Energy Efficient Multi-Core Processing

Charles Leech and Tom J. Kazmierski

(Invited Paper)

***Abstract*—This paper evaluates the present state of the art of energy-efficient embedded processor design techniques and demonstrates, how small, variable-architecture embedded processors may exploit a run-time minimal architectural synthesis technique to achieve greater energy and area efficiency whilst maintaining performance. The picoMIPS architecture is presented, inspired by the MIPS, as an example of a minimal and energy efficient processor. The picoMIPS is a variable-architecture RISC microprocessor with an application-specific minimised instruction set. Each implementation will contain only the necessary datapath elements in order to maximise area efficiency. Due to the relationship between logic gate count and power consumption, energy efficiency is also maximised in the processor therefore the system is designed to perform a specific task in the most efficient processor-based form. The principles of the picoMIPS processor are illustrated with an example of the discrete cosine transform (DCT) and inverse DCT (IDCT) algorithms implemented in a multi-core context to demonstrate the concept of minimal architecture synthesis and how it can be used to produce an application specific, energy efficient processor.**

***Index Terms*—Embedded processors, application specific architectures, MIPS architecture, digital synthesis, energy efficient design, low power design.**

Review Paper
DOI: 10.7251/ELS1418003L

I. Introduction

THE energy efficiency of embedded processors is essential in mobile electronics where devices are powered by batteries. Processor performance has been increasing over the last few decades at a rate faster than the developments in battery technologies. This has led to a significant reduction of the battery life in mobile devices from days to hours. Also, new mobile applications demand higher performance and more graphically intensive processing. These demands are currently being addressed by many-core, high-frequency architectures which can deliver high-speed processing necessary to meet the tight execution deadlines. These two contradictory demands, the need to save energy and the requirement to deliver outstanding performance must be addressed by entirely new approaches. A number of research directions have appeared. Heterogeneous and reconfigurable embedded many-core systems can improve energy efficiency while maintaining high speed through judicious task scheduling and hardware adaptability. In a heterogeneous system, such as the ARM big.LITTLE architecture [1] smaller cores are employed to process simple and less demanding tasks to save energy while larger cores handle high performance and energy hungry processing when necessary. Reconfigurable architectures use flexible interconnect, power gating and software control within each core, thus achieving heterogeneity within the core. Reconfigurable cores can be configured in this way as either slower, but energy efficient processors, or faster, high-performance cores.. A number of approaches have been proposed to save energy within a core. Dynamic Voltage and Frequency Scaling (DVFS) [2] is a popular and well established technique where the supply voltage and the clock frequency are scaled to trade energy for performance and vice-versa. DVFS is typically implemented by including voltage regulators and phase-lock loop controlled clocks in the processor. The architecture is modified to allow the operating system to select a desired voltage and frequency through writing data to a DVFS control register. At any desired performance level, the operating system will put the processor into a minimum energy consumption mode. DVFS has proved very effective especially in applications where high performance is peaking only during a small fraction of the operating time as significant energy savings are achieved. Many other energy saving design techniques are currently being explored at the circuit, architecture and even system level. For example supply voltage in bus drivers can be reduced to extremely low levels to reduce bus energy consumption. New SRAM designs are being developed where energy consumption is reduced to extremely low levels in both the on-chip caches and the external memories. The architecture of processor cores are traditionally determined from a compromise of speed, power consumption, scalability, maintainability and extensibility. However, applications have different characteristics that require specific hardware implementations to enable optimal performance and therefore a system should be able to adapt its architecture to each application scenario. In this paper, we aim to demonstrate, through the evaluation of present technology, how small, variable-architecture embedded processors may exploit a run-time minimal architectural synthesis technique to achieve greater energy and area efficiency whilst maintaining performance.

II. Overview of Energy Efficiency Technologies

This section presents the current state of research in energy efficient technologies in multi-core systems for both traditional power saving techniques and novel technologies including heterogeneous and reconfigurable architectures. Through the analysis of present technology, we aim to demonstrate how a greater performance, energy efficiency and area efficiency balance can be achieved.

The introduction of multi-core structures to processor architectures has caused a significant increase in the power

Manuscript received 29 May 2014. Accepted for publication 12 June 2014.

C. Leech and T. J. Kazmierski are with the University of Southampton, UK (e-mail: {cl19g10, tjk}@ecs.soton.ac.uk).

consumption of these systems. In addition, the gap between average power and peak power has widened as the level of core integration increases [3]. A global power manager policy, such as that proposed by Isci et al, that has per-core control of parameters such as voltage and frequency levels is required in order to provide effective dynamic control of the power consumption [3]. Metrics such as performance-per-watt [4], [5], average and peak power or energy-delay product [6] are all used to quantify the power or energy efficiency of a system in order to evaluate it, however properties are prioritised differently depending on the application requirements. Modelling and simulation of multi-core processors is also an important process in order to better understand the complex interactions that occur inside a system and cause power and energy consumption [7]–[11]. For example, the model created by Basmadjian et al is tailored for multi-core architectures in that it accounts for resource sharing and power saving mechanisms [8].

A. Energy Efficiency techniques in Static Homogeneous Multi-core Architectures

Many energy efficiency and power saving technologies are already integrated into processor architectures in order to reduce power dissipation and extend battery life, especially in mobile devices. A combination of technologies is most commonly implemented to achieve the best energy efficiency whilst still allowing the system to meet performance targets [4]. Techniques to increase energy efficiency can be applied at many development levels from architecture co-design and code compilation to task scheduling, run-time management and application design [12]. A summary and analysis of these technologies is presented in the following section.

1) Dynamic Voltage and Frequency Scaling: Dynamic Voltage and Frequency Scaling (DVFS) is a technique used to control the power consumption of a processor through fine adjustment of the clock frequency and supply voltage levels [3], [4], [12], [13]. High levels are used when meeting performance targets is a priority and low levels (known as CPU throttling) are used when energy efficiency is most important or high performance is not required. When the supply voltage is lowered and the frequency reduced, the execution of instructions by the processor is slower but performed more energy efficiently due to the extension of delays in the pipeline stages. The rise and fall times for logic circuitry is increased along with the clock period meaning performance targets for applications must be relaxed. DVFS can be used in homogeneous multi-core architectures to emulate heterogeneity by controlling the frequency and supply to each core individually [14]. Each core therefore appears as though it has different delay properties however the architectures are still essentially identical. This per-core DVFS mechanism is investigated by Wonyoung et al who conclude that significant energy saving opportunities exist where on-chip integrated voltage regulators are used to provide nanosecond scale voltage switching [13]. DVFS can also be combined with thread migration to reduce energy consumption [4], [15]. Cai et al cite the problem that present DVFS energy saving techniques on multi-core systems assume one hardware context for each core whereas simultaneous multi-threading (SMT) is commonly implemented which causes these techniques to be less effective. Their novel technique, known as thread shuffling, uses concepts of thread criticality and thread criticality degree instead of thread-delay to maps together threads with similar criticality degree. This accounts for SMT when implementing DVFS and thread migration and achieves energy savings of up to 56% without impeding performance at all.

2) Clock Gating and Clock Distribution: Clock gating is a process, applied at the architectural design phase, to insert additional logic between the clock source and clock input of the processor's circuitry. During program execution, it reduces power consumption by logically disconnecting the clock of synchronous logic circuits to prevent unnecessary switching. Classed as a Dynamic Power Management (DPM) technique, as it is applied at run-time along with other techniques such as thread scheduling and DVFS to optimise the power/performance trade-off of a system [12]. The clock gating and distribution techniques implemented by Qualcomm in the Hexagon processor are analysed by Bassett et al on their ability to improve the energy efficiency of a digital signal processors (DSP) [16]. A low power clock network is implemented using multi-level clock gating strategies and spine-based clock distribution. The 4 levels of clock gating allows different size regions of the chip to be deactivated, from entire cores down to single logic cells. Further power reduction is achieved through a structured clock tree that aims to minimise the power consumed in distributing the clock signal across the chip whilst avoiding clock skew and delay. The clock tree structure (CTS) examined by Bassett et al is tested to give a 2 time reduction in skew over traditional CTS while power tests show reductions in power consumption by 8% for high-activity and over 35% for idle mode. Large portions of the chip will spend the majority of their time in idle more therefore high efficiency in this mode is critical.

3) Power Domains: Power domains are regions of a system or processor that are controlled from a single supply and can be completely powered down in order to minimise power consumption without entirely removing the power supply to the system. Power domains can be used dynamically and when used in conjunction with clock gating, lead to further improvements in energy efficiency. The ARM Cortex-A15 MPCore processor supports multiple power domains both for the core and for the surrounding logic [17]. Figure 1 shows these domains, labelled Processor and Non-Processor, that allow large parts of the processor to be deactivated. Smaller internal domains, such as CK_GCLKCR, are implemented to allow smaller sections to be deactivated for finer performance and power variations.

Power domains are often coupled with power modes as a means of switching on or off several power domains in order to enter low power, idle or shutdown states. The Cortex-A15 features multiple power modes with specific power domain configurations such as Dormant mode, where some Debug and L2 cache logic is powered down but the L2 cache RAMs are powered up to retain their state, or Powerdown mode where all power domains are powered down and the processor state

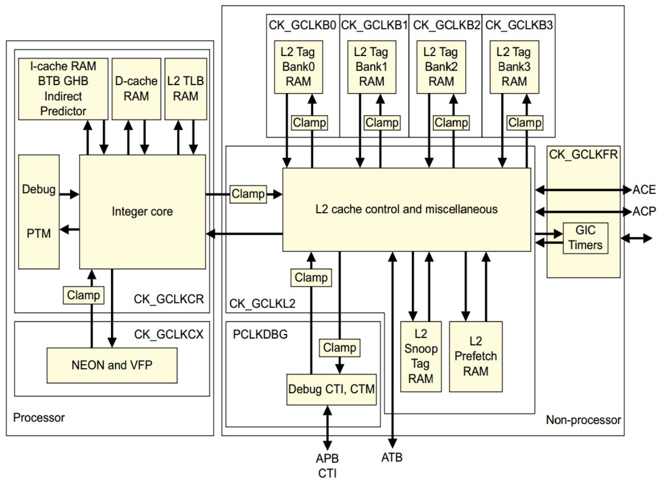

Fig. 1. The ARM Cortex-A15 features multiple power domains for the core and surrounding logic, reprinted from [17].

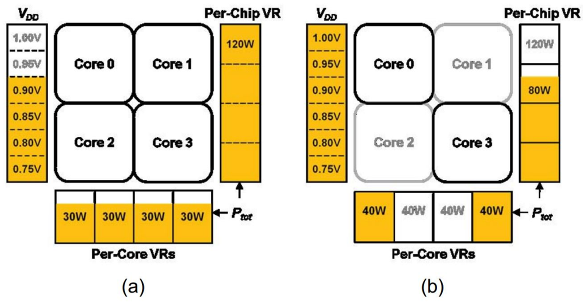

Fig. 2. Per-core power domains can provide reduce power consumption and higher performance levels, reprinted from [18].

is lost [17]. In multi-core architectures, power domains can be used to power down individual cores when idle or during non-parallel tasks in order to manage power consumption. Sinkar et al [18] and Ghasemi et al [19] present low-cost, per-core voltage domain solutions in order to improve performance in power-constrained processors. Figure 2 shows how Sinkar's mechanism can provide reduced chip power consumption but maintain per-core performance. In (a), a chip-wide power domain is shown with all cores active at the same level. In contrast, (b) shows a per-core power domain which allows unnecessary cores to be powered down and active cores to provide a higher performance level while still reducing the overall chip power level.

In the work by Rotem et al [20], topologies containing multiple clock, voltage and frequency domains are investigated in order to build a high-end chip multiprocessor (CMP) considering both power constraints and performance demands. These domains are controlled using DVFS techniques in connection with clock gating and are exercised by simulating power supply constrains. Results showed that multiple power domains can be beneficial for fully threaded applications whereas a single power domain is more suitable for light-threaded workloads. Power domains can be linked to DVFS techniques if the domain contains multiple voltage levels. Per-core power domains therefore enables per-core DVFS control such that each core can exploit run-time performance variations in multi-threaded applications [13], [18], [20].

4) Pipeline Balancing: Pipeline balancing (PLB) is a technique used to dynamically adjust the resources of the pipeline of a processor such that it retains performance while reducing power consumption [21]. The delay constraints on microarchitectural pipeline stages can be modified in order to make them more power efficient, in a similar way to DVFS, when the performance demand of the application is relaxed [22]. In work by Bahar et al, PLB operates in response to instruction per cycle (IPC) variations within a program [21]. The PLB mechanism dynamically reduces the issue width of the pipeline to save power or increases it to boost throughput. An 8-wide issue pipeline that has its unified issue queue divided into a left and right arbiter, separate left and right register files and functional units. A control unit is included can deactivate the right arbiter and its functional units to allow the pipeline to enter a low power mode. They show that this PLB technique can reduce power consumption of the issue queue and execution unit by up to 23% and 13% respectively with only an average performance loss of 1% to 2% [21]. Power Balanced pipelines is a concept in which the power disparity of pipeline stages is reduced by assigning different delays to each microarchitectural pipe stage while guaranteeing a certain level of performance/throughput [22]. In a similar fashion to [21], the concept uses cycle time stealing to redistribute cycle time from low power stages to stages that can perform more efficiently if given more time. Static power balancing is performed during design time to identify power heavy circuitry in pipe stages for which consumption remains fairly constant for different programs and reallocate cycle time accordingly. Dynamic power balancing is implemented on top of this to manage power fluctuations within each workload and further reduce the total power cost. Balancing of power rather than delay can result in a 46% power consumption reduction by the processor with no loss in throughput for a FabScalar processor over a baseline comparison. Power savings are also greater at lower frequencies.

5) Caches and Interconnects: It is not only the design of the processor's internal circuitry that is important in maintaining energy efficiency. Kumar et al conclude, from a paper examining the interconnections in multi-core architectures, that careful co-design of the interconnect, caches and the processor cores is required to achieve high performance and energy efficiency [23]. Through several theoretical examples, they examine parameters such as the area, power and bandwidth costs required to implement the processor-cache interconnect, showing that large caches sizes can constrain the interconnect's size and large interconnects can be power-hungry and inefficient. Zeng et al also recognise the high level of integration that is inherent in CMPs and attempt to reduce the interconnect power consumption by developing a novel cache coherence protocol [24]. Using their technique, results show that an average of 16.3% of L2 cache accesses could be optimised and as every access consumes time and power, an average 9.3% power reduction is recorded while increasing system performance by 1.4%.

B. Energy Efficiency techniques in Heterogeneous Multi-core Architectures

A heterogeneous or asymmetric multi-core architecture is composed of cores of varying size and complexity which are designed to compliment each other in terms of performance and energy efficiency [6]. A typical system will implement a small core to process simple tasks, in an energy efficient way, while a larger core provides higher performance processing for when computationally demanding tasks are presented. The cores represent different points in the power/performance design space and significant energy efficiency benefits can be achieved by dynamically allocating application execution to the most appropriate core [25]. A task matching or switching system is also implemented to intelligently assign tasks to cores; balancing a performance demand against maintaining system energy efficiency. These systems are particularly good at saving power whilst handling a diverse workload where fluctuations of high and low computational demand are common [26].

A heterogeneous architecture can be created in many different ways and many alternative have been developed due to the heavy research interest in this area. Modifications to general purpose processors, such as asymmetric core sizes [11], custom accelerators [27], varied caches sizes [14] and heterogeneity within each core [5], [28], have all been demonstrated to introduce heterogeneous features into a system.

One of the most prominent and successful heterogeneous architectures to date is the ARM big.LITTLE system. This is a production example of a heterogeneous multiprocessor system consisting of a compact and energy efficient “LITTLE” Cortex-A7 processor coupled with a higher performance “big” Cortex-A15 processor [26]. The system is designed with the dynamic usage patterns of modern smart phones in mind where there are typically periods of high intensity processing followed by longer periods of low intensity processing [29]. Low intensity tasks, such as texting and audio, can be handled by the A7 processor enabling a mobile device to save battery life. When a period of high intensity occurs, the A15 processor can be activated to increase the system’s throughput and meet tighter performance deadlines. A power saving of up to 70% is advertised for a light workload, where the A7 processor can handle all of the tasks, and a 50% saving for medium workloads where some tasks will require allocation to the A15 processor.

Kumar et al present an alternative implementation where two architectures from the Alpha family, the EV5 and EV6, are combined to be more energy and area efficient than a homogeneous equivalent [6], [25]. They demonstrate that a much higher throughput can be achieved due to the ability of a heterogeneous multi-core architecture to better exploit changes in thread-level parallelism as well as inter- and intra- thread diversity [6]. In [25], they evaluate the system in terms of its power efficiency indicating a 39% average energy reduction for only a 3% performance drop [25].

Composite Cores is a microarchitectural design that reduces the migration overhead of task switching by bringing heterogeneity inside each individual core [28]. The design

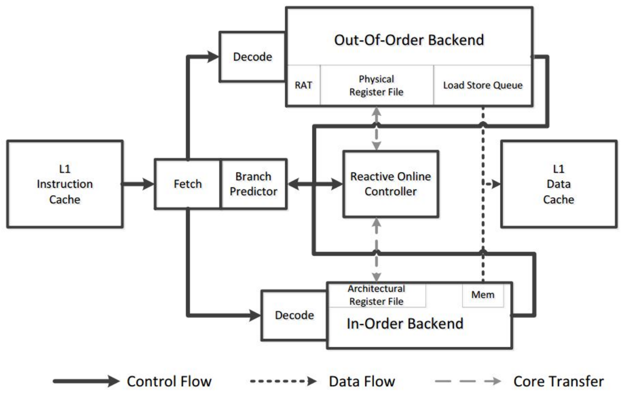

Fig. 3. The microarchitecture for Composite Cores, featuring two μEngines, reprinted from [28].

contains 2 separate backend modules, called μEngines, one of which features a deeper and more complex out-of-order pipeline, tailored for higher performance, while the other features a smaller, compact in-order pipeline designed with energy efficiency in mind. Figure Due to the high level of hardware resource sharing and the small μEngine state, the migration overhead is brought down from the order of 20,000 instructions to 2000 instructions. This greatly reduces the energy expenditure associated with migration and also allows more of the task to be run in an efficient mode. Their results show that the system can achieve an energy saving of 18% using dynamic task migration whilst only suffering a 5% performance loss.

Using both a heterogeneous architecture and hardware reconfiguration, a technique called Dynamic Core Morphing (DCM) is developed by Rodrigues et al to allow the shared hardware of a few tightly coupled cores to be morphed at runtime [5]. The cores all feature a baseline configuration but reconfiguration can trigger the re-assignment of high performance functional units to different cores to speed up execution. The efficiency of the system can lead to performance per watt gains of up to 43% and an average saving of 16% compared to a homogeneous static architecture.

The energy efficiency benefits of heterogeneity can only be exploited with the correct assignment of tasks or applications to each core [7], [10], [30]–[32]. Tasks must be assigned in order to maximise energy efficiency whilst ensuring performance deadlines are met. Awan et al perform scheduling in two phases to improve energy efficiency; task allocation to minimise active energy consumption and exchange of higher energy states to lower, more energy efficient sleep states [7]. Alternatively, Calcado et al propose division of tasks into m-threads to introduce fine-grain parallelism below thread level [33]. Moreover, Saha et al include power and temperature models into an adaptive task partitioning mechanism in order to allocate task according to their actual utilisations rather than based on a worst case execution time [10]. Simulation results confirm that the mechanism is effective in minimising energy consumption by 55% and reduces task migrations by 60% over alternative task partitioning schemes.

Tasks assignment can also be performed in response to program phases which naturally occur during execution when the resource demands of the application change. Phase detection is used by Jooya and Analoui to dynamically re-assigning programs for each phase to improve the performance and power dissipation of heterogeneous multi-core processors [31]. Programs are profiled in dynamic time intervals in order to detect phase changes. Sawalha et al also propose an online scheduling technique that dynamically adjusts the program-to-core assignment as application behaviour changes between phases with an aim to maximise energy efficiency [32]. Simulated evaluation of the scheduler shows energy saving of 16% on average and up to 29% reductions in energy-delay product can be achieved as compared to static assignments.

C. Energy Efficiency techniques in Reconfigurable Multi-core Architectures

Reconfigurability is another property that has the potential to increase the energy and area efficiency of processors and systems on chip by introducing adaptability and hardware flexibility into the architecture. Building on the innovations that heterogeneous architectures bring, reconfigurable architectures aim to achieve both energy efficiency and high performance but within the same processor and therefore meet the requirements of many embedded systems. The flexible heterogeneous Multi-Core processor (FMC) is an example of the fusion of these two architectures that can deliver both a high throughput for uniform parallel applications and high performance for fluctuating general purpose workloads [34]. Reconfigurable architectures are dynamic, adjusting their complexity, speed and performance level in response to the currently executing application. With this property in mind, we disregard systems that are statically reconfigurable but fixed while operating, such as traditional FPGAs, considering only architectures that are run-time reconfigurable.

1) Dynamic Partial Reconfiguration: FPGA manufacturers such as Xilinx and Altera now offer a mechanism called Dynamic Partial Reconfiguration (DPR) [35] or Self-Reconfiguration (DPSR) [36] to enable reconfiguration during run-time of the circuits within an FPGA, allowing a region of the design to change dynamically while other areas remain active [37]. The FPGA's architecture is partitioned into a static region consisting of fixed logic, control circuits and an embedded processor that control and monitor the system. The rest of the design space is allocated to a dynamic/reconfigurable region containing a reconfigurable logic fabric that can be formed into any circuit whenever hardware acceleration is required.

PDR/PDSR presents energy efficiency opportunities over fixed architectures. PDR enables the system to react dynamically to changes in the structure or performance and power constraints of the application, allowing it to address inefficiencies in the allocation of resources and more accurately implement changing software routines as dynamic hardware accelerators [35]. These circuits can then be easily removed or gated when they are no longer required to reduce power consumption [38]. PDR can also increase the performance of an FPGA based system because it permits the continued operation of portions of the dynamic region unaffected by reconfiguration tasks. Therefore, it allows multiple applications to be run in parallel on a single FPGA [36]. This property also improves the hardware efficiency of the system as, where separate devices were required, different tasks can now be implemented on a single FPGA, reducing power consumption and board dimensions. In addition, PDR reduces reconfiguration times due to the fact that only small modification are made to the bitstream over time and the entire design does not need to be reloaded for each change.

A study into the power consumption patterns of DPSR programming was conducted by Bonamy et al [9] to investigate to what degree the sharing of silicon area between multiple accelerators will help to reduce power consumption. However, many parameters must be considered to assess whether the performance improvement outweighs preventative factors such as reconfiguration overhead, accelerator area and idle power consumption and as such any gain can be difficult to evaluate. Their results show complex variations in power usage at different stages during reconfiguration that is dependent on factors like the previous configuration and the contents of the configured circuit. In response to these experiments, three power models are proposed to help analyse the trade-off between implementing tasks as dynamically reconfigurable, in static configuration or in full software execution.

Despite clear benefits, several disadvantages become apparent with this form of reconfigurable technology. As was shown above, the power consumption overhead associated with programming new circuits can effectively imposed a minimum size or usage time on circuits for implementation to be validated. In addition, a baseline power and area cost is also always created due to the large static region which continuously consumes power and can contain unnecessary hardware. Finally, the FPGA interconnect reduces the speed and increases the power consumption of the circuit compared to an ASIC implementation because of an increased gate count required to give the system flexibility.

2) Composable and Partitionable Architectures: Partitioning and composition are techniques employed by some dynamically reconfigurable systems to provide adaptive parallel granularity [39]. Composition involves synthesising a larger logical processor from smaller processing elements when higher performance computation or greater instruction or thread level parallelism (ILP or TLP) is required. Partitioning on the other hand will divide up a large design in the most appropriate way and assign shared hardware resources to individual cores to meet the needs of an application.

Composable Lightweight Processors (CLP) is an example of a flexible architectural approach to designing a Chip Multiprocessor (CMP) where low-power processor cores can be aggregated together dynamically to form larger single-threaded processors [39]. The system has an advantage over other reconfigurable techniques in that there are no monolithic structure spanning the cores which instead communicate using a microarchitectural protocol. In tests against a fixed-granularity processor, the CLP has been shown to provide a

42% performance improvement whilst being on average 3.4 times as area efficient and 2 times as power efficient.

Core Fusion is a similar technique to CLP in that it allows multiple processors to be dynamically allocated to a single instruction window and operated as if there were one larger processor [40]. The main difference from CLP is that Core Fusion operates on conventional RISC or CISC ISAs giving it an advantage over CLP in terms of compatibility. However, this also requires that the standard structures in these ISAs are present and so can limit the scalability of the architecture.

3) Coarse Grained Reconfigurable Array Architectures: Coarse-Grained Reconfigurable Array (CGRA) architectures represent an important class of programmable system that act as an intermediate state between fixed general purpose processors and fine-grain reconfigurable FPGAs. They are designed to be reconfigurable at a module or block level rather than at the gate level in order to trade-off flexibility for reduced reconfiguration time [41].

One example of a CGRA designed with energy efficiency as the priority is the Ultra Low Power Samsung Reconfigurable Processor (ULP-SRP) presented by Changmoo et al [42]. Intended for biomedical applications as a mobile healthcare solution, the ULP-SRP is a variation of the ADRES processor [43] and uses 3 run-time switch-able power modes and automatic power gating to optimise the energy consumption of the device. Experimental results when running a low power monitoring application show a 46.1% energy consumption reduction compared to previous works.

III. Case Study - picoMIPS

In this section, we present the picoMIPS architecture as an example of a minimal and energy efficient processor implementation. The key points of the architecture will be described and evaluated, showing how it is a novel concept in minimal architecture synthesis. Developments are proposed to the architecture, that can improve performance and maintain energy efficiency, using the technologies described in the previous section.

The picoMIPS architecture is foremost a RISC microprocessor with a minimised instruction set architecture (ISA). Each implementation will contain only the necessary datapath elements in order to maximise area efficiency as the priority. For example, the instruction decoder will only recognise instructions that the user specifies and the ALU will only perform the required logic or arithmetic functions. Due to the proportionality between logic gate count and power consumption, energy efficiency is also maximised in the processor therefore the system is designed to perform a specific task in the most efficient processor-based form.

By synthesising the picoMIPS as a microprocessor, a baseline configuration is established upon which functionality can be added or removed, in the form of instructions or functions, while incurring only minimal changes to the area consumption of the design. If the task was implemented as a specific dedicated hardware circuit, any changes to the functionality could have a large influence on the area consumption of the design. Figure 4 shows an example configuration for the picoMIPS which can accommodate the majority of the simple RISC instructions. It is a Harvard architecture, with separate program and data memories, although the designer may choose to exclude a data memory entirely. The user can also specify the widths of each data bus to avoid unnecessary opcode bits from wasting logic gates.

The principles of the picoMIPS processor have been implemented in a few projects to demonstrate the concept of minimal architecture synthesis and how it can be used to produce an application specific, energy efficient processor. The discrete cosine transform (DCT) algorithm, a stage in JPEG compression, was synthesised into a processor architecture based on the picoMIPS concept. The resulting processor was more area efficient than a GPP due to the removal of unnecessary circuitry however its functionality was also reduced to performing only those functions which appear in the DCT algorithm. The processor can also be compared to a dedicated ASIC hardware implementation of the DCT algorithm. An ASIC implementations have a much higher performance and throughput of data however this is at the cost of area and energy efficiency. The picoMIPS therefore represents a balance between the two, sacrificing some performance for area and energy efficiency benefits.

The picoMIPS has also been implemented to perform the DCT and inverse DCT (IDCT) in a multi-core context [44]. A homogeneous architecture was deployed with the same single core structure, as in figure 4, being replicated 3 times. The cores are connected via a data bus to a distribution module as shown in figure 5 where block data is transferred to each core in turn. This structure theoretically triples the throughput of the system as it can process multiple data blocks in parallel.

As a microprocessor architecture, the picoMIPS can implement many of the technologies discussed in section II in order to improve energy efficiency. Clock gating, power domains and DVFS will all benefit the system however the area overhead of implementing them must first be considered as necessary. Pipeline balancing and caching can be integrated into more complex picoMIPS architectures however these are performance focused improvements and so are not priorities in the picoMIPS concept. The expansion of the system to multi-core is also one that can be employed to improve performance. Moreover, a heterogeneous architecture could be implemented to allow the picoMIPS to process multiple different applications simultaneously using several tailored ISAs. Reconfigurability can also be applied to picoMIPS to create an architecture which can be specific to each application that is executed, effectively creating a general purpose yet application specific processor. This property would require run-time synthesis algorithms to detect and develop the instructions and functional units that are required, before executing the application.

IV. Conclusion

A new concept of variable-architecture application-specific approach to embedded processor design has been presented. Over the past few decades, the trend in processor architectures, has evolved from single core to homogeneous multi-core and

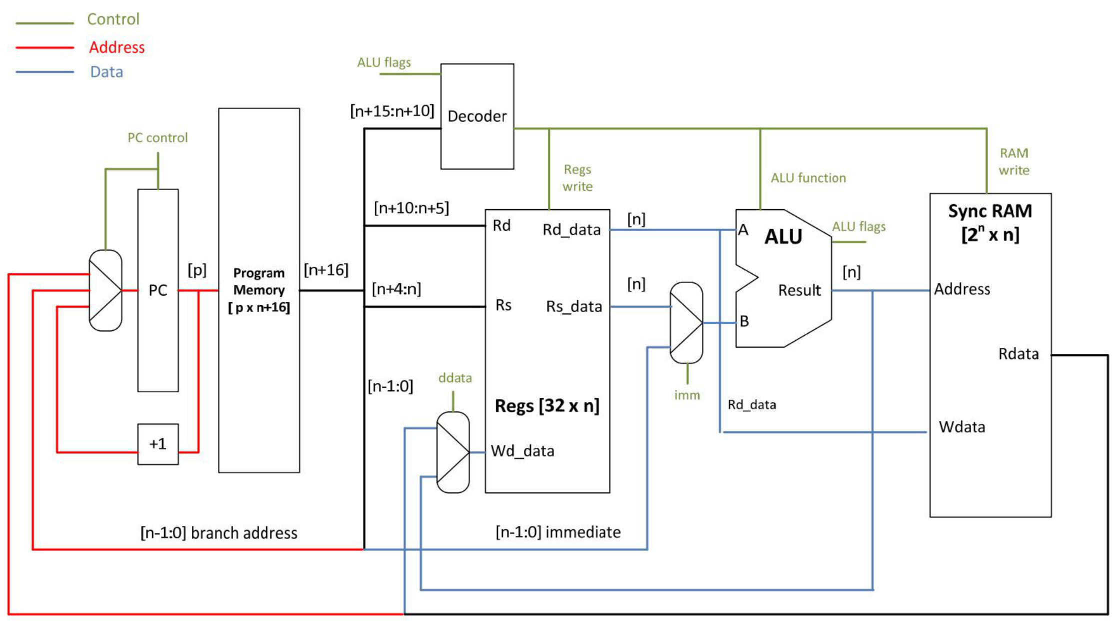

Fig. 4. An example implementation of the picoMIPS architecture.

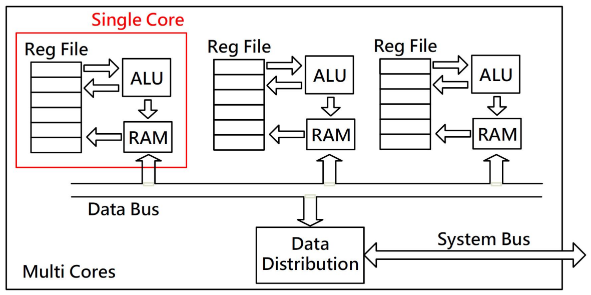

Fig. 5. A Multi-core implementation of the picoMIPS architecture, reprinted from [44].

into heterogeneous multi-core at the present. The proposed approach lends itself easily to a multi or many core architecture design where the performance is improved by enabling the simultaneous execution of threads independently on each core. The basic core design, of a core datapath and accelerators, may be replicated many times and integrated with some form of interconnect. This may form a homogeneous many-core processor, as all the cores are identical when no accelerators are connect. However, the variable-architecture processor is classed as a heterogeneous many-core processor due to the ability of run-time reconfiguration to make each core specific to the particular application that is currently executing on it during normal operation. Moreover, per-core DVFS controls can further differentiate each core using fine grain voltage and frequency adjustments that will affect the power consumption and performance of the core.

In addition to per-core DVFS control, an even finer granularity of control could be permitted through the use of per-accelerator DVFS controls. This links in with the per-accelerator power domains feature which is the first step towards DVFS control. This allows a range of operating modes for each accelerator to allow fine tuning of the systems energy consumption. Accelerators which feature in critical paths of the architecture could be run at higher DVFS levels in order to reduce their latency.

The core level in a many-core system is the smallest duplicative region of the design featuring individual cores that contain their own core datapath and application specific accelerators. An intermediate layer of shared accelerators is implemented to allow neighbouring cores to share accelerators should they required additional hardware support. This approach is similar to core morphing, where cores are weak and strong in the execution of different instruction types. These levels can also form the basis for power domains so that a multi-level power control system can be implemented to allow fine grain control of the power consumption of the chip.

ACKNOWLEDGMENT

This work was supported by the Engineering and Physical Sciences Research Council (EPSRC), UK under grant number EP/K034448/1 ” PRiME: Power-efficient, Reliable, Many-core Embedded systems” website: (*http://www.prime-project.org/*).

REFERENCES

[1] P. Greenhalgh, “big. LITTLE processing with ARM Cortex-A15 & Cortex-A7,” *ARM White Paper*, 2011.
[2] P. Pillai and K. G. Shin, “Real-time dynamic voltage scaling for low-power embedded operating systems,” *ACM SIGOPS Operating Systems Review*, vol. 35, no. 5, pp. 89–102, 2001.
[3] C. Isci, A. Buyuktosunoglu, C.-Y. Chen, P. Bose, and M. Martonosi, “An Analysis of Efficient Multi-Core Global Power Management Policies: Maximizing Performance for a Given Power Budget,” in *Microarchitecture, 2006. MICRO-39. 39th Annual IEEE/ACM International Symposium on*, 2006, pp. 347–358.
[4] V. Hanumaiah and S. Vrudhula, “Energy-efficient Operation of Multicore Processors by DVFS, Task Migration and Active Cooling,” *Computers, IEEE Transactions on*, vol. 63, no. 2, pp. 349–360, 2012.

[5] R. Rodrigues, A. Annamalai, I. Koren, S. Kundu, and O. Khan, "Performance Per Watt Benefits of Dynamic Core Morphing in Asymmetric Multicores," in *Parallel Architectures and Compilation Techniques (PACT), 2011 International Conference on*, 2011, pp. 121–130.

[6] R. Kumar, D. Tullsen, P. Ranganathan, N. Jouppi, and K. Farkas, "Single-ISA heterogeneous multi-core architectures for multithreaded workload performance," in *Computer Architecture, 2004. Proceedings. 31st Annual International Symposium on*, 2004, pp. 64–75.

[7] M. Awan and S. Petters, "Energy-aware partitioning of tasks onto a heterogeneous multi-core platform," in *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2013 IEEE 19th*, 2013, pp. 205–214.

[8] R. Basmadjian and H. De Meer, "Evaluating and modeling power consumption of multi-core processors," in *Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy), 2012 Third International Conference on*, 2012, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6221107

[9] R. Bonamy, D. Chillet, S. Bilavarn, and O. Sentieys, "Power consumption model for partial and dynamic reconfiguration," in *Reconfigurable Computing and FPGAs (ReConFig), 2012 International Conference on*, 2012, pp. 1–8.

[10] S. Saha, J. Deogun, and Y. Lu, "Adaptive energy-efficient task partitioning for heterogeneous multi-core multiprocessor real-time systems," in *High Performance Computing and Simulation (HPCS), 2012 International Conference on*, 2012, pp. 147–153.

[11] D. . Woo and H.-H. Lee, "Extending Amdahl's Law for Energy-Efficient Computing in the Many-Core Era," *Computer*, vol. 41, no. 12, pp. 24–31, 2008.

[12] B. de Abreu Silva and V. Bonato, "Power/performance optimization in FPGA-based asymmetric multi-core systems," in *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on*, 2012, pp. 473–474.

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

[14] B. de Abreu Silva, L. Cuminato, and V. Bonato, "Reducing the overall cache miss rate using different cache sizes for Heterogeneous Multi-core Processors," in *Reconfigurable Computing and FPGAs (ReConFig), 2012 International Conference on*, 2012, pp. 1–6.

[15] Q. Cai, J. Gonzalez, G. Magklis, P. Chaparro, and A. Gonzalez, "Thread shuffling: Combining DVFS and thread migration to reduce energy consumptions for multi-core systems," in *Low Power Electronics and Design (ISLPED) 2011 International Symposium on*, 2011, pp. 379–384.

[16] P. Bassett and M. Saint-Laurent, "Energy efficient design techniques for a digital signal processor," in *IC Design Technology (ICICDT), 2012 IEEE International Conference on*, 2012, pp. 1–4.

[17] ARM, *ARM Cortex-A15 MPCore Processor Technical Reference Manual*, ARM, June 2013, pages 53 - 63.

[18] A. Sinkar, H. Ghasemi, M. Schulte, U. Karpuzcu, and N. Kim, "Low-Cost Per-Core Voltage Domain Support for Power-Constrained High-Performance Processors," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 22, no. 4, pp. 747–758, 2013.

[19] H. Ghasemi, A. Sinkar, M. Schulte, and N. S. Kim, "Cost-effective power delivery to support per-core voltage domains for power-constrained processors," in *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, 2012, pp. 56–61.

[20] E. Rotem, A. Mendelson, R. Ginosar, and U. Weiser, "Multiple clock and Voltage Domains for chip multi processors," in *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*, 2009, pp. 459–468. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5375435

[21] R. Bahar and S. Manne, "Power and energy reduction via pipeline balancing," in *Computer Architecture, 2001. Proceedings. 28th Annual International Symposium on*, 2001, pp. 218–229.

[22] J. Sartori, B. Ahrens, and R. Kumar, "Power balanced pipelines," in *High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on*, 2012, pp. 1–12.

[23] R. Kumar, V. Zyuban, and D. Tullsen, "Interconnections in multi-core architectures: understanding mechanisms, overheads and scaling," in *Computer Architecture, 2005. ISCA '05. Proceedings. 32nd International Symposium on*, 2005, pp. 408–419.

[24] H. Zeng, J. Wang, G. Zhang, and W. Hu, "An interconnect-aware power efficient cache coherence protocol for CMPs," in *Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on*, 2008, pp. 1–11.

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

[26] P. Greenhalgh, "big.LITTLE Processing with ARM Cortex-A15 & Cortex-A7," ARM, Tech. Rep., September 2011.

[27] H. M. Waidyasooriya, Y. Takei, M. Hariyama, and M. Kameyama, "FPGA implementation of heterogeneous multicore platform with SIMD/MIMD custom accelerators," in *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, 2012, pp. 1339–1342.

[28] A. Lukefahr, S. Padmanabha, R. Das, F. Sleiman, R. Dreslinski, T. Wenisch, and S. Mahlke, "Composite Cores: Pushing Heterogeneity Into a Core," in *Microarchitecture (MICRO), 2012 45th Annual IEEE/ACM International Symposium on*, 2012, pp. 317–328.

[29] B. Jeff, "Advances in big.LITTLE Technology for Power and Energy Savings," ARM, Tech. Rep., September 2012.

[30] S. Zhang and K. Chatha, "Automated techniques for energy efficient scheduling on homogeneous and heterogeneous chip multi-processor architectures," in *Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific*, 2008, pp. 61–66.

[31] A. Z. Jooya and M. Analoui, "Program phase detection in heterogeneous multi-core processors," in *Computer Conference, 2009. CSICC 2009. 14th International CSI*, 2009, pp. 219–224.

[32] L. Sawalha and R. Barnes, "Energy-Efficient Phase-Aware Scheduling for Heterogeneous Multicore Processors," in *Green Technologies Conference, 2012 IEEE*, 2012, pp. 1–6.

[33] F. Calcado, S. Louise, V. David, and A. Merigot, "Efficient Use of Processing Cores on Heterogeneous Multicore Architecture," in *Complex, Intelligent and Software Intensive Systems, 2009. CISIS '09. International Conference on*, 2009, pp. 669–674.

[34] M. Pericas, A. Cristal, F. Cazorla, R. Gonzalez, D. Jimenez, and M. Valero, "A Flexible Heterogeneous Multi-Core Architecture," in *Parallel Architecture and Compilation Techniques, 2007. PACT 2007. 16th International Conference on*, 2007, pp. 13–24.

[35] M. Santambrogio, "From Reconfigurable Architectures to Self-Adaptive Autonomic Systems," in *Computational Science and Engineering, 2009. CSE '09. International Conference on*, vol. 2, 2009, pp. 926–931.

[36] J. Zalke and S. Pandey, "Dynamic Partial Reconfigurable Embedded System to Achieve Hardware Flexibility Using 8051 Based RTOS on Xilinx FPGA," in *Advances in Computing, Control, Telecommunication Technologies, 2009. ACT '09. International Conference on*, 2009, pp. 684–686.

[37] S. Bhandari, S. Subbaraman, S. Pujari, F. Cancare, F. Bruschi, M. Santambrogio, and P. Grassi, "High Speed Dynamic Partial Reconfiguration for Real Time Multimedia Signal Processing," in *Digital System Design (DSD), 2012 15th Euromicro Conference on*, 2012, pp. 319–326.

[38] S. Liu, R. Pittman, A. Forin, and J.-L. Gaudiot, "On energy efficiency of reconfigurable systems with run-time partial reconfiguration," in *Application-specific Systems Architectures and Processors (ASAP), 2010 21st IEEE International Conference on*, 2010, pp. 265–272.

[39] K. Changkyu, S. Sethumadhavan, M. S. Govindan, N. Ranganathan, D. Gulati, D. Burger, and S. Keckler, "Composable Lightweight Processors," in *Microarchitecture, 2007. MICRO 2007. 40th Annual IEEE/ACM International Symposium on*, 2007, pp. 381–394.

[40] E. Ipek, M. Kirman, N. Kirman, and J. F. Martinez, "Core fusion: accommodating software diversity in chip multiprocessors," in *Proceedings of the 34th annual international symposium on Computer architecture*, ser. ISCA '07. New York, NY, USA: ACM, 2007, pp. 186–197. [Online]. Available: http://doi.acm.org/10.1145/1250662.1250686

[41] Z. Rakossy, T. Naphade, and A. Chattopadhyay, "Design and analysis of layered coarse-grained reconfigurable architecture," in *Reconfigurable Computing and FPGAs (ReConFig), 2012 International Conference on*, 2012, pp. 1–6.

[42] K. Changmoo, C. Mookyoung, C. Yeongon, M. Konijnenburg, R. Soojung, and K. Jeongwook, "ULP-SRP: Ultra low power Samsung Reconfigurable Processor for biomedical applications," in *Field-Programmable Technology (FPT), 2012 International Conference on*, 2012, pp. 329–334.

[43] F. J. Veredas, M. Scheppler, W. Moffat, and B. Mei, "Custom implementation of the coarse-grained reconfigurable ADRES architecture for multimedia purposes," in *Field Programmable Logic and Applications, 2005. International Conference on*, 2005, pp. 106–111.

[44] G. Liu, "Fpga implementation of 2d-dct/idct algorithm using multi-core picomips," Master's thesis, University of Southampton, School of Electronics and Computer Science, September 2013.