechanical parameters, as shown in Table II, thereby energy-latency tradeoff the MEMS.

In order to optimizing a precise analytical formula of pull-in voltage, which is used in Section (IV-B), sensitivity analysis coupled with parametric sweep have been performed, as shown in Algorithm 2. As a result, our analytical model of evaluating pull-in voltage at various gaps demonstrates a close fit to the one obtained from FEA, as shown in Figure 4(a). The following section describes how to evaluate energy-latency trade-offs:

Algorithm 1 Pull-in analysis based FEA parametric sweep.

```
Define: Spring width:=W, Spring length:=L, Actuation gap:=g.
Define: Constant: Actuation area(A), Gate thickness(h), Dimple
    gap:=g_d.
Output: $(V_{pi})$
1: Parametric sweep L = $5\times10^{-6}$:$10^{-6}$:$5\times10^{-5}$.
2: Set    W=$5\times10^{-6}$.
3: Calculate $\frac{\partial V_{pi}}{\partial g}$, $\frac{\partial V_{pi}}{\partial W}$, $\frac{\partial V_{pi}}{\partial L}$, $\frac{\partial V_{pi}}{\partial g}$ (Sensitivity analysis)
4: $V_{pi} \simeq \sqrt[2]{\frac{\beta \times Lg^3}{\varepsilon_0 WA}}$; $\beta = 3.87 \times 10^{-4}$.
```

A. Structural Stiffness

The structural stiffness of MEM relays subjected to an electrostatic force is modelled using FEA. In this paper, It is assumed that MEMS exhibit a linear elastic deformation. To solve coupled problems with complex geometry, Arbitrary Lagrangian-Eulerian (ALE) was used by the COMSOL tool to obtain the equilibrium point between electrostatic force and mechanical structure. This method diverges as the MEMS displacement approaches the pull-in point. This is attributed to the fact that this is the last point where behind it MEMS collapses non-linearly. At this point, electrostatic force equals to the spring restoring force. Having calculated the pull-in voltage (V_{pi}) and correspond displacement (Z) by the COMSOL tool, the structural spring constant can be calculated as follows:

$$F_{ele.} \mid_{pullin} = F_{spring} \Longrightarrow k \mid_{structure} = \frac{V_{pi}^2}{2Z} \frac{\partial C(Z)}{\partial Z} \quad . \quad (1)$$

B. Energy-Latency Analysis of MEMS

Results in figure 4(b) demonstrate switching energy of MEMS by using FEA as a function of gap distance ratio ($\frac{g_d}{g}$), and resonant frequency (w_0). As it can be seen, increasing ($\frac{g_d}{g}$) causes an almost linear increase of switching energy at low (w_0). Alternatively, switching energy increases exponentially with increasing resonant frequency (w_0), by sweeping the ratio of ($\frac{L}{W}$), at high ($\frac{g_d}{g}$). Figure 5(a-b) shows simulation results of mechanical delay time as a function of gap ratio ($\frac{g_d}{g}$), resonant frequency (w_0), and quality factor (Q). One observation can be made that T_{mech} is inversely proportional with (w_0), and it is linearly proportional with the increase in ($\frac{g_d}{g}$), which is consistent with the theoretical predictive equation in [12]. These results clearly indicate the trade-off between switching energy and mechanical delay time of MEM relays. As an example, it is found that at ($\frac{g_d}{g}$)=0.6 every ∼3.3× increases in switching energy can be traded-off for a ∼2.5× reduction in the MEMS delay.

IV. Proposed Approach

Using the optimized relay design (Section III) a MEMS-based runtime power gating controller is proposed. Figure 6 shows our proposed MEMS-based power controller coupled with Exynos 5422, used as a case-study of heterogeneous system. As can be seen, our proposed approach interacts with runtime performance-energy state management to suitably identify opportunities for switching the idle big cores off under bursty workload scenarios. This is enabled through a charge pump connected to the power switch network (PSN) based-on MEMS. In the following sections we briefly describe our approach, highlighting the platform and runtime interactions.

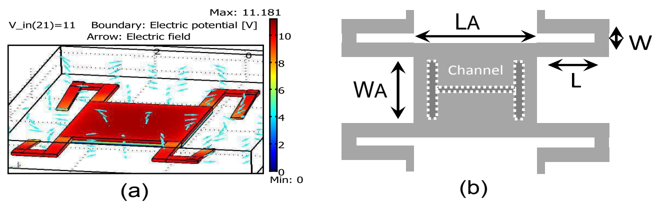


Figure 3: (a) FEA-simulated pull-in voltage and displacement; (b) simplified sketch, symbols L, W, L_A, W_A and h denote, respectively, spring length/width, actuation area length/width, and thickness of the suspended gate [12].

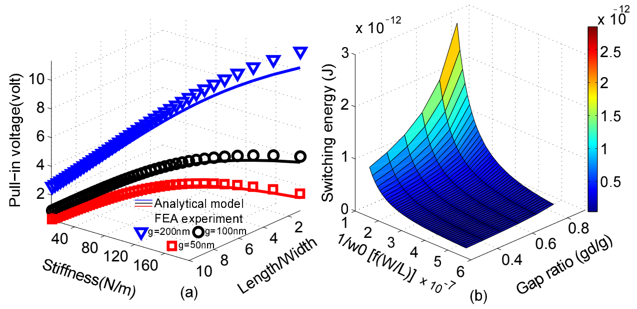


Figure 4: (a) A comparison of the pull-in voltage for three different gaps obtained from full finite element model and the analytical model; (b) Switching energy at Q=1 based FEA of g_0=100nm and A=450um^2 as a function of g_d and resonant frequency.

Table II: Current MEM relay physical parameters based on COMSOL multiphysics tool.

MEMS Area (um^2)	Pull-in voltage (volt)	Switching energy (pJ)	Mechanical delay (us)	Stiffness (N/m)	Mass (pg)	Viscous damping (uN.s/m)	Actuation gap (nm)	Actuation Capacitance (fF)
450	2.6-11.3	0.1-3.2	0.15-1	10.14-192.6	1.1-2.9	50	100	60-200

A. *Hardware Platform and Performance Counters*

The Odroid-XU3 system on chip (SoC) platform is chosen as a case study in this work, Figure 6(a). The platform consists of a 28 nm application processor Exynos 5422, featuring a high-performance Cortex-A15 quad-core processor block, a low-power Cortex-A7 quad-core block, a Mali-T628 GPU, and a 2GB DRAM LPDDR3. The platform also contains power and temperature sensors for different CPU, GPU, and memory blocks. It supports multiple power domains and can facilitate operating with a number of pre-set supply voltages and operational frequency values ranging from 200MHz to 2000MHz in 100MHz step. Additionally, it has system software-supported core disabling and affinity control features, typically used for energy-efficiency [1].

To enable the monitoring of energy-performance states, we designed a custom system software routine following ARM's technical specification manual that can report different performance counter values at pre-defined regular intervals. The routine can be used as a wrapper around the application binaries. This routine together with its libraries is currently being considered for a public release.

B. *Energy-Performance State Models*

Figure 7 depicts the power consumption of the *ferret* application used to study the impact of different thread to core allocations and operating frequencies. The power measurements were obtained through our performance counter routine (Section IV-A). As expected the power consumption increases as the operating frequency is increased from 200MHz to 1400MHz, and as more cores are allocated for the given application. Figure 8 shows the power consumption and execution time when the system is operating at the maximum frequency. The apparent power saturation is caused by the system engaging in automatic thermal throttling.

The relationship between energy consumption (Figure 7), types of cores (big, LITTLE), frequency, and number of cores of the Odroid-XU3 platform can be theoretically rendered as [13]

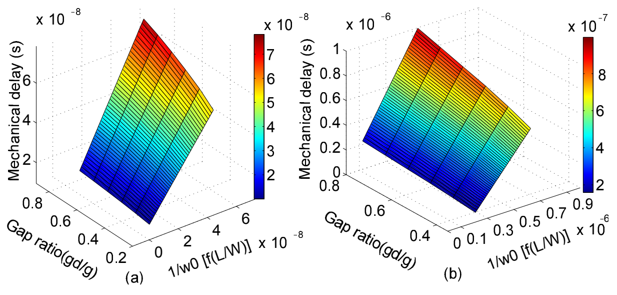
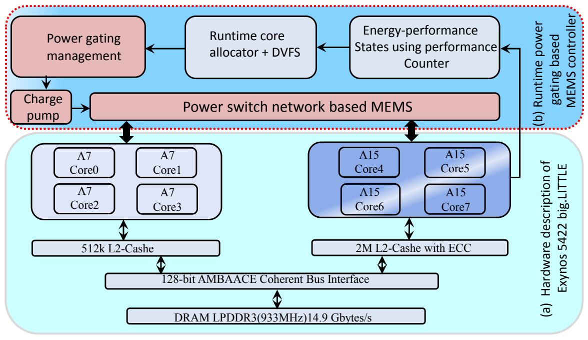


Figure 5: (a) T$_{mech}$ as a function of gap ratio and resonant frequency obtained from FEA at Q=5; (b) T$_{mech}$ as a function of gap ratio and resonant frequency obtained from FEA at Q=1.



Power gating management
Runtime core allocator + DVFS
Energy-performance States using performance Counter
Charge pump
Power switch network based MEMS
(b) Runtime power gating based MEMS controller
A7 Core0
A7 Core1
A7 Core2
A7 Core3
A15 Core4
A15 Core5
A15 Core6
A15 Core7
512k L2-Cashe
2M L2-Cashe with ECC
128-bit AMBAACE Coherent Bus Interface
DRAM LPDDR3(933MHz)14.9 Gbytes/s
(a) Hardware description of Exynos 5422 big.LITTLE

Figure 6: (a) Exynos 5422 block diagram; (b) proposed runtime power gating.

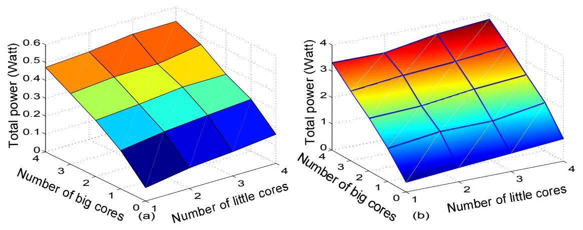


Figure 7: Total power for *ferret* application at (a) 200 MHz (b)1400 MHz.

$$E(V,f) = \frac{N_{A7}I_{A7}V_{A7}}{f_{A7}} + \frac{N_{A15}I_{A15}V_{A15}}{f_{A15}} + \varepsilon_1(x) \quad , \quad (2)$$

where N_{A7} and N_{A15} is the number of A7 and A15 cores, V_{A7} and V_{A7} are the voltages of A_7 and A_{15} cores, I_{A7} and I_{A7} are the currents of A_7 and A_{15} cores, respectively, $\varepsilon_1(x)$ represents the background energy due to leakage, interconnects and memory access. Eq. (2) can be used to model energy consumption for all applications, with high accuracy up to 5% error rate. The detailed modelling results can be found in [3].

To enable power gating in Odroid-XU3, the PSN also adds energy consumption due to charging and discharging transition. If C_R and C_M are the total capacitance, which is charged or discharged during the transition of MEMS and CMOS respectively, then the switching energy of this transition can be theoretically evaluated as:

$$E_S(R) = U_R C_R V_{pi}^2 \simeq U_R \frac{\epsilon A}{g-z} V_{pi}^2 \simeq U_R \frac{\beta \times Lg^3}{W(g-z)} \quad , \quad (3)$$

$$E_S(M) = U_M W_M C_M V_g^2 \quad , \quad (4)$$

where U_R and U_M is the number of parallel MEMS and CMOS power switch, respectively. W_M represents the width of sleep transistor. For a given amount of time that cores A_{15} and/or A_7 are in active or sleep mode, the energy per power gate switching cycle is:

$$E_R(V,f) = \frac{N_{A7}I_{A7}V_{A7}}{f_{A7}} + \frac{N_{A15}I_{A15}V_{A15}}{f_{A15}} + E_S(R) \quad , \quad (5)$$

$$E_M(V,f) = \frac{N_{A7}I_{A7}V_{A7}}{f_{A7}} + \frac{N_{A15}I_{A15}V_{A15}}{f_{A15}} + E_S(M) + \varepsilon_3 \quad (6)$$

where ε_3 represents energy consumption due to leakage current of sleep transistors. Eqs. (5) and (6) will be used to evaluate energy overhead caused by power gating.

C. *Runtime Cores Allocation and DVFS*

The proposed runtime core allocation and DVFS control is implemented by Algorithm 2. This algorithm determines the workload type, the core allocation and DVFS for an application. This is based on comparing monitored performance counter values to pre-determined thresholds obtained from off-line characterization experiments at design time. Based on experience from [3] [14] , we classify workloads by their demands on processing (CPU) and communication (memory). Workloads are therefore divided into three types: CPU-intensive (CI), memory-intensive (MI)and mix of memory-intensive and CPU-intensive (MIX).To classify the workloads we propose the following equation:

$$IPC_{normalized} = K * (IR - memory\ access)/cycle \quad (7)$$

where IR is instruction retired, cycle is unhalted CPU cycles, and K is (1/IPC_{max}). IPC_{max} can be obtained from manufacturer literature. The algorithm first collects the performance counter values (instruction retired, memory access, unhalted CPU cycles) and compute $IPC_{normalized}$. This value is compared with the higher IPCN threshold. An application is CI if it stresses the CPUs to high IPCN. If the IPCN reading is lower than the higher IPCN threshold but higher than the lower IPCN threshold, the application is of the type MIX. If the IPCN reading is lower than the lower IPCN threshold the application is MI. For CPU-intensive applications increased number of parallel big cores with high frequency is used. Conversely, for memory-intensive applications increased number of parallel LITTLE cores is used with high frequency. For applications with combinations of the two, both LITTLE and big cores are allocated. Typical values for $IPCN_H$ and $IPCN_L$ are 0.32, and 0.22 respectively.

These values and this method is based on extensive experiments from running a large number of established and synthetic benchmarks including those from the Parsec-3.0 suite

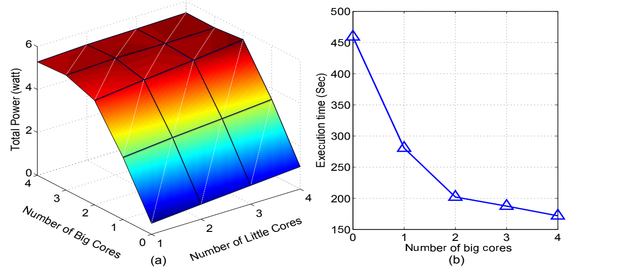


Figure 8: (a) Measured power of *ferret* application at 2000 MHz big-cores and 1400 MHz LITTLE-cores; (b) Execution time when 4 LITTLE cores fully operated with various big cores number.

on the Odroid XU-3. Runtime controls using Algorithm II render bursty workloads, and hence opportunities to power gate the idle cores.

Algorithm 2 Runtime cores allocation and DVFS.

```
Input: Effective performance counter values (instruction retired,memory access,unhalted CPU cycles);
Constant: Parameters $IPCN_H, IPCN_L$;
Output: WL_type, allocated_cores and DVFS
Compute: $IPC_{normalized}$
 1: If: $IPC_{normalized} >= IPCN_H$;
 2:     WL_type = CI ;                          ▷ CPU-intensive
 3:     Allocated_cores big cores only;
 4:     DVFS $f_{A15}$=Max.;
 5: Else if: $IPC_{normalized} <= IPCN_L$;
 6:     WL_type = MI;                           ▷ Memory-intensive
 7:     Allocated_cores little cores only;
 8:     DVFS $f_{A7}$=Max.;
 9: Else: WL_type = MIXED;                      ▷ Combination
10:     Allocated_cores ;
11:     DVFS $f_{A7}$=Max., $f_{A15}$=Max.
12: End if
```

D. *Power-Gating Management*

Based on the opportunities exposed by runtime control (Section IV-C) power gating of cores is enabled by the interface shown in figure. 9. As can be seen, on every interval a number of flag registers are overwritten by the system software depending on the number of idle big cores. As an example, when two big cores (core 6 and core 7) are free the corresponding flag bits are set to 1 indicating the opportunity of power gating. These bits are then used to enable the charge pumps, which are used for shutting those cores.

V. Results

A number of experiments are carried out in emulated environment in COMSOL tool, which are further cross-validated on Exynos 5422 platform. The evaluation setup is first explained highlighting this environment, followed by extensive application case studies and trade-offs analysis.

A. *Evaluation Setup*

1) Energy Measurement of Power Gating Circuitry: Figure 10 shows the emulation environments used to evaluate both CMOS- and MEMS-based power gating circuitries. As can be seen, both setup consist of PSN coupled with the heterogeneous cores (only core A15 is shown for demonstration purpose). The CMOS-based emulation environment has been

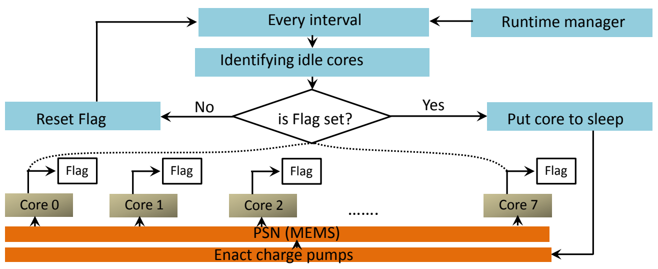


Figure 9: Hardware flowchart of the power gating management interface.

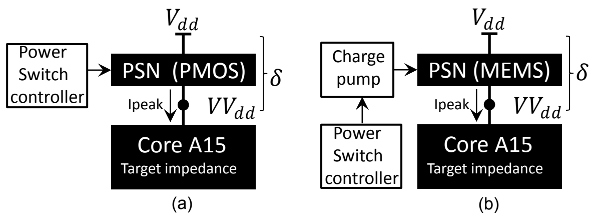


Figure 10: Power gating circuitry based: (a) PMOS transistors; (b) MEM relays.

developed using Cadence Spice tool, while that of MEMS-based environment has been developed using COMSOL multiphysics tool. A key aspect for effective emulation is to determine the target impedance (Z_{target}) of the active cores. Establishing target impedance of the active core, that should be met over a broad of frequency band, can be computed by assuming a 5% allowable ripple in the core virtual voltage (VV_{dd}), and a 50% drawn current in the rise and fall time of the processor clock [15].

$$Z_{target} = \frac{0.1 \times VV_{dd}}{I_{peak}} \tag{8}$$

In our experiment, the maximum current drawn by the A15 per core in the case of (for CPU-intensive application) is measured to be I_{peak}=1A at f=2GHz. For other operating frequencies and workload types the rated current ($I < I_{peak}$) can also be accurately estimated. Furthermore, for fair comparison between MEMS and CMOS based power gating circuitry, it is assumed that the allowable voltage drop (δ), as show in Figure 10, is around 0.1V. Therefore the number (U_M), width (W_M), and switching energy E_S(M) of power transistors in the PSN are tuned so that it can deliver the maximum current with the allowable voltage drop (δ). Alternatively, PSN of MEM relays are evaluated as indicated in Table II.

2) Idle Power Measurement of Exynos 5422: Using the setup (Section IV-A) a case study application (*ferret*, part of PARSEC benchmarks) is executed in single cortex A15 (core7). The aim is to demonstrate in details the application state dependences over different times and frequencies. The application execution consists of two key states. State 1 characterizes core idle state at low frequency, while state 2 shows active state when the application is instantiated and exercised, as shown in Figure 11.

B. Application Case Studies

Figure 12 shows the state transition diagram of two different applications resulting from runtime core allocation and DVFS

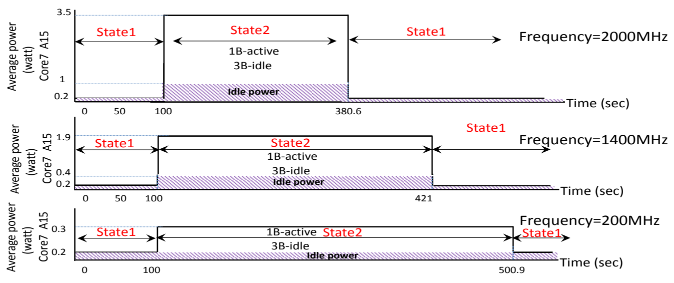


Figure 11: Idle power dissipation of Exynos 5422 big.LITTLE octa-core heterogeneous platform exercising *ferret* application in only one big core.

control in Algorithm II. For demonstration purposes two different applications have been chosen as a case study: Figure 12(a) for CPU-intensive application, and Figure 12(b) for memory-intensive application. As expected the *ferret* application exercises the workloads using mostly the big cores. The state transition in *ferret* starts from state 1 (idle) to s23 (active 2B-1L at 200MHz). These states are then followed by s21 (active 3B-1L at 400MHz) and s22 (active 4B-1L at 400MHz). The application return back upon completion. The corresponding execution time of each state is normalised and annotated in percentage on the transition edge. Since the application remains in s23 most of the time, it gives the opportunity of shutting down the only idle big core, thereby saving higher energy, as can be seen in Table III (a).

In the case of the memory-intensive application the runtime control allocates LITTLE cores at higher frequency. This gives opportunity for our MEMS-based to disable the big cores and achieving energy reduction. For example, since the application execute in s21 state most of the time it benefits from disabling three big cores and to achieve 32% energy reduction.

Figure 13 shows the comparative energy consumptions of 3 different applications: *ferret*, *fluidanimate*, and *bodytrack*. These applications are executed with three different runtime controllers. The first controller is a traditional ondemand governor typically available in modern Linux operating system. The second controller is the runtime core allocator and DVFS shown in Algorithm II without using any power gating. The third is our proposed runtime controller featuring the MEMS-based power gating circuitry (figure7). From the Figure, two key observations can be made. Firstly, the ondemand governor, which is agnostic to core allocation management, only controls operating frequencies based on CPU usage. As such there is no power gating opportunity of the cores, resulting in high dynamic and leakage energy consumption. The runtime controller (Algorithm II) allocates the number of cores and DVFS based on **power normalized performance**. However, due to no power gating, an effective energy minimization is limited. Our work integrates MEMS-based energy reduction and achieved upto 20% less energy consumption on top of 43% savings achieve by the runtime controller. The second observation is related to power gating opportunities exposed by different applications. As can be seen the best energy savings (50%) is achieved by memory-intensive applications. This is because these application favor allocation of LITTLE cores, and hence generate along bursts of idle periods for big cores.

Using Eqs. (5) and (6), the normalized energy overhead caused by PMOS power gating circuitry over that of MEMS has been evaluated as shown in Figure 14. Two further observations can be made. Firstly, increased the number of power gated cores causes a reduction in energy savings of MEMS due to the high switching energy of MEMS compared with that of CMOS. Secondly, increasing the core execution frequency leads to improve energy savings of MEMS power gating .

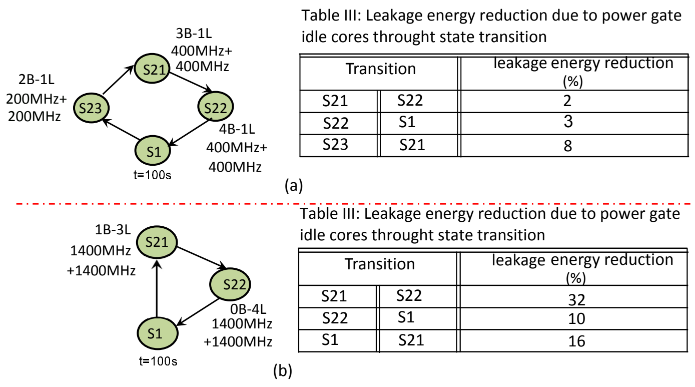


Table III: Leakage energy reduction due to power gate idle cores throught state transition

Transition		leakage energy reduction (%)
S21	S22	2
S22	S1	3
S23	S21	8

Table III: Leakage energy reduction due to power gate idle cores throught state transition

Transition		leakage energy reduction (%)
S21	S22	32
S22	S1	10
S1	S21	16

Figure 12: Test bench of state transition based (a) CPU-intensive *ferret* application; (b) Memory-intensive *fluidanimate* application.

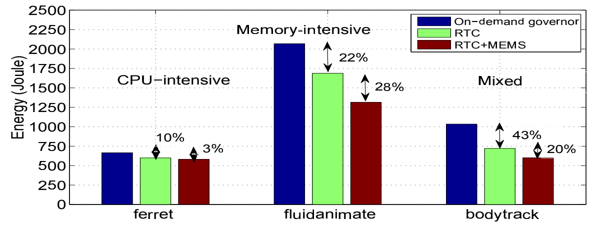


Figure 13: Energy comparison of *ferret*, *fluidanimate*, and *bodytrack* applications.

C. Trade-off Analysis

The energy savings in our MEMS-based approach is achieved at the cost of latency overheads. The major contributors of these overheads are the wake-up core latency and the charge pumps enact. Table III shows the energy-latency overhead of CMOS- and MEMS-based switch. However, the actual impact of these overheads will depend on the nature of the burst workloads.

Table III: Latency overhead

A15 (core/cluster) wake-up latency	A7 (core/cluster) wake-up latency	Charge pump latency/energy
(600/2230)us	(250/1650)us	0.5us/2pJ

VI. Conclusions

A MEMS-based power gating approach for heterogeneous many-core platforms exercising bursty workloads is proposed. Theoretical analysis gave optimized relay models for a MEMS-based hardware/software interaction at runtime energy minimization. The interaction is aided by workload classification, followed by resource allocation and DVFS. Experiments using real benchmark applications show that the proposed approach effectively reduces energy consumption compared to that of traditional power minimization approach.

We envisage that the proposed controller would be useful for commercial-off-the-shelf components, enabling low-cost integration for holistic energy minimization. Future work includes fabrication and practical validation using different heterogeneous architectures.

VII. Acknowledgement

This work is supported by the EPSRC (project PRiME, grant EP/K034448/1). The first author is also supported by studentship funding from the Ministry of Iraqi Higher Education and Scientific Research.

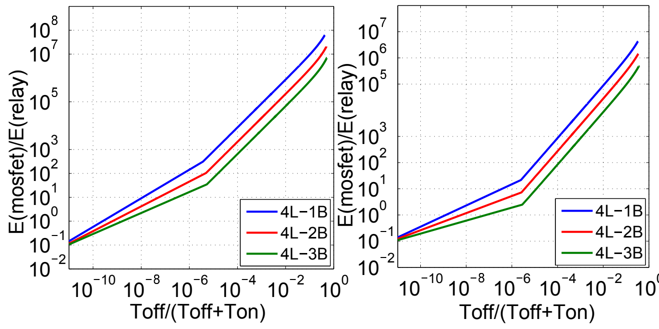


Figure 14: Energy gain vs (1-D) for various core allocations for a design power gated with MOSFETs and MEMS for *ferret* application at (a) 2000MHz; (b) 200MHz.

References

[1] S. Skalicky, S. Lopez, M. Lukowiak, and A. G. Schmidt, "A parallelizing matlab compiler framework and run time for heterogeneous systems," in *HPCC-CSS-ICESS*, pp. 232–237, IEEE, 2015.

[2] S. Yang, R. A. Shafik, G. V. Merrett, E. Stott, J. M. Levine, J. Davis, and B. M. Al-Hashimi, "Adaptive energy minimization of embedded heterogeneous systems using regression-based learning," in *PATMOS, 2015 25th International Workshop on*, pp. 103–110, Sept 2015.

[3] A. Aalsaud, R. Shafik, A. Rafiev, F. Xia, S. Yang, and A. Yakovlev, "Power–aware performance adaptation of concurrent applications in heterogeneous many-core systems," in *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*, pp. 368–373, ACM, 2016.

[4] R. Gensh, A. Aalsaud, A. Rafiev, F. Xia, A. Iliasov, A. Romanovsky, and A. Yakovlev, *Experiments with odroid-xu3 board.* Newcastle University, Computing Science, 2015.

[5] E. Musoll, "Hardware-based load balancing for massive multicore architectures implementing power gating," *TCAD*, vol. 29, no. 3, pp. 493–497, 2010.

[6] J. N. Mistry, B. M. Al-Hashimi, D. Flynn, and S. Hill, "Sub-clock power-gating technique for minimising leakage power during active mode," in *DATE*, pp. 1–6, March 2011.

[7] J. Charles, P. Jassi, N. S. Ananth, A. Sadat, and A. Fedorova, "Evaluation of the intel® core i7 turbo boost feature," in *IISWC*, pp. 188–197, IEEE, 2009.

[8] K. Ma and X. Wang, "Pgcapping: exploiting power gating for power capping and core lifetime balancing in cmps," in *PACT*, pp. 13–22, ACM, 2012.

[9] H. Fariborzi, M. Spencer, V. Karkare, J. Jeon, R. Nathanael, C. Wang, F. Chen, H. Kam, V. Pott, T.-J. K. Liu, *et al.*, "Analysis and demonstration of mem-relay power gating," in *CICC, 2010 IEEE*, pp. 1–4, IEEE, 2010.

[10] K. Mori, Z. Tran, G. T. Dao, and M. E. Ramon, "Method of forming an electromechanical power switch for controlling power to integrated circuit devices and related devices," July 22 2014. US Patent 8,786,130.

[11] V. Pott, H. Kam, R. Nathanael, J. Jeon, E. Alon, and T.-J. K. Liu, "Mechanical computing redux: Relays for integrated circuit applications," *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2076–2094, 2010.

[12] M. Spencer, F. Chen, C. C. Wang, R. Nathanael, H. Fariborzi, A. Gupta, H. Kam, V. Pott, J. Jeon, T.-J. K. Liu, *et al.*, "Demonstration of integrated micro-electro-mechanical relay circuits for vlsi applications," *JSSC*, vol. 46, no. 1, pp. 308–320, 2011.

[13] V. Petrucci, O. Loques, and D. Mossé, "Lucky scheduling for energy-efficient heterogeneous multi-core systems," in *HotPower*, (Berkeley, CA), USENIX, 2012.

[14] F. Xia, A. Rafiev, A. Aalsaud, M. Al-Hayanni, J. Davis, J. Levine, A. Mokhov, A. Romanovsky, R. Shafik, A. Yakovlev, *et al.*, "Voltage, throughput, power, reliability, and multicore scaling," *Computer*, vol. 50, no. 8, pp. 34–45, 2017.

[15] B. Amelifard *et al.*, "Optimal selection of voltage regulator modules in a power delivery network," in *DAC*, pp. 168–173, IEEE, 2007.