Open access • Journal Article • DOI:10.1504/IJCAET.2014.065419

A survey of techniques for improving energy efficiency in embedded computing systems — Source link ↗

Sparsh Mittal

Institutions: Iowa State University

Related papers:

- The Case for Energy-Proportional Computing

- A Survey of Architectural Techniques For Improving Cache Power Efficiency

- A Survey of Techniques For Improving Energy Efficiency in Embedded Computing Systems

- Dynamic voltage and frequency scaling: the laws of diminishing returns

- A survey on techniques for improving the energy efficiency of large-scale distributed systems

A survey of techniques for improving energy efficiency in embedded computing systems

Sparsh Mittal

A Survey of Techniques For Improving Energy Efficiency in Embedded Computing Systems

Sparsh Mittal

Future Technologies Group
Oak Ridge National Laboratory (ORNL)
Oak Rdge, TN, USA
Email: sparsh0mittal@gmail.com

Abstract: Recent technological advances have greatly improved the performance and features of embedded systems. With the number of just mobile devices now reaching nearly equal to the population of earth, embedded systems have truly become ubiquitous. These trends, however, have also made the task of managing their power consumption extremely challenging. In recent years, several techniques have been proposed to address this issue. In this paper, we survey the techniques for managing power consumption of embedded systems. We discuss the need of power management and provide a classification of the techniques on several important parameters to highlight their similarities and differences. This paper is intended to help the researchers and application-developers in gaining insights into the working of power management techniques and designing even more efficient high-performance embedded systems of tomorrow.

Biographical notes: Sparsh Mittal received the B.Tech. degree in electronics and communications engineering from IIT, Roorkee, India and the Ph.D. degree in computer engineering from Iowa State University, Ames, IA, USA. He is currently working as a Post-Doctoral Research Associate at Oak Rige National Laboratory (ORNL), TN, USA. In his B.Tech. degree, he was the graduating topper of the batch and his major project was awarded institute silver medal. He was awarded scholarship and fellowship from IIT Roorkee and Iowa State University. His research interests include cache architectures, embedded systems, energy efficiency and multicore systems.

.

1 Introduction

Recent years have witnessed a phenomenal growth in features and applications of embedded systems. Embedded systems such as mobile computing systems now offer integration of video camera, net browser, wireless data modem and phone. Further, it has been estimated that the number of mobile devices has now become almost equal to the population of the world [1]. These trends, however, have also presented significant challenges for managing power consumption of embedded systems. Many portable systems have stringent power budgets, such as 1 or 2W, while ultra low-power embedded systems (e.g. wearable systems) have a power budget of a few milli watts [2]. This is in sharp contrast with general-purpose and graphics processors which have the power budget of up to a few hundred watts [3]. The low power budgets of embedded systems present severe demands for improving

their energy efficiency. As an example, a 3G mobile phone receiver requires nearly 40 GOPS (giga operations per second) to handle a 14.4 Mbps channel and doing this in a power budget of 1W would require energy efficiency of 25pJ per operation [4]. Thus, to continue to scale their performance and ensure reliability, longevity and adoption in wide range of applications, power management has become extremely important for embedded systems.

In this paper, we highlight the need of power management in embedded systems and survey several research works which are aimed at improving energy efficiency of embedded systems. To provide insights into the working of these techniques, we classify them on the basis of their key research idea. We believe that this survey will help the researchers and designers in understanding the state-of-the-art in power management of embedded systems and also motivate them to further improve the energy efficiency of embedded systems.

In a paper of this length, it is not possible to do justice to the broad range of developments in the field of embedded systems and hence, we take the following

approach to limit the scope of the paper. We include only those research works that propose methods for improving energy efficiency and also evaluate it. Those works which only evaluate performance improvement are not included although they may also lead to better energy efficiency. We review application and architectural level techniques and not circuit-level techniques. Since different techniques have been evaluated using different platforms and methodologies, we only focus on their fundamental research idea and do not present the qualitative results.

The remainder of this paper is organized as follows. Section 2 provides a background on embedded systems and also highlights the need of power management. Section 3 provides an overview and classification of power management techniques in embedded systems. Section 4 discusses some of these techniques in detail. Finally, Section 5 provides concluding remarks and also discusses the future challenges.

2 Background

An embedded system is a computing system which is designed for specific control functions and is embedded as part of the complete device which may include hardware and mechanical parts. Thus, in contrast with general-purpose computers (e.g. desktop), an embedded system performs a few pre-defined tasks, with very specific requirements. Typical examples of embedded systems include MP3 players, smart cameras and cellular phones.

2.1 Sources Of Power Consumption

We briefly review the sources of power consumption in embedded systems and refer the reader to previous work [5, 6] for more details. The power consumption of embedded systems can be broadly divided in two categories, namely dynamic power and static power. The dynamic power (P_{dyn}) consumption arises from charging and discharging of the load capacitance, and the short circuit currents. The leakage power (P_{leak}) arises due to leakage currents that flow even when the device is inactive. Thus, we have

$$P_{dyn} = \alpha C V^2 F$$
$$P_{leak} = I_{leak} V$$

Here α shows the switching activity, F shows the operating frequency and V shows the operating voltage. I_{leak} shows the leakage current. With CMOS scaling the leakage power is increasing dramatically [7]. DVFS based techniques work by reducing dynamic energy, while the techniques which transition the system to low-power aim to reduce leakage energy.

2.2 Importance of power management

Power management in embedded systems is important for the following reasons.

2.2.1 Limited Size and Battery

For battery-operated mobile embedded systems, energy supply is a crucial limitation. Power consumption leads to heating, which is unacceptable in several domains such as wearable embedded systems. Further, the small size of these systems also limits the amount of heat-dissipation that can be managed. Smaller power consumption enables use of smaller power supplies and reduced heat-dissipation overhead, which also reduces the cost, weight and area of embedded systems. Thus power management can lead to easier system design.

2.2.2 Ensuring Longevity

A 15 degree Celsius rise in temperature increases the device failure rates by up to a factor of two [8]. Thus, power dissipation has deleterious effect on reliability of embedded systems and this phenomenon may be crucial for medical devices and mission-critical systems.

2.2.3 Addressing Inefficiency Arising due to Over-provisioning of Resources

In embedded systems, idle intervals arise for several reasons, such as pessimistic estimate of worst-case execution time and inherent slack due to relaxed deadline etc. Despite this, the designers need to provision resources to meet the worst-case performance requirement which leads to energy wastage. Thus, dynamic energy saving techniques can use runtime adaption to trade performance for saving energy. Also, since the embedded systems are typically used for well-defined applications, static techniques can be easily used for per-application tuning of resources.

2.2.4 Meeting Performance Requirements

In recent years, embedded processors are used to execute resource-intensive applications (e.g. multimedia processing [9–11]) that were originally designed for general-purpose processors. To meet these performance demands, modern embedded processors use many complex features such as multi-cores, multi-level caches etc. [12–16]. These trends have influenced the design of embedded systems to be optimized for higher performance, instead of lower power consumption.

2.2.5 Power Challenges Posed by CMOS Scaling

The advancements in CMOS technology have greatly increased the on-chip transistor densities and speeds. These trends have led to a technology-imposed *utilization wall* which limits the fraction of the chip that can be simultaneously used at full speed within the

power budget. Thus, today the processor performance is primarily constrained by energy efficiency and it has been estimated that, if left addressed, power challenges may end future performance scaling [17, 18]. Conversely, techniques for improving energy efficiency can enable the designers to scale performance by executing parallel computations without violating the power budget.

2.2.6 Trends in Usage Pattern

In recent years, mobile computing devices have become the key platform for the mobile convergence applications, e.g. web browsing, imaging, and video streaming. Due to these features, embedded systems have become ubiquitous. Thus, while an individual portable system consumes much less power than a server in the data center, the large user-base of embedded systems makes their total power consumption very high.

2.2.7 Enabling Green Computing

It has been estimated that the ICT (Information and communications technology) contributes nearly 3% in the overall carbon footprint [19]. Thus, power management in embedded systems is also important for achieving the goals of green computing.

3 Overview

Based on their main energy saving approach, we classify the techniques into following categories.

1. DVFS (dynamic voltage and frequency scaling) and power-aware scheduling based techniques [20–74].

2. Using low power modes, called power mode management (PMM) [25, 27, 32, 37, 56, 73, 75–81]. Note that this is also sometimes referred to as 'dynamic power management'. We however use a more specific term 'power mode management' to avoid any confusion.

3. Microarchitectural techniques for saving energy in specific components e.g. main memory ([82–84]), cache [6, 16, 35, 85–103], scratchpad memory [104], TLB [105] or making other changes to memory hierarchy e.g. adding extra components [106, 107].

4. Using unconventional-cores such as DSP or GPUs of FPGAs [108–117].

4 Power Management Techniques

We now discuss some power management techniques in detail. As evident from the classification in previous section, several techniques can be classified in more than one category (e.g. DVFS and PMM). For sake of brevity, we discuss them in one category only.

4.1 DVFS and Power-Aware Scheduling based Techniques

For sake of convenience, we discuss power-aware scheduling techniques along with DVFS since these techniques often make scheduling decisions with a view to use DVFS for saving energy.

DVFS is a technique for altering the voltage and/or frequency of a computing system based on performance and power requirements. For CMOS circuits, dynamic power is related with voltage and frequency as $P \propto FV^2$ and hence, by reducing the frequency, the voltage at which the circuit needs to be operated for stable operation can also be lowered, which leads to energy saving. Several commercial microprocessors support DVFS technology for saving power, e.g. AMD PowerNow and and Intel's SpeedStep. The limitation of DVFS is that it harms the performance and hence, it may increase execution time or lead to missed deadlines. Also, DVFS requires programmable clock generator and DC-DC converter which incur energy overhead. Further, voltage transitions may require time on the order of tens of microseconds ([30]). Finally, due to increase in leakage energy and trend of using multi-core processor instead of increasing clock frequency, the returns from DVFS are diminishing.

Hua et al. [44] study the opportunity available for saving energy in embedded systems using DVFS. Since supporting a large number of voltage levels causes overhead (e.g. area and power overhead of voltage regulators, overhead of transitions), their work explores the optimal number of voltage levels and their values to implement on the multiple-voltage system for achieving energy efficiency. They have shown that systems which provides 3 or 4 voltages are nearly as energy efficient as the ideal system that can vary the voltage arbitrarily.

Kianzad et al. [54] use genetic algorithm to integrate task scheduling and voltage scaling under a single iterative optimization loop. Their technique searches the solution space to find an assignment and ordering of tasks on each processing element and generates a schedule such that deadline constraints are met and the power consumption is minimized. Further, their technique distributes the slack proportionately to different tasks and uses DVFS to save energy. They propose techniques for saving energy in both homogeneous and heterogeneous multiprocessor embedded systems.

In several multimedia applications, missing some task deadlines can be acceptable since it remains unnoticed to human visual and auditory system. Hua et al. [45] utilize this fact, along with the information on statistical task execution time to propose techniques to save energy in embedded systems by dynamic voltage scaling. They have proposed two algorithms. The first algorithm ensures achieving highest completion ratio with lowest possible energy consumption. The second algorithm deliberately drops some tasks to create slack for saving additional energy, such that application-

specific quality-of-service constraint is fulfilled. Thus, their algorithms provide opportunity to exercise trade-off between achieving high energy saving and achieving high task completion ratio (i.e. low deadline miss ratio).

Choi et al. [46] propose a DVFS technique which enables achieving a precise energy-performance trade-off. Their technique makes use of runtime information about the external memory access statistics and chooses the optimal CPU clock frequency and the corresponding minimum voltage level based on the ratio of the on-chip computation time to the off-chip access time. Their technique lowers the CPU frequency in the memory-bound region of a program to keep the performance degradation to a low value.

Gheorghita et al. [47] propose a technique for saving energy in embedded systems by utilizing the knowledge of operation modes. As an example, a portable MP3 player may have different application scenarios e.g. actual use for listening, connection with computer etc. Further, the player provides two operation modes viz. mono or stereo. Since mono mode requires less computation power, battery power can be saved by using smaller voltage at the time of low resource usage. This observation can also be used to guide design-time choices to provide specific operation modes for specific use-case scenarios. They also show the application of their technique for both hard and soft real-time systems.

Saewong et al. [58] propose four DVFS techniques for saving energy in embedded systems. Their first technique selects a single frequency which is used for the entire execution, such that the task-deadline is met and energy is minimized. This technique is suitable for systems where the overhead of DVFS is very high. The third technique finds suitable frequency like the first technique with the difference that it takes decision in the order of priority of tasks. Hence, if meeting the deadline of a higher priority task requires running other tasks at larger frequency than what is minimally required for meeting their deadlines, extra slack is created which can be used to save extra energy in low-priority tasks by further reducing their frequency. Another technique uses non-linear optimization to find the optimal frequency for every task. This technique, however, has large complexity and hence is unsuitable for on-line use. The fourth technique additionally monitors the actual execution time of tasks and further minimizes the energy consumption of tasks whose execution times are less than the pre-reserved worst-case execution times.

Quan et al. [59] propose two DVFS algorithms for saving energy in real-time embedded systems. The first algorithm finds the minimum constant speed that can be applied throughout the execution of the whole tasks set, such that the processor is shut down when idle. The second algorithm produces both the constant speed and schedule of variable voltage for minimizing energy. The second algorithm always saves more energy that the first algorithm.

Zhu et al. [60] propose a technique for saving energy in multiprocessor systems. Their technique allows the processors to share the reclaimed slack which arises from a shorter execution time in one of the processors. Using this extra slack, the speed of future tasks can be reduced which results in energy savings. They show that sharing the slack also helps in ensuring that all the deadlines are met. They show the effectiveness of their technique for both tasks with dependence (i.e. precedence constraints) and without dependence.

Xian et al. [61] present a technique for scheduling in multiprocessor systems to save energy. For scheduling periodic real-time tasks, their technique uses earliest deadline first scheduling to ensure meeting the deadlines of all tasks while minimizing energy consumption. Since this problem is NP-hard, they present a polynomial time heuristic method. For this, the problem is solved as a load-balancing problem assuming that unbounded and continuous range of frequencies are available. In the second step, this solution is modified to account for the maximum available frequency and the bounded discrete frequencies.

Kan et al. [41] propose a DVFS technique for saving energy in soft real-time embedded systems. They find the optimal frequency for a task assuming availability of continuous range of frequencies. Afterwards, from the actually available frequencies, the closest frequencies which are smaller and larger than the optimal frequency are chosen and fraction of time each of them should be used to meet the deadline is decided. They also suggest that difference in average-case and worst-case execution time gives rise to multiple deadlines and hence, if the first deadline is missed, the algorithm can try to meet the next deadline while conserving extra amount of energy.

Yang et al. [71] propose technique for mapping concurrent tasks onto a heterogeneous multiprocessor platform in an energy efficient manner. On different processors, the execution time and energy consumption of the tasks are different and hence, their technique finds different task-ordering and processor-assignment possibilities , and generate a Pareto-optimal set, where every point is better than any other one in at least one way (i.e., either energy efficiency or execution speed). This information is used at runtime to save energy using dynamic voltage scaling and to find a global energy efficient solution.

Kim et al. [30] discuss per-core DVFS technique for saving energy in embedded systems. Scaling CPU frequency slows down the CPU-bound operations but has little effect on memory-bound operations and based on this fact, the voltage and frequency are reduced during memory-bound intervals of an application. Their algorithm finds suitable voltage/frequency setting in an offline manner. Use of on-chip regulators enables fine-grained voltage transitions using which the memory-bound intervals can be more effectively exploited. They also show that per-core DVFS scheme can better exploit the scaling opportunities in the individual threads and thus provides significant improvement over system-wide DVFS scheme. Further, they study the effect of voltage

transition time, overhead and regulator losses on the benefit obtained from DVFS.

4.2 Using Power Modes

In embedded systems, the hardware typically provides a range of operating modes which can be used to save energy. Different modes consume different amount of power and take different time to return back to the normal mode. In general, the modes with lower energy consumption also take the largest time to return to the normal mode and vice versa. For saving energy while keeping the performance loss bounded, these modes should be judiciously used. Also, while a low-power mode can be used when the system is idle, the system must return to the normal mode for actually servicing a request or performing the task.

Li et al. [78] propose a method for selecting the power modes for the optimal power management of embedded systems under timing and power constraints. Their method determines the schedule of mode transitions such that the whole system can meet all power and timing constraints.

Hoeller et al. [79] propose an interface for power management of hardware and software components. They method allows applications to express when certain components are not being used and based on this information, individual components, subsystems or the whole system can be transitioned to low-power modes. This frees the programmer from the task of individually managing the power consumption of each component.

Huang et al. [76] propose an energy saving technique which works by adaptively controlling the power mode of the embedded system according to historical arrivals of tasks. Their technique takes decision regarding when to transition the system to low-power from normal-power mode or vice versa, based on the relative time overhead and energy advantage from mode transition and the consideration of meeting the deadlines of the tasks.

Bhatti et al. [26] present an online framework to integrate DVFS with power-mode management (PMM) scheme to save energy in embedded systems. Their scheme utilizes conventional DVFS and PMM schemes and uses machine-learning approach to adapt at runtime to the best-performing policy for any given workload. To save energy, DVFS policy makes use of dynamic slack while PMM makes use of idle time intervals. They have shown that their technique achieves energy savings comparable to the best-performing policy at any time. Also, their framework allows use of existing as well as new DVFS and PMM schemes.

Niu et al. [6] propose a technique to save both leakage and dynamic energy in embedded systems by integrating DVFS and PPM. In the case when processor is active, their technique chooses a processor speed such that the dynamic and leakage power consumption are balanced. Further, in the case when the processor is idle, the coming tasks are delayed as much as possible, such that

their deadlines are not missed and scattered, short inter-task idle intervals are merged in a few large idle intervals. Large idle intervals lead to reduced mode transition overhead, since the processor can stay in either idle or active state continuously for longer time.

Kim et al. [55] study the trade-off between voltage scaling and dynamic power-mode management. Under the assumption that voltage scaling does not reduce energy consumption in peripheral devices, voltage scaling increases the execution time and thus the leakage energy consumption of peripheral devices is increased and opportunity to transition them to low-power mode is reduced. Towards this, they propose a technique which exploits task-slack by partitioning the task execution into several intervals and shuts down the unneeded peripheral device on a per-interval basis.

Shin et al. [32] propose a technique for saving energy by integrating DVFS and PMM. They note that in real-time embedded systems, idle intervals can arise due to either inherent slacks, need to maintain priorities or early completion of tasks than their worst case estimates. To exploit inherent slack for saving energy, they use an offline DVFS approach. Further, they use an online approach which evaluates both DVFS and PMM to find the best way to exploit the remaining two kinds of idle intervals for saving energy.

Cheng et al. [80] propose an online technique for performing energy-aware I/O scheduling for hard real-time systems. Their technique utilizes device slack to perform power mode transitions to save energy, while maintaining temporal correctness. Their technique performs inter-task scheduling and not intra-task scheduling, since it may lead to missed deadlines which may have severe consequences in hard real-time systems. The decision to transition are taken based on break-even time calculation, which shows the minimum time the device needs to be idle for the mode transition to provide positive energy savings.

Awan et al. [81] propose an approach for saving energy in embedded systems using multiple low-power modes. Their technique computes the break-even time for each mode using offline analysis. Further, since early completion of high-priority task creates slack, their technique accumulates this task and uses it to save extra leakage energy in lower priority tasks by allowing the device to stay in low-power mode for longer time.

4.3 Saving Energy in Specific Components

Several researchers propose microarchitectural techniques for saving energy in specific components of embedded systems. These techniques leverage application properties or variation in workload to dynamically reconfigure the component of the system to save energy.

Yang et al. [82] discuss a technique for saving main memory energy in embedded systems. Their technique uses software-based RAM compression to increase the effective size of the memory. The memory compression

is used only for those applications which may gain benefit in performance or energy from the compression. For such applications, compression of memory data and swapped-out pages is performed in an online manner, thus dynamically adjusting the size of the compressed RAM area.

Trajkovic et al. [83] propose a buffering based technique for saving energy in low-power embedded systems. Their technique is based on the observation that since the DRAMs allow the row to be left 'on' after a memory access; if in a synchronous DRAM (SDRAM), two memory access (i.e. read/write) operation are done in a same activate-precharge cycle, then the overhead of activation and precharging can be avoided. Using this observation, their technique prefetches additional cache blocks on read accesses and combines multiple blocks (which are to be written to the same DRAM row) in write accesses. Their technique uses small storage structures to store the extra prefetched lines and to buffer the writes to the same DRAM row. By adapting the above mentioned write-combining and prefetching schemes for each application, their technique reduces the memory power consumption.

Reddy et al. [89] present an approach for saving cache energy in multitasking embedded systems. Their algorithm selects best cache partitioning for different running applications in offline manner and uses this information to allocate cache at runtime. Their algorithm also reduces inter-task interference in a preemptive multitasking environment.

Tsai et al. [106] propose a technique for saving energy in embedded processors by using a memory structure called "Trace Reuse cache" (TRC). The TRC is used at the same level of memory hierarchy as conventional instruction cache. TRC reuses the retired instructions from the pipeline back-end of a processor and efficiently delivers the instructions in the form of traces. Thus, TRC enables the processor to achieve a higher instruction rate, which leads to improvement in both performance and energy efficiency.

Hajimiri et al.[92] integrate cache reconfiguration and code compression to improve both performance and energy efficiency of embedded systems. For a single-level cache hierarchy, their technique performs exhaustive exploration of the cache design space by varying different parameters such as line size, associativity and total size; and simulating each one of the resultant configuration. Based on the results, the configuration with minimum energy can be selected. The code compression scheme integrates synergistically with cache reconfiguration, since code compression improves performance by reducing memory traffic and bandwidth usage and thus it partially offsets the performance loss resulting from cache reconfiguration.

It is well-known that there exists a large intra- and inter-program variation in cache requirement of different applications. Using this, the cache can be dynamically reconfigured for each program or program phase and the unused cache is turned off to save energy. Based on this

idea, Albonesi [103] proposes selective-ways approach where some of the ways of the cache are turned off to save energy, such that performance degradation remains bounded.

Zhang et al. [99] propose a highly-configurable cache architecture for facilitating dynamic reconfiguration to save energy in embedded systems. Their cache architecture contains four separate banks that can operate as four separate ways. By concatenating these ways, the associativity of the cache can be changed to either 1, 2 or 4. Also, if desired, some ways can be shut down. Further, by configuring the fetch unit to fetch different size of cache lines, the line size (block size) of cache can also be altered.

Several researchers use this architecture to save cache energy. For single-core systems, Wang et al. [86] profile several possible configurations of L1 data cache, L1 instruction cache and unified L2 cache in offline manner. At runtime, different possible combinations of two-level cache hierarchy are explored to find the most energy efficient configuration. For multi-core systems with private L1 caches (data and instruction) and shared L2 cache, Wang et al. [16] propose using static profiling for exploring different possible combinations and dynamically reconfiguring L1 cache and partitioning L2 cache for saving energy. Similarly, Rawlins et al. [118] discuss their technique for saving cache energy in heterogeneous dual-core systems by tuning the size of L1 cache, while addressing the issues presented by multicore operation such as core-interactions, data coherence etc.

Kin et al. [102] propose a technique for saving energy in embedded systems by filtering access to cache. Their technique places a small filter cache in front of the conventional L1 cache. Based on temporal locality of data access, most of the accesses are served from the data present in the filter cache and hence the number of L1 accesses is reduced which saves dynamic energy. The trade-off involved in their technique is that for achieving a reasonably high hit-rate in the filter cache, large filter cache is required which, in turn, increases its access time and energy consumption.

Some researchers have used scratchpad memory for saving energy in embedded systems. Scratchpad memory refers to on-chip SRAM that is mapped into an address space disjoint from the off-chip memory but connected to the same address and data buses. Compared to off-chip memory, both cache and scratchpad allow much faster access. The main difference between the cache and scratchpad is that the cache access may lead to either hit or miss, while the scratchpad guarantees a single-cycle access time. Steinke et al. [104] propose a technique for saving energy in embedded systems by utilizing scratchpad memory. At compile time their technique inserts copy functions into the application to copy a set of basic blocks which are frequently executed. Afterwards, at certain predetermined points in the execution of the program, their technique copies part of the program in the scratchpad and then executes the program from the scratchpad itself. This reduces the

access to cache and thus leads to saving of energy. The tradeoff in their approach is that copying instructions into scratchpad consumes more time and energy than a hardware-controlled cache fill.

Benini et al. [107] discuss the design of application-specific memory to save energy in embedded systems. Their technique works by mapping most frequently accessed locations onto a small memory that can be placed on-chip. Since such a memory is very close to the processor, access to it causes much smaller overhead than accessing large background memory. Their technique does not use cache as the local memory, since caches incur the overhead of tag comparison. Instead they use a local memory and create application-specific decoding logic which is automatically synthesized at design time based on profiling the embedded application on the processor. Compared to scratchpad, their local memory and decoding logic ensure that most frequently accessed data are stored in a small number of contiguous memory addresses.

Bournoutian et al. [91] present a technique for saving energy in embedded systems by reducing L1 cache misses. Their technique employs a small list to hold the set-number from which this line was evicted. On any cache eviction, this list is searched and if an entry with same set number is found, it indicates that the set is in probable state of thrashing. For such sets, a flag is turned on. When a cache miss is observed in sets with the flag turned on, a complementary set is also searched and if the data is found there, the cache access leads to hit. Thus, their technique effectively doubles the associativity of heavily used sets. This reduces the L1 cache misses which in turn reduces the overall execution time and power consumption by avoiding secondary memory accesses.

Mohapatra et al. [35] present a power management technique which integrates microarchitectural level, OS level and middleware level schemes for saving energy in mobile handheld devices. At microarchitectural level, they use static profiling to explore various cache configurations to find the energy-optimal configuration. At OS level, they use DVFS to save energy by exploiting the slack. At middleware level, they use network traffic regulation and admission control policies to save energy in network interface. They also study the interaction of these schemes to further optimize them for saving energy.

4.4 Using Unconventional Cores

As discussed before, the performance of modern computing systems is primarily shaped by power concerns. In such a regime, "unconventional" cores (or platforms) such as GPUs, FPGAs, ASICs and DSPs etc. [119, 120] hold a great promise for improving application performance and energy efficiency. For this reason, several researchers have used unconventional cores for power management of embedded systems.

Mu et al. [110] compare the energy efficiency of GPUs with that of DSPs for high performance embedded computing (HPEC) benchmark suite which includes a broad range of signal processing applications. They have observed that although GPU provides at least an order of magnitude better performance than the DSP, its energy efficiency (measured in performance per watt) is inferior to that of the DSP.

Mencer et al. [112] compare the energy efficiency of FPGAs with that of DSPs for IDEA (International Data Encryption Algorithm). They have observed that the FPGA provides an order of magnitude better energy efficiency than the DSPs.

Timm et al. [108] compare the performance and energy efficiency of CPU with that of GPUs for several applications, such as matrix multiplication and FFT etc. That have observed that GPU offers significant performance advantage over CPU and hence, despite consuming larger peak power, it outperforms CPU in energy efficiency. They also note that the advantage of GPU reduces for applications which do not provide substantial parallelism.

Ou et al. [121] propose a technique for energy-efficient mapping of embedded signal-processing applications on FPGA. They use dynamic-programming based approach for mapping beamforming applications which are used in air-borne or sea-borne vehicles. They have shown that compared to a greedy algorithm, their technique provides much larger energy savings.

Wang et al. [109] compare the performance and energy benefits of utilizing the integrated GPU and DSP cores to offload or share the compute-intensive tasks of CPU. They test three mobile computing systems viz. TI's OMAP3530, Qualcomn's Snapdragon S2, and Nvidia's Tegra 2. All these systems integrate both CPU and GPU. Further, TI's OMAP also integrates DSP and exposes it for programming by the user. They test three applications viz. FFT, matrix multiplication and 2D stencil. They have observed that by effectively using GPU and DSP along with CPU, significant improvement in performance and energy efficiency can be achieved.

Stitt et al. [113] propose an approach for moving critical software loops to reconfigurable hardware for saving energy. Typical benchmark programs spend a large fraction (e.g. 80%) of their time in a small portion of code. Thus, by Amdahl's law, to improve overall performance, this portion can be implemented on a application-specific or reconfigurable hardware which provides much better performance than the software. They demonstrate their approach by using ASIC and FPGA and observe large energy savings over a software-only implementation.

Llamocca et al. [115] compare the performance and energy efficiency of GPU and FPGA for 2D FIR (finite-impulse response) filter. This program finds application in video processing. They have observed that although FPGA provides lower performance than the GPU, it outperforms GPU on the metric of energy efficiency. The high performance of GPU is due to higher frequency and its ability to exploit massive parallelization present in the algorithm.

Fowers et al. [116] evaluate sliding window program on multi-core CPU, FPGA and GPU. This program has applications in digital signal processing. They have observed that FPGA provides an order of magnitude better performance while using an order of magnitude less energy compared to both CPU and GPU. These results demonstrate the utility of FPGAs for implementation of embedded system performing high-definition video processing.

5 Concluding Remarks

The next generation mobile computing systems will possess capabilities for high-speed video processing and communication which will require at least an order of magnitude better energy efficiency than what is available in state-of-the-art systems. This clearly highlights the need of power management in embedded systems. To cope with these challenges, power management is necessary at all levels, viz. chip-design level, microarchitectural level, application level and system level.

In this paper, we reviewed several power management techniques for embedded systems and classified them based on their key research idea. It is hoped that by providing insights into the working of power management techniques, this paper would help the researchers in addressing the challenges of power consumption and architecting highly-energy efficient embedded systems of tomorrow.

References

[1] International Telecommunication Union. http://www.itu.int/dms_pub/itu-d/opb/ind/D-IND-ICTOI-2012-SUM-PDF-E.pdf, 2012.

[2] H. Ghasemzadeh and R. Jafari, "Ultra low power signal processing in wearable monitoring systems: A tiered screening architecture with optimal bit resolution," *ACM Transactions in Embedded Computing Systems (TECS)*, vol. 9, no. 4, 2013.

[3] GeForce GTX 590. http://www.geforce.com/hardware/desktop-GPUs/geforce-gtx-590/specifications, 2013.

[4] W. J. Dally, J. Balfour, D. Black-Shaffer, J. Chen, R. C. Harting, V. Parikh, J. Park, and D. Sheffield, "Efficient embedded computing," *Computer*, vol. 41, no. 7, pp. 27–32, 2008.

[5] J. Butts and G. Sohi, "A static power model for architects," in *MICRO*, pp. 191–201, ACM, 2000.

[6] L. Niu and G. Quan, "Reducing both dynamic and leakage energy consumption for hard real-time systems," in *2004 international conference on Compilers, architecture, and synthesis for embedded systems*, pp. 140–148, ACM, 2004.

[7] ITRS, "International Technology Roadmap for Semiconductors." http://www.itrs.net/Links/2011ITRS/2011Chapters/2011ExecSum.pdf, 2011.

[8] D. Anderson, J. Dykes, and E. Riedel, "More than an interface-SCSI vs. ATA," in *2nd USENIX Conference on File and Storage Technologies (FAST03)*, pp. 245–257, 2003.

[9] S. Gupta, S. Mittal, S. Dasgupta, and A. Mittal, "MIMO Systems For Ensuring Multimedia QoS Over Scarce Resource Wireless Networks," *ACM International Conference On Advance Computing, India*, 2008.

[10] A. Pande and S. Mittal, "BayWave: BAYesian WAVElet-based Image Estimation," *Int. J. of Signal and Imaging Systems Engineering (IJSISE)*, vol. 2, no. 4, pp. 155–162, 2009.

[11] S. Gupta et al., "Guaranteed QoS with MIMO Systems for Scalable Low Motion Video Streaming Over Scarce Resource Wireless Channels," in *Proceedings Second International Conference On Information Processing*, pp. 452–466, IK International Pvt Ltd, 2008.

[12] ARM, "ARM11MPCore processor." http://www.arm.com, 2013.

[13] MIPS, "MIPS32 1004K." http://www.mips.com/, 2013.

[14] NVIDIA, "Tegra 4." http://www.nvidia.com/object/tegra-4-processor.html, 2013.

[15] A. Munir, S. Ranka, and A. Gordon-Ross, "High-performance energy-efficient multicore embedded computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 4, pp. 684–700, 2012.

[16] W. Wang, P. Mishra, and S. Ranka, "Dynamic cache reconfiguration and partitioning for energy optimization in real-time multicore systems," in *Design Automation Conference*, pp. 948–953, ACM, 2011.

[17] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *International Symposium on Computer Architecture (ISCA)*, pp. 365–376, 2011.

[18] G. Venkatesh et al., "Conservation cores: reducing the energy of mature computations," in *ACM SIGARCH Computer Architecture News*, vol. 38, pp. 205–218, 2010.

[19] L. Smarr, "Project greenlight: Optimizing cyber-infrastructure for a carbon-constrained world," *Computer*, vol. 43, no. 1, pp. 22–27, 2010.

[20] B. Brock and K. Rajamani, "Dynamic power management for embedded systems," in *IEEE International SOC Conference (Systems-on-Chip)*, pp. 416–419, 2003.

[21] P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," in *ACM SIGOPS Operating Systems Review*, vol. 35, pp. 89–102, ACM, 2001.

[22] S. Srinivasan, J. Park, and V. Mooney III, "Combining data remapping and voltage/frequency scaling of second level memory for energy reduction in embedded systems," *Compiler Optimizations For Power-Aware Computing*, p. 12, 2003.

[23] R. Jejurikar and R. Gupta, "Dynamic slack reclamation with procrastination scheduling in real-time embedded systems," in *42nd annual Design Automation Conference*, DAC '05, (New York, NY, USA), pp. 111–116, ACM, 2005.

[24] J. Luo and N. K. Jha, "Battery-aware static scheduling for distributed real-time embedded systems," in *38th Design Automation Conference*, pp. 444–449, 2001.

[25] A. A. Fröhlich, "A comprehensive approach to power management in embedded systems," *International Journal of Distributed Sensor Networks*, vol. 2011, 2011.

[26] K. Bhatti, C. Belleudy, and M. Auguin, "Power management in real time embedded systems through online and adaptive interplay of DPM and DVFS policies," in *IEEE/IFIP 8th International Conference on Embedded and Ubiquitous Computing (EUC)*, pp. 184–191, 2010.

[27] M. Marinoni, M. Bambagini, F. Prosperi, F. Esposito, G. Franchino, L. Santinelli, and G. Buttazzo, "Platform-aware bandwidth-oriented energy management algorithm for real-time embedded systems," in *IEEE 16th Conference on Emerging Technologies & Factory Automation (ETFA)*, pp. 1–8, 2011.

[28] X. Qi, D. Zhu, and H. Aydin, "Global reliability-aware power management for multiprocessor real-time systems," in *Proc. of the IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2010.

[29] H. Yun, P.-L. Wu, A. Arya, C. Kim, T. Abdelzaher, and L. Sha, "System-wide energy optimization for multiple DVS components and real-time tasks," *Real-Time Systems*, vol. 47, no. 5, pp. 489–515, 2011.

[30] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks, "System level analysis of fast, per-core DVFS using on-chip switching regulators," in *IEEE 14th International Symposium on High Performance Computer Architecture (HPCA)*, pp. 123–134, 2008.

[31] Y. Zhang and K. Chakrabarty, "A unified approach for fault tolerance and dynamic power management in fixed-priority real-time embedded systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 1, pp. 111–125, 2006.

[32] Y. Shin, K. Choi, and T. Sakurai, "Power optimization of real-time embedded systems on variable speed processors," in *international conference on Computer-aided design*, pp. 365–368, 2000.

[33] M. T. Schmitz, B. M. Al-Hashimi, and P. Eles, "Energy-efficient mapping and scheduling for dvs enabled distributed embedded systems," in *Design, Automation and Test in Europe Conference and Exhibition*, pp. 514–521, 2002.

[34] F. Firouzi, M. Salehi, F. Wang, S. Fakhraie, and S. Safari, "Reliability-aware dynamic voltage and frequency scaling," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 304–309, IEEE, 2010.

[35] S. Mohapatra, R. Cornea, N. Dutt, A. Nicolau, and N. Venkatasubramanian, "Integrated power management for video streaming to mobile handheld devices," in *eleventh ACM international conference on Multimedia*, pp. 582–591, ACM, 2003.

[36] J. L. March, J. Sahuquillo, S. Petit, H. Hassan, and J. Duato, "Power-aware scheduling with effective task migration for real-time multicore embedded systems," *Concurrency and Computation: Practice and Experience*, 2012.

[37] Z. Karakehayov and E. Vassev, "Energy efficiency with runtime models for energy-aware embedded systems," in *Software Engineering Workshop (SEW), 2011 34th IEEE*, pp. 106–111, IEEE, 2011.

[38] R. M. Santos, J. Santos, and J. D. Orozco, "Power saving and fault-tolerance in real-time critical embedded systems," *Journal of Systems Architecture*, vol. 55, no. 2, pp. 90–101, 2009.

[39] M. Koedam, S. Stuijk, and H. Corporaal, "Exploiting inter and intra application dynamism to save energy," in *Digital System Design (DSD), 2011 14th Euromicro Conference on*, pp. 708–715, IEEE, 2011.

[40] D. E. Mera and N. G. Santiago, "Low power software techniques for embedded systems running real time operating systems," in *53rd IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1061–1064, 2010.

[41] E. Y. Kan, W. Chan, and T. Tse, "Leveraging performance and power savings for embedded systems using multiple target deadlines," in *10th International Conference on Quality Software (QSIC)*, pp. 473–480, IEEE, 2010.

[42] X. Chen, Q. Liu, and J. Lai, "A new power-aware scheduling algorithm for distributed system," in *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on & Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, pp. 338–343, IEEE, 2010.

[43] M.-F. Chang and W.-Y. Liang, "Learning-directed dynamic voltage and frequency scaling for computation time prediction," in *IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 1023–1029, 2011.

[44] S. Hua and G. Qu, "Approaching the maximum energy saving on embedded systems with multiple voltages," in *IEEE/ACM international conference on Computer-aided design*, p. 26, 2003.

[45] S. Hua, G. Qu, and S. S. Bhattacharyya, "An energy reduction technique for multimedia application with tolerance to deadline misses," in *Design Automation Conference*, pp. 131–136, IEEE, 2003.

[46] K. Choi, R. Soma, and M. Pedram, "Fine-grained dynamic voltage and frequency scaling for precise energy and performance trade-off based on the ratio of off-chip access to on-chip computation times," in *Conference on Design, automation and test in Europe-Volume 1*, p. 10004, 2004.

[47] S. V. Gheorghita, T. Basten, and H. Corporaal, "Application scenarios in streaming-oriented embedded system design," in *International Symposium on System-on-Chip*, pp. 1–4, IEEE, 2006.

[48] K. Choi, R. Soma, and M. Pedram, "Dynamic voltage and frequency scaling based on workload decomposition," in *International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 174–179, 2004.

[49] L. Yuan and G. Qu, "Analysis of energy reduction on dynamic voltage scaling-enabled systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 12, pp. 1827–1837, 2005.

[50] D. Wu, B. M. Al-Hashimi, and P. Eles, "Scheduling and mapping of conditional task graph for the synthesis of low power embedded systems," in *Computers and Digital Techniques, IEE Proceedings*, vol. 150, pp. 262–73, IET, 2003.

[51] Z. Shao, M. Wang, Y. Chen, C. Xue, M. Qiu, L. T. Yang, and E.-M. Sha, "Real-time dynamic voltage loop scheduling for multi-core embedded systems," *Circuits and Systems II: Express Briefs, IEEE Transactions on*, vol. 54, no. 5, pp. 445–449, 2007.

[52] R. N. Mahapatra and W. Zhao, "An energy-efficient slack distribution technique for multimode distributed real-time embedded systems," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 16, no. 7, pp. 650–662, 2005.

[53] Y. Cho, N. Chang, C. Chakrabarti, and S. Vrudhula, "High-level power management of embedded systems with application-specific energy cost functions," in *43rd Design Automation Conference*, pp. 568–573, ACM, 2006.

[54] V. Kianzad, S. S. Bhattacharyya, and G. Qu, "CASPER: an integrated energy-driven approach for task graph scheduling on distributed embedded systems," in *IEEE International Conference on Application-Specific Systems, Architecture Processors (ASAP)*, pp. 191–197, 2005.

[55] M. Kim and S. Ha, "Hybrid run-time power management technique for real-time embedded system with voltage scalable processor," in *ACM SIGPLAN Notices*, vol. 36, pp. 11–19, ACM, 2001.

[56] V. Devadas and H. Aydin, "On the interplay of dynamic voltage scaling and dynamic power management in real-time embedded applications," in *8th ACM international conference on Embedded software*, pp. 99–108, ACM, 2008.

[57] F. Gruian and K. Kuchcinski, "Lenes: task scheduling for low-energy systems using variable supply voltage processors," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 449–455, IEEE, 2001.

[58] S. Saewong and R. Rajkumar, "Practical voltage-scaling for fixed-priority rt-systems," in *IEEE Real-Time and Embedded Technology and Applications Symposium*, pp. 106–114, 2003.

[59] G. Quan and X. Hu, "Energy efficient fixed-priority scheduling for real-time systems on variable voltage processors," in *Design Automation Conference*, pp. 828–833, 2001.

[60] D. Zhu, R. Melhem, and B. R. Childers, "Scheduling with dynamic voltage/speed adjustment using slack reclamation in multiprocessor real-time systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 7, pp. 686–700, 2003.

[61] C. Xian, Y.-H. Lu, and Z. Li, "Energy-aware scheduling for real-time multiprocessor systems with uncertain task execution time," in *Design Automation Conference*, pp. 664–669, IEEE, 2007.

[62] C. Park, H. Kim, and J. Kim, "A low-power implementation of 3d graphics system for embedded mobile systems," in *2006 IEEE/ACM/IFIP Workshop on Embedded Systems for Real Time Multimedia*, pp. 53–58, IEEE Computer Society, 2006.

[63] J. Luo and N. K. Jha, "Power-conscious joint scheduling of periodic task graphs and aperiodic tasks in distributed real-time embedded systems," in *2000 IEEE/ACM international conference on Computer-aided design*, pp. 357–364, IEEE Press, 2000.

[64] D. Rakhmatov and S. Vrudhula, "Energy management for battery-powered embedded systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 2, no. 3, pp. 277–324, 2003.

[65] J. Liu, P. H. Chou, N. Bagherzadeh, and F. Kurdahi, "Power-aware scheduling under timing constraints for mission-critical embedded systems," in *38th Design Automation Conference*, pp. 840–845, 2001.

[66] H.-C. Wang and C.-W. Yao, "Task migration for energy conservation in real-time multi-processor embedded systems," in *International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 393–398, IEEE, 2011.

[67] K.-M. Cho, C.-H. Liang, J.-Y. Huang, and C.-S. Yang, "Design and implementation of a general purpose power-saving scheduling algorithm for embedded systems," in *IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 1–5, 2011.

[68] L. Niu, "Energy efficient scheduling for real-time embedded systems with qos guarantee," *Real-Time Systems*, vol. 47, no. 2, pp. 75–108, 2011.

[69] Y. Ma, N. Sang, W. Jiang, and L. Zhang, "Feedback-controlled security-aware and energy-efficient scheduling for real-time embedded systems," *Embedded and Multimedia Computing Technology and Service*, pp. 255–268, 2012.

[70] P. Yang and F. Catthoor, "Pareto-optimization-based run-time task scheduling for embedded systems," in *First IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, pp. 120–125, IEEE, 2003.

[71] P. Yang, P. Marchal, C. Wong, S. Himpe, F. Catthoor, P. David, J. Vounckx, and R. Lauwereins, "Managing dynamic concurrent tasks in embedded real-time multimedia systems," in *15th international symposium on System Synthesis*, pp. 112–119, ACM, 2002.

[72] W. Yuan, K. Nahrstedt, S. V. Adve, D. L. Jones, and R. H. Kravets, "Grace-1: Cross-layer adaptation for multimedia quality and battery energy," *IEEE Transactions on Mobile Computing*, vol. 5, no. 7, pp. 799–815, 2006.

[73] P. De Langen and B. Juurlink, "Trade-offs between voltage scaling and processor shutdown for low-energy embedded multiprocessors," *Embedded Computer Systems: Architectures, Modeling, and Simulation*, pp. 75–85, 2007.

[74] P.-C. Chang, I.-W. Wu, J.-J. Shann, and C.-P. Chung, "ETAHM: An energy-aware task allocation algorithm for heterogeneous multiprocessor," in *Design Automation Conference (DAC)*, pp. 776–779, 2008.

[75] S. K. Shukla and R. K. Gupta, "A model checking approach to evaluating system level dynamic power management policies for embedded systems," in *High-Level Design Validation and Test Workshop, 2001. Proceedings. Sixth IEEE International*, pp. 53–57, IEEE, 2001.

[76] K. Huang, L. Santinelli, J.-J. Chen, L. Thiele, and G. C. Buttazzo, "Adaptive power management for real-time event streams," in *Asia and South Pacific Design Automation Conference*, pp. 7–12, 2010.

[77] Q. Muhammad Yasir, M.-M. Klaus D, *et al.*, "Data cache-energy and throughput models: design exploration for embedded processors," *EURASIP journal on embedded systems*, vol. 2009, 2010.

[78] D. Li, P. H. Chou, and N. Bagherzadeh, "Mode selection and mode-dependency modeling for power-aware embedded systems," in *Asia and South Pacific Design Automation Conference*, p. 697, IEEE Computer Society, 2002.

[79] A. Hoeller, L. Wanner, and A. Fröhlich, "A hierarchical approach for power management on mobile embedded systems," *From Model-Driven Design to Resource Management for Distributed Embedded Systems*, pp. 265–274, 2006.

[80] H. Cheng and S. Goddard, "Online energy-aware I/O device scheduling for hard real-time systems," in *Conference on Design, automation and test in Europe*, pp. 1055–1060, European Design and Automation Association, 2006.

[81] M. A. Awan and S. M. Petters, "Enhanced race-to-halt: A leakage-aware energy management approach for dynamic priority systems," in *23rd Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 92–101, IEEE, 2011.

[82] L. Yang, R. P. Dick, H. Lekatsas, and S. Chakradhar, "Online memory compression for embedded systems," *ACM Trans. Embed. Comput. Syst.*, vol. 9, pp. 27:1–27:30, Mar. 2010.

[83] J. Trajkovic, A. Veidenbaum, and A. Kejariwal, "Improving SDRAM access energy efficiency for low-power embedded systems," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 7, no. 3, p. 24, 2008.

[84] M. Mamidipaka and N. Dutt, "On-chip stack based memory organization for low power embedded architectures," in *Design, Automation and Test in Europe Conference and Exhibition, 2003*, pp. 1082–1087, IEEE, 2003.

[85] S. Mittal and Z. Zhang, "EnCache: Improving Cache Energy Efficiency Using A Software-Controlled Profiling Cache," in *IEEE International Conference On Electro/Information Technology*, (Indianapolis, USA), May 2012.

[86] W. Wang and P. Mishra, "Dynamic reconfiguration of two-level caches in soft real-time embedded systems," in *VLSI, 2009. ISVLSI'09. IEEE Computer Society Annual Symposium on*, pp. 145–150, IEEE, 2009.

[87] W. Wang, S. Ranka, and P. Mishra, "A general algorithm for energy-aware dynamic reconfiguration in multitasking systems," in *24th International Conference on VLSI Design (VLSI Design), 2011*, pp. 334–339, IEEE, 2011.

[88] S. Mittal, Z. Zhang, and Y. Cao, "CASHIER: A Cache Energy Saving Technique for QoS Systems," *26th International Conference on VLSI Design (VLSID)*, pp. 43–48, 2013.

[89] R. Reddy and P. Petrov, "Cache partitioning for energy-efficient and interference-free embedded multitasking," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 9, no. 3, p. 16, 2010.

[90] M. Paul and P. Petrov, "Dynamically Adaptive I-Cache Partitioning for Energy-Efficient Embedded Multitasking," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, no. 99, pp. 1–14, 2011.

[91] G. Bournoutian and A. Orailoglu, "Miss reduction in embedded processors through dynamic, power-friendly cache design," in *Design Automation Conference (DAC)*, pp. 304–309, 2008.

[92] H. Hajimiri, K. Rahmani, and P. Mishra, "Synergistic integration of dynamic cache reconfiguration and code compression in embedded systems," in *International Green Computing Conference and Workshops (IGCC)*, pp. 1–8, IEEE, 2011.

[93] S. Mittal, *Dynamic cache reconfiguration based techniques for improving cache energy efficiency*. PhD thesis, Iowa State University, 2013.

[94] P. Petrov and A. Orailoglu, "Performance and power effectiveness in embedded processors customizable partitioned caches," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 20, no. 11, pp. 1309–1318, 2001.

[95] C.-Y. Tseng and H.-C. Chen, "The design of way-prediction scheme in set-associative cache for energy efficient embedded system," in *WRI International Conference on Communications and Mobile Computing*, vol. 3, pp. 3–7, IEEE, 2009.

[96] M. Alipour, M. E. Salehi, and K. Moshari, "Cache power and performance tradeoffs for embedded applications," in *IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, pp. 26–31, 2011.

[97] S. Mittal and Z. Zhang, "Palette: A cache leakage energy saving technique for green computing," in *HPC: Transition Towards Exascale Processing*, Advances in Parallel Computing, IOS Press, 2013.

[98] J. W. Kwak and J. H. Choi, "Selective access to filter cache for low-power embedded systems," in *43rd Hawaii International Conference on System Sciences (HICSS)*, pp. 1–8, IEEE, 2010.

[99] C. Zhang, F. Vahid, and W. Najjar, "A highly configurable cache architecture for embedded systems," in *ISCA*, pp. 136–146, IEEE, 2003.

[100] S. Mittal and Z. Zhang, "MANAGER: A Multicore Shared Cache Energy Saving Technique for QoS Systems," tech. rep., Electrical and Computer Engineering, Iowa State University, Iowa, USA, 2013.

[101] L. Chen, X. Zou, J. Lei, and Z. Liu, "Dynamically reconfigurable cache for low-power embedded system," in *Third International Conference on Natural Computation (ICNC)*, vol. 5, pp. 180–184, IEEE, 2007.

[102] J. Kin, M. Gupta, and W. Mangione-Smith, "The filter cache: an energy efficient memory structure," in *30th International symposium on Microarchitecture (MICRO)*, pp. 184–193, IEEE Computer Society, 1997.

[103] D. Albonesi, "Selective cache ways: On-demand cache resource allocation," in *32nd Annual IEEE International Symposium on Microarchitecture (MICRO)*, pp. 248–259, 1999.

[104] S. Steinke, N. Grunwald, L. Wehmeyer, R. Banakar, M. Balakrishnan, and P. Marwedel, "Reducing energy consumption by dynamic copying of instructions onto onchip memory," in *15th IEEE International Symposium on System Synthesis*, pp. 213–218, 2002.

[105] J.-H. Choi, J.-H. Lee, S.-W. Jeong, S.-D. Kim, and C. Weems, "A low power TLB structure for embedded systems," *Computer Architecture Letters*, vol. 1, no. 1, pp. 3–3, 2002.

[106] Y. Tsai and C. Chen, "Energy-efficient trace reuse cache for embedded processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,*, no. 99, pp. 1–14, 2011.

[107] L. Benini, A. Macii, E. Macii, and M. Poncino, "Increasing energy efficiency of embedded systems by application-specific memory hierarchy generation," *Design & Test of Computers, IEEE*, vol. 17, no. 2, pp. 74–85, 2000.

[108] C. Timm, A. Gelenberg, F. Weichert, and P. Marwedel, "Reducing the Energy Consumption of Embedded Systems by Integrating General Purpose GPUs," *TU, Dep. of Computer Science*, 2010.

[109] Y.-C. Wang and K.-T. T. Cheng, "Energy and performance characterization of mobile heterogeneous computing," in *IEEE Workshop on Signal Processing Systems (SiPS)*, pp. 312–317, 2012.

[110] S. Mu, C. Wang, M. Liu, D. Li, M. Zhu, X. Chen, X. Xie, and Y. Deng, "Evaluating the potential of graphics processors for high performance embedded computing," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1–6, 2011.

[111] Y.-C. Wang and K.-T. Cheng, "Energy-optimized mapping of application to smartphone platforma case study of mobile face recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 84–89, 2011.

[112] O. Mencer, M. Morf, and M. J. Flynn, "Hardware software tri-design of encryption for mobile communication units," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 5, pp. 3045–3048, 1998.

[113] G. Stitt, F. Vahid, and S. Nematbakhsh, "Energy savings and speedups from partitioning critical software loops to hardware in embedded systems,"

[114] P. Biswas, S. Banerjee, N. Dutt, P. Ienne, and L. Pozzi, "Performance and energy benefits of instruction set extensions in an FPGA soft core," in *International Conference on VLSI Design*, pp. 6–pp, IEEE, 2006.

[115] D. Llamocca, C. Carranza, and M. Pattichis, "Separable FIR filtering in FPGA and GPU implementations: Energy, Performance, and Accuracy considerations," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, pp. 363–368, IEEE, 2011.

[116] J. Fowers, G. Brown, P. Cooke, and G. Stitt, "A performance and energy comparison of FPGAs, GPUs, and multicores for sliding-window applications," in *ACM/SIGDA international symposium on Field Programmable Gate Arrays*, pp. 47–56, ACM, 2012.

[117] L. Strozek and D. Brooks, "Efficient architectures through application clustering and architectural heterogeneity," in *International conference on Compilers, architecture and synthesis for embedded systems*, pp. 190–200, ACM, 2006.

[118] M. Rawlins and A. Gordon-Ross, "CPACT-The conditional parameter adjustment cache tuner for dual-core architectures," in *IEEE 29th International Conference on Computer Design (ICCD), 2011*, pp. 396–403, IEEE, 2011.

[119] S. Mittal, S. Gupta, and S. Dasgupta, "FPGA: An Efficient And Promising Platform For Real-Time Image Processing Applications," in *National Conference On Research and Development In Hardware Systems (CSI-RDHS)*, (India), 2008.

[120] S. W. Keckler, W. J. Dally, B. Khailany, M. Garland, and D. Glasco, "GPUs and the future of parallel computing," *Micro, IEEE*, vol. 31, no. 5, pp. 7–17, 2011.

[121] J. Ou, S. Choi, and V. K. Prasanna, "Performance modeling of reconfigurable soc architectures and energy-efficient mapping of a class of application," in *11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pp. 241–250, 2003.