ABSTRACT

Emerging technologies provide SoCs with fine-grained DVFS capabilities in both space (number of domains) and time (transients in the order of tens of nanoseconds). Analyzing these systems requires cycle-accurate accounting of rapidly-changing dynamics and complex interactions among accelerators, interconnect, memory, and OS. We present an FPGA-based infrastructure that facilitates such analyses for high-performance embedded systems. We show how our infrastructure can be used to first generate SoCs with loosely-coupled accelerators, and then perform design-space exploration considering several DVFS policies under full-system workload scenarios, sweeping spatial and temporal domain granularity.

1. INTRODUCTION

Dynamic Voltage-Frequency Scaling (DVFS) is a consolidated technique to help meeting the increasing requirements for energy-efficient computing by adjusting the operating point of a system to its changing workloads [24, 25]. Modern multi-core processors adjust independently the operating points of their various subsystems (subsets of cores, L3 caches, interconnect...) through the use of multiple voltage/frequency (VF) domains [22]. Increasing the number of these VF domains helps achieving performance and power targets by enabling a more fine-grained application of DVFS. The trend is growing across many classes of integrated circuits, from server processors to systems-on-chip (SoC) for embedded applications. According to a global survey performed by Synopsys among its customers in 2011, about 30% of the respondents used more than 10 clock domains and between 4 and 10 voltage domains in their designs [33].

Meanwhile, the efficiency benefits of hardware specialization [14] is favoring the rise of heterogeneous multi-core architectures that combine processor cores with special-function accelerators [6, 30]. Accelerators mitigate the challenges of dark silicon [12, 28] by executing a specific task faster and more efficiently than software. As the number of processors and accelerators integrated on the same chip keeps growing, the opportunities for power and energy savings with DVFS also increase. Intuitively, the biggest savings would come from the ability of controlling each component independently and promptly. In response to such needs, the development of integrated voltage regulators (IVRs) has become the focus of many research works [1, 7, 27, 29].

Typical IVRs are switching regulators that store energy in capacitors and/or inductors and deliver that energy at a potential controlled by a switching signal. Die-integrated switched-capacitor regulators boast fast transient response, high peak efficiency (>90%), and minimal technology requirements. Efforts have been made to preserve such efficiency while delivering sufficient power densities [1, 8]. Switched-inductor regulators, on the other hand, naturally support higher power densities. Indeed, switched-inductor regulators are used in most discrete, board-level power management infrastructures and can be designed to have >90% efficiency. However, they remain difficult to build into die-integrated systems due to the low quality factor and correspondingly low efficiency available in die-integrable inductor technologies [31]. Still, efficiency factors in the 70%-85% range have been reached with state-of-the-art inductor technology [11, 27, 29].

Thanks to this technology progress, IVRs hold the promise of enabling an unprecedented degree of fine-grained power management both in space (multiple distinct voltage domains) and in time (with transient responses in the order of nanoseconds). For example, coupling a dedicated IVR to each accelerator would make it possible to rapidly activate/deactivate it (and adapt its VF operating point) independently from the other components of the SoC. The benefits in terms of energy-efficient performance could be very large. Designers of embedded SoCs need practical solutions to explore and exploit these new DVFS capabilities.

The main contribution of this work is an infrastructure to analyze the impact of fine-grained DVFS. We consider a combination of software and hardware mechanisms that benefit the efficiency of high-performance embedded applications running on heterogeneous SoCs. These SoCs integrate many hardware accelerators that efficiently execute complex tasks on large data sets. Analyzing these scenarios requires accounting of cycle-accurate dynamics and the complex interactions of the accelerators with the interconnect, the external memory, the processors and the operating system. Simulation is inadequate to satisfy such requirements [3]; hence, we propose an FPGA-based infrastructure for heterogeneous SoCs supporting multiple independent VF domains. Its flexibility allows us to quickly build SoC designs, combining different kinds of accelerators, each coupled with an instance of a dedicated hardware controller that can enforce a local DVFS policy.

In summary, our infrastructure enables pre-silicon tuning and design exploration of DVFS policies. By applying it to three SoC case studies, we show how fine spatial granularity, coupled with the hardware controllers, is key to achieve energy savings up to 85% in the accelerator operations. Furthermore, we show that the benefits of temporal granularity are highly dependent on the application and the degree of contention for shared resources. Across our experiments, we observe an overall energy reduction of up to 75% combined with a 10% performance improvement.

2. EMULATION INFRASTRUCTURE

As the number of specialized accelerators grows on embedded SoCs, it is critical to introduce some sort of regularity to keep the SoC design and testing process manageable. A right balance is represented by a tile-based architecture where tiles can host processor cores or accelerators, interconnected by a Network-on-Chip (NoC). While all tiles are not necessarily required to have the same size, this helps the physical design process and can be achieved by combining smaller accelerators within a tile. Thanks to its modularity, an NoC-based tile-architecture is known to efficiently reduce the complexity of SoC physical design (place and route, clock distribution, power grid layout) [2, 9]. Moreover, an NoC offers a
natural synchronization barrier among clock domains. Dual-clock FIFO buffers, placed between routers' local ports and the tiles preserve transfer-bursts throughput and prevent data loss across clock domains [26] through a back-pressure mechanism. Indeed, NoC-based architectures have already been implemented in chip-prototypes together with voltage regulators [23].

Following these principles, we designed a flexible SoC architecture which is at the base of our emulation infrastructure. The top of Fig. 1 shows an instance of this architecture that is based on a 4×4 2D-Mesh NoC. Any instance of our heterogeneous SoC integrates four types of tiles: CPU tiles contain a general purpose processor, ACC tiles host distinct accelerators, the I/O tile interfaces Ethernet, UART and JTAG for system monitoring, and each MEM tile hosts a memory controller, accessing a distinct DDR channel. In this example, twelve accelerators are distributed across four VF domains, (labeled D0 to D3 and enclosed by the dotted lines). A voltage regulator is associated with each VF domain to control the supply voltage of its components. When all the components of a domain are inactive, they can be turned off. Only one tile per domain (colored in light green) is equipped with a DVFS controller, including a Phase-Locked Loop (PLL), which is shared within its domain.

**Accelerator Tile.** The bottom of Fig. 1 shows the block diagram of an accelerator tile with the DVFS controller for its VF domain. The accelerator is composed of multiple hardware blocks that interact through a local private memory. The control interface (config) exposes the accelerator’s configuration parameters to the operating system as memory-mapped registers to be set by the device driver. Data, instead are exchange between DDR and the private memory through direct memory access (DMA). The DMA controller (DMAC) translates the accelerator’s read/write requests into NoC packets. Transfers are initiated directly by the accelerators, specifically by two dedicated “communication blocks”: Read and Write. Typically, in an efficient accelerator design, computation and communication are well balanced and run concurrently, injecting/ejecting up to one flit per cycle per direction. For designs with many accelerators, NoC congestion naturally leads computation stage to stall waiting for the arrival of new data. This condition offers opportunities to exploit fine-grained DVFS, which can reduce the wasting of supply power while contributing also to alleviate resource contention. To dynamically detect these situations, a tile is equipped with probes for the accelerator activity (shaded gray circles in Fig. 1). These probes detect: (i) whether the accelerator is enabled; (ii) if it is computing, transferring data, or both; and (iii) if the tile is receiving back-pressure from the NoC, either due to congestion or temporarily unavailable access to main memory. A probe to monitor DVFS behavior (red circle in Fig. 1) is also present in each domain. Information from probes is recorded with performance counters and exposed to an external Ethernet interface for profiling purposes, while the DVFS controller processes it in order to apply the desired power management policy.

**Configurable Fine-grained DVFS Controller.** Fig. 2 shows the components of our controller. The central block is a finite state machine (FSM) responsible for regulating voltage and frequency for its local domain. It drives the signal vctrl1, that translates into a voltage reference for a generic IVR. Additionally, it dynamically reconfigures the PLL control logic, which may vary depending on the particular PLL implementation. Note the need for synchronization flip flops on the paths between the DVFS controller and the PLL state machine. The latter, in fact, must be clocked by the external reference clock (refclk), to make sure we keep driving appropriately the PLL configuration pins while its output frequency is transitioning from one operation point to the other. Fig. 2 also shows the feedback compensation clock fb, required to obtain a correct phase locking, and the clock buffers, represented as triangles along clocks’ paths, which are the entry points to the clock distribution network.

A set of memory-mapped registers, represented in the lower portion of Fig. 2, allows reconfiguration from software, which can override the local decisions in favor of a new system-level policy. The policy actuation is based on the information provided by performance counters, which are incremented every time a specific condition holds within the context of the local VF domain. For example, Fig. 2 shows two counters: one is incremented if an accelerator in the domain is idle while the other counts the number of cycles in which back-pressure is applied at any network interface of the domain.

The clock-gating logic in the lower-right part of Fig. 2 completes the DVFS logic. Gating is activated on any transition to preserve functional correctness of the accelerator. The DVFS FSM, instead, is designed to be robust to the transition of the clock frequency, as freezing this logic would lead to a deadlock condition. During the actual VF transitions a watchdog is set to go off after the transient time of the VR. In addition, when both frequency and voltage have been updated, a configurable timeout is set to allow a sweep of the temporal granularity. Thanks to the request-acknowledge protocol, the PLL transient doesn’t need to be timed. In our emulation in-
Fine-grained DVFS policies. To manage VF domains on our infrastructure, we provide a HW/SW interface to enforce configurable policies on each domain. Configurability is key to enable a design space exploration of the power-management options on embedded systems driven by the target applications.

Policy “none” (PN) maintains a specified VF operating point.

Policy “traffic” (PT) is based on the observation of back-pressure signals at the interface between a tile and the interconnect, which is a measure of the current resource contention. The controller initiates a VF transition based on a user-defined threshold.

Policy “burst” (PB) combines the observation of the traffic with computation over communication ratio of accelerators. Both parameters are compared to software-specified thresholds.

Policy “limit” (PL) can be activated in combination with any of the other policies to ensure a fair distribution of the power envelop across multiple accelerators. It consists in a DVFS supervisor daemon that scans the system with a configurable period and prevents all accelerators from running at maximum speed and power dissipation at the same time, according to a specified aggregated power cap. Priority among VF domains is rotated following a round-robin scheme.

The flow chart in Fig. 3 provides an overview of the interaction of the policies with the DVFS actuation logic. If the controller ends in the state “step-down” or “step-up”, then a VF transition is initiated. By sweeping the configuration parameters for these policies we defined twenty-five different settings (Table 1).

3. ENERGY ESTIMATION FLOW

Frequency Scaling on FPGA. Modern FPGAs feature several clocking resources which are typically required to support a wide variety of I/O protocols. We target the Xilinx Virtex-7 XC7V2000T FPGA, which allows us to place up to 24 PLLs. Run-time reconfiguration of the PLLs, however, incurs high latency. Hence we define the frequency division factors at synthesis time and then select the clock line at run-time. To map the PLL control logic shown in Fig. 2 on FPGA, we instantiate a glitch-free clock multiplexer and the clock buffers. A 2:1 clock MUX is available as a primitive black-box in the FPGA components library. It allows the circuit to switch between two clocks with the guarantee that the period from one rising edge to the other will always be at least as large as the period of the slower clock and no glitch can occur. A clock buffer is automatically placed at the output of the MUX. Switching among four clocks, however, would require a tree of such multiplexers, with consequent waste of global clock buffers. Therefore, we designed a 4:1 clock MUX with gating logic and only one global buffer. An optional buffer can be placed before the gating logic to help timing closure, but this reduces the number of independent domains supported.

By replicating this block, we are able to implement an SoC with up to twelve domains in our target FPGA where the DVFS controller can correctly emulate the frequency scaling. Thanks to a user-guided placement of the clock buffers, the design closes at 100MHz as the fastest frequency. Other PLL frequencies are accordingly set to match the ratios for the different operating points.

Accelerator Characterization. Accurate emulation of frequency scaling, combined with a fast Ethernet interface to the SoC probes allows us to monitor run-time statistics, including the number of cycles $C_i$ spent in each operating point. To determine the energy dissipated when accelerators are running, we combine this information with data on static and dynamic power obtained from an RTL power-estimation flow. First, we synthesize the RTL of the accelerators for a standard-cell technology and perform power estimation with switching activity back-annotation. For this, we use a commercial 32nm CMOS library with nominal voltage equal to 1V and we target a frequency of 1GHz. Then, we compute the energy consumption of each accelerator as: $E = \sum_{i=0}^{n} E^i + C_i$, where $N$ represents the number of operating points and $E^i$ represent the energy consumption per clock cycle. The total energy is then obtained by aggregating the energy consumption of all accelerators. The values for $E^i$ are obtained by first re-characterizing the standard-cell and the SRAM libraries with detailed SPICE-level simulations, and then repeating the power analysis for every selected operating point. Timing analysis is also performed to verify that the circuit meets all constraints. With this flow, we designed, and characterized across four operating points, 17 accelerators for various computational kernels from the PERFECT Benchmark Suite [4], listed in Table 2. The 0.1V step for voltage scaling allows the regulator to achieve high power conversion efficiency (~90%) [18]. Such step is decreased to 0.05V for the slower operating point due to additional constraints imposed by the SRAM libraries.
4. FULL-SYSTEM CASE STUDIES

By combining these heterogeneous and high-performance accelerators, we designed multiple SoC instances. The goal is to showcase how the described hardware/software FPGA infrastructure can be effectively used to analyze the impact of fine-grained power management on embedded systems. We present three case studies of accelerator-based SoCs. Each of these is generated by plugg-}
Figure 8: Design WAMI-APP: Normalized delay and energy savings across different VF domains and DVFS policies.

Figure 9: Design WAMI-APP: energy breakdown over time for the policies pn0 (left), pt14 (center) and pt14 with PL (right).

the temporal granularity for PT and PB. The reason is that the specific data-transfer pattern of each accelerator directly affects the statistics measured by the DVFS controller. When temporal granularity and policies thresholds are not properly configured for the accelerator’s specific traffic signature, both energy savings and delay are penalized. On the other hand, there is a clear trend as we increase spatial granularity: more VF domains yield usually a delay improvement and always considerable (more than 50%) energy savings; this is the case even if DVFS is not used (PN policy).

Across all policies, pb24 delivers the best delay improvement (10× less than pn0) while pb25, combined with the supervisor daemon, achieves the largest energy saving, consuming about 15% of the baseline energy. To understand better how the combination of the local fine-grained hardware policy and the software supervisor can achieve this result, we can look at Fig. 5: each of these charts show the aggregated energy that is spent over time for the execution of an experiment with a particular policy. Each colored area breaks the energy-delay product into single-accelerator contributions. The units on the horizontal axis are probing time frames of 16.7 ms, which is the time allowed to the Ethernet interface to collect statistics from all probes. If we look at the chart on the left (policy pn0) we notice that all accelerators dissipate almost the same amount of energy at every time frame, until completion. Conversely, the central figure shows how the DVFS controller under policy pb25 modulates the energy dissipation during the execution. Interestingly, however, the variation of the energy over time is very similar across most of the domains: note that the thickness of the filled lines remains visibly constant over time for each accelerator until completion. This scenario suggests that the decisions of a DVFS controller, based on the traffic at the local interconnect, may be sub-optimal if taken simultaneously by all other controllers in the system. Finally, the right chart of Fig. 5 confirms the benefits of activating policy PL. All areas shrank considerably, leading to a major decrease of the energy-delay product. The unbalanced bias that the daemon gives to the DVFS controllers reduces the interference across the accelerators’ traffic patterns. Therefore, the accelerators spend less time dissipating power while waiting for a transaction to complete. The result is an energy savings of more than 50% with respect to the same policy with no software supervisor.

TWELVE FFT2D: homogeneous accelerators. In the second case study, an accelerator for the ubiquitous FFT2D kernel is replicated twelve times. By comparing the results of Fig. 6 and Fig. 7 with the corresponding ones for the previous case study, we notice much less variation across the experiment runs as we sweep either the temporal or spatial granularity. In particular, if we don’t consider the black bars that correspond to the activation of PL, these runs show similar energy savings for the cases of two, four or twelve domains. In order to understand this behavior, we measured the NoC injection rate and the traffic at the two memory controllers tiles: as soon as more than two FFT2D accelerators are activated, the queues at the memory tiles interfaces get quickly filled up and all tiles start receiving back-pressure from the NoC. This condition of extremely high congestion forces all regulators to slow down, thus giving more slack to the DDR and the NoC to complete the pending transactions. As soon as traffic decreases below the configured threshold, however, all accelerators tend to speed up again, thus bringing back the congestion. Such cyclic behavior is confirmed by the comparison between the chart on the left and the one in the middle of Fig. 7. Policy pt14 is only adding noise to the energy distribution over time as the slightly visible ripple suggests. On the other hand, the activation of PL brings more than 50% extra energy savings and the twelve FFT2D accelerators complete their execution consuming 38% of the baseline energy.

WAMI-APP: accelerators with data dependencies. Complex embedded SoC applications, are usually implemented as the composition of many interacting accelerators with data-dependency relations: e.g. one accelerator produces input data for another accelerators, which can start executing only after the first terminates. We use our infrastructure to analyze the implications of such dependencies with the WAMI-APP case study. For this image-processing application, data-dependency relations apply to a single input frame, while multiple frames can be processes in a pipelined fashion, allowing more accelerators to execute in parallel. To exploit such parallelism, we wrote a multi-thread program, based on the standard p-threads library, where each thread controls a distinct accelerator. Note that the complexity of invoking system-calls and device drivers is hidden by our software layer.

The resulting parallelism of the WAMI-APP case study and consequent opportunities for energy savings remain somewhat limited.
due to a heavily unbalanced distribution of the workload and dissipated power. Note, from Table 2, that change-detection accounts for almost 40% of the energy spent per cycle by all accelerators.

Results in Fig. 8 show that most policies have a modest impact on energy savings, with little variations as we move towards finer temporal granularity. With respect to spatial granularity, no improvements are obtained beyond four VF domains: probes’ data show that four is the largest number of accelerators running in parallel during WAMI-APP experiments. When comparing the three charts of Fig. 9, we again observe that the shape of the energy dissipation over time visibly changes for many accelerators when PL is enabled. The overall result, however, is dominated by change detection, which not only is responsible for most of the power budget, but it is also the longest-running accelerator.

5. RELATED WORK

DFVS is a well-studied technique for reducing the power consumption of SoCs, especially in the context of processors [13, 17] or application scheduling [19, 21]. Indeed, this solution is becoming more and more relevant in SoC design to increase energy efficiency [16]. However, there is a lack of research examining the effects of voltage regulators on accelerator-based SoCs, especially when adopting fine-grained power management. There are different analytical studies on voltage regulators to analyze dynamics and overheads. Park et al. [20] propose a model to analyze the overhead of DC-DC converters, while Jevetic et al. [15] analyze switched-capacitor converters for fine-grained DFVS of many-core systems. Wang et al. present an analytical study and comparison of on-chip and off-chip regulators [32]. FPGA-based emulation has been proposed to perform power analysis on chip multiprocessors [5]. These works, however, do not evaluate the effects of the fine-grained power management on a heterogeneous SoC. We designed a full-system infrastructure to explore such effects while sweeping spatial and temporal DFVS granularity and showed its applicability to embedded systems with many accelerators.

Kim et al. [18] analyze the effects of on-chip regulators in multiprocessor systems, highlighting the importance of increasing the spatial granularity to better control the chip. Rangan et al. analyze thread motion as a technique for fine-grained power management [21]. DFVS has been applied to accelerators by Dasika et al. who adopt micro-architectural solutions (i.e. Razor flip-flops) to scale clock frequency and voltage according to the surrounding environment [10]. However, they do not apply global techniques to coordinate the scaling in case of multiple concurrent accelerators under complex full-system scenario. Kornaros and Pnevmatikatos [19] propose an FPGA emulation of DFVS, but do not consider the effects of fine-grained power management.

6. CONCLUDING REMARKS

We presented an FPGA-based emulation infrastructure that facilitates the rapid prototyping of heterogeneous SoCs, which vary in number and type of integrated accelerators. Furthermore, it enables the analysis of the impact of fine-grained DFVS, by combining frequency-scaling emulation with energy characterization data of the accelerators at different VF operating points. We demonstrated our HW/SW infrastructure through three full-system case studies, where we were able to determine which DFVS policy configuration offers higher energy savings and performance improvements through an indirect control on the interconnect traffic. In summary, our infrastructure assists designers with pre-silicon analysis, tuning and design exploration of fine-grained power management of heterogeneous embedded systems.

7. REFERENCES

[33] White, M. A. Low power is everywhere. Synopsys Insight Newsletter (online), 2012.