An FPGA-Based Infrastructure for Fine-Grained DVFS Analysis in High-Performance Embedded Systems

Paolo Mantovani[†] Emilio G. Cota[†] Kevin Tien[$]
Christian Pilato[†] Giuseppe Di Guglielmo[†] Ken Shepard[$] Luca P. Carloni[†]

[†]Department of Computer Science [$]Department of Electrical Engineering
Columbia University - New York, NY, USA
{paolo,cota,pilato,giuseppe,luca}@cs.columbia.edu {ktien,shepard}@ee.columbia.edu

ABSTRACT

Emerging technologies provide SoCs with fine-grained DVFS capabilities both in 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 well-established technique to help meeting the increasing requirements for energy-efficient computing by adjusting the operating point of a system to its changing workloads [28, 29]. Modern multi-core processors regulate independently the operating points of their various subsystems through the use of multiple voltage/frequency (VF) domains [6, 26]. 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 (SoCs) 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 [37].

Meanwhile, the efficiency benefits of hardware specialization [17] is favoring the rise of heterogeneous multi-core architectures that combine processor cores with special-function accelerators [7, 34]. Accelerators mitigate the challenges of dark silicon [15, 32] 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, 8, 31, 33]

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, 10]. *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 [35]. Still, efficiency factors in the 70%-85% range have been reached with state-of-the-art inductor technology [14, 31, 33].

Thanks to this technology progress, IVRs hold the promise of enabling an unprecedented degree of fine-grained power management both in space (with multiple distinct voltage domains) and in time (with transient responses in the order of nanoseconds). Therefore, we built an infrastructure that allows SoC designers to practically explore and exploit these new DVFS capabilities. 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 for the cycle-accurate dynamics and complex interactions of the accelerators with the interconnect, memory, processors and 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 by 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 analyze workload's power dissipation and performance sensitivity to time-space granularity of DVFS and show that our hardware controllers can save up to 85% of accelerators' energy.

2. DVFS INFRASTRUCTURE FOR SOCS

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 good balance is given by a tile-based architecture where tiles can host processor cores or accelerators, interconnected by a Network-on-Chip (NoC) [23]. While all tiles are not necessarily required to have the same size, this helps the physical design process and can be

DAC '16, June 05 - 09, 2016, Austin, TX, USA

ISBN 978-1-4503-4236-0/16/06...$15.00

DOI: http://dx.doi.org/10.1145/2897937.2897984

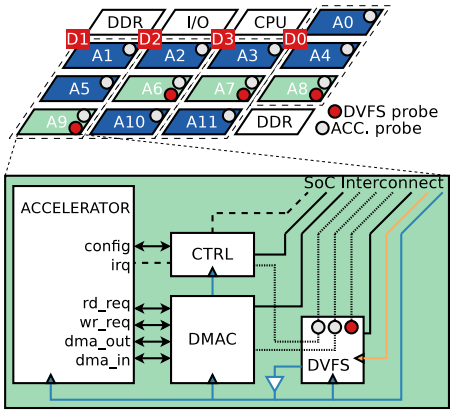

Figure 1: Top: tile-based SoC high-level view. Bottom: block diagram of an accelerator tile with DVFS controller.

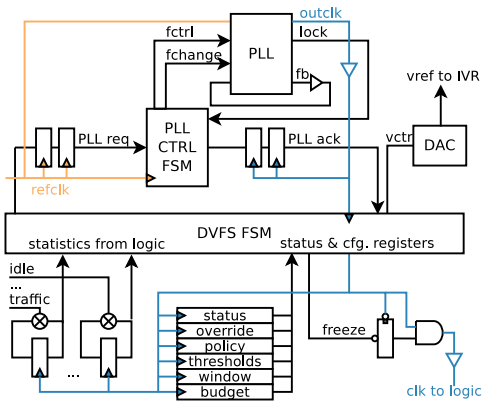

Figure 2: Simplified DVFS controller block diagram.

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 place and route, clock distribution, power grid layout, etc. [2, 12]. A packet-switched NoC with a latency-insensitive protocol naturally decouples the run-time operations of the various tiles [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 [30] through a *back-pressure* mechanism. Indeed, NoC-based architectures have already been implemented in chip prototypes together with voltage regulators [27].

Following these principles, we implemented a flexible architecture to support the analysis of fine-grained power control for heterogeneous SoCs. Any instance of our SoC integrates four types of tiles: *CPU tiles* contain general-purpose processors, *ACC tiles* host distinct accelerators, *I/O tiles* interface Ethernet, UART and JTAG for system monitoring, and each *DDR tile* hosts a memory controller, accessing a distinct DDR channel. The top of Fig. 1 shows an example of our architecture that uses a 4×4 2D-Mesh NoC and includes twelve accelerators 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 private local memory [11]. 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 exchanged between DDR and the private memory through direct memory access (DMA). The DMA controller 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 `vctrl` 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 controller. The latter, in fact, must be clocked by the external reference clock (`refclk`), to 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, shown in the lower portion of Fig. 2, allows reconfiguration from software, which can override the local decisions in favor of a 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 components in the lower-right part of Fig. 2 complete the DVFS logic and freeze accelerators during transient time. We make the DVFS FSM robust to clock frequency transitions to avoid the risk of deadlock when applying clock gating. 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. The minimum timeout is 64 cycles, which is a conservative transient time for IVRs [31].

Table 1: Policies configurations.

POLICY	OPERATING POINT	WINDOW (CYCLES)	THRESHOLDS TRAFFIC	THRESHOLDS BURST	PL
PN0	1.0V 1.0GHz	-	-	-	-
PN1	0.90V 0.9GHz	-	-	-	-
PN2	0.80V 0.8GHz	-	-	-	-
PN3	0.75V 0.6GHz	-	-	-	-
PT[4-14]	variable	131,072-64	4,096-32	-	N/Y
PB[15-25]	variable	131,072-64	4,096-32	114,688-56	N/Y

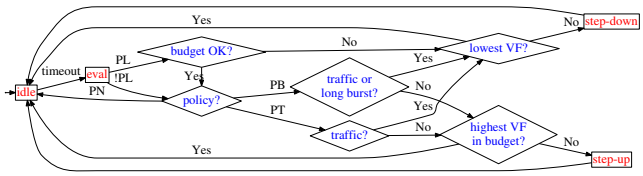

Figure 3: DVFS policies flow chart.

Fine-grained DVFS policies. The flow chart in Fig. 3 provides an overview of the four DVFS policies described below. When the controller ends either in the state "step-down" or "step-up", then a VF transition is initiated. Note that all thresholds can be tuned at run-time via software. By sweeping the configuration parameters we define twenty-five different settings, summarized in Table 1.

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 observations of the traffic and computation over communication time ratio of accelerators. Both parameters are compared to software-specified thresholds.

Policy "limit" (PL) can be activated in combination with other policies to ensure a fair distribution of the power envelope 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.

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 used a high-end Xilinx Virtex-7 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 and guarantees 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 glitch-free 4:1 clock MUX with gating logic and only one global buffer. An optional buffer can be placed before the clock-gating latch to help timing closure at the cost of supporting fewer independent VF domains. By replicating the logic of Fig. 2, we can implement an SoC with up to twelve frequency-scaling-enabled domains on the target FPGA. 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.

Table 2: Energy estimates for a 32nm CMOS technology.

	ACC.	AREA (μm^2)	ENERGY PER CLOCK CYCLE (pJ) 1.0V 1.0GHz	0.9V 0.9GHz	0.8V 0.8GHz	0.75V 0.6GHz
MIX	FFT2D	828,641	75.19	64.92	56.76	55.83
	D-FILTER	694,050	28.82	23.68	19.62	19.41
	IMAGE-WARP	362,945	36.01	29.98	25.79	25.50
	PFA-INTERP1	478,905	198.62	155.08	119.27	108.28
	PFA-INTERP2	565,417	169.91	133.16	103.07	94.33
WAMI-APP.	ADD	87,446	3.89	3.01	2.33	2.28
	GRADIENT	1,782,904	42.09	31.99	24.37	25.02
	DEBAYER	698,765	22.81	17.42	13.79	13.78
	GRAYSCALE	418,337	13.30	10.18	7.96	8.08
	HESSIAN	1,251,278	34.98	27.09	21.11	21.53
	MULT	123,329	5.89	4.57	3.59	3.58
	RESHAPE	58,939	3.09	2.39	1.87	1.80
	SD-UPDATE	1,457,124	38.11	29.42	22.76	23.36
	STEEP.-DESC.	1,595,080	39.17	29.67	22.61	23.30
	SUBTRACT	600,697	1.05	11.89	9.15	9.41
	WARP	451,177	25.57	20.00	15.71	15.37
	CHANGE-DET.	1,463,797	146.33	113.31	86.70	81.86

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 i. 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: $E = \sum_{i=0}^{N} E_i^c * C_i$, where N is the number of operating points and factors E_i^c 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^c 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 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%) [21]. 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

We present three case studies of accelerator-based SoCs mapped to the proposed FPGA infrastructure and show the effects of fine-grained power management tuning for each workload scenario. Each SoC is generated by plugging a set of accelerators in the architecture of Fig. 1 and includes one CPU tile running Linux, one I/O tile, and two DDR tiles. The three case studies differ for the number and types of accelerators: (1) MIX features ten independent heterogeneous accelerators; (2) 12-FFT2D features twelve in-

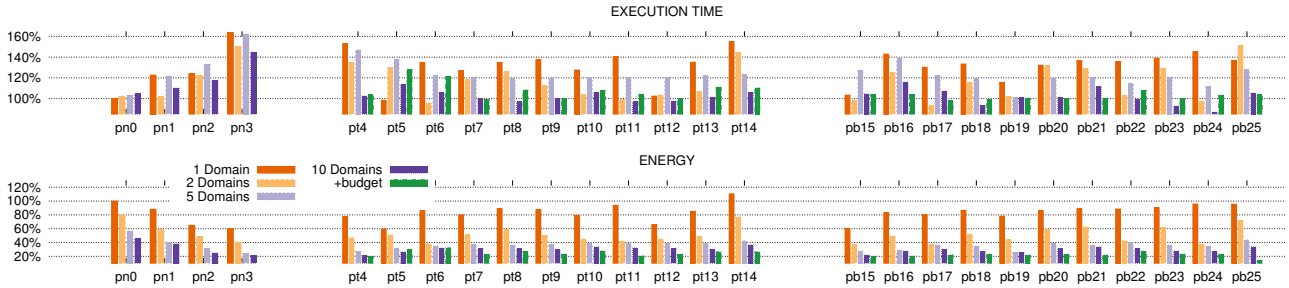

Figure 4: Design MIX: normalized execution time and energy savings across varying VF-domain count and DVFS policies.

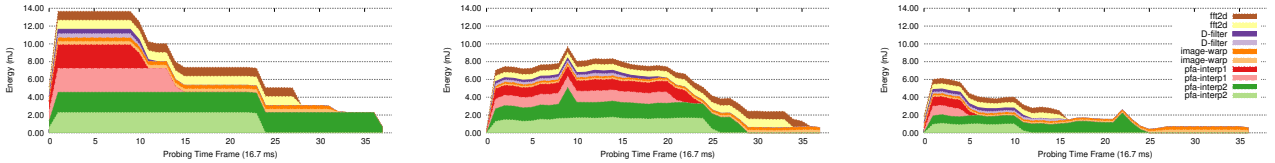

Figure 5: Design MIX: energy breakdown over time for the policies *pn0* (left), *pb25* (center) and *pb25* with PL (right).

dependent homogeneous accelerators (twelve copies of the FFT2D design); and (3) WAMI-APP features twelve heterogeneous inter-dependent accelerators that implement a portion of the *Wide Area Motion Imagery* computer-vision application [4].

For each case study, we repeat a similar set of experiments, analyzing the effects of all policies listed in Table 1. For policy PN we vary the operating point that is applied as a fixed VF pair to a domain. For the DVFS policies PT and PB we vary the values of the parameters *window*, which determines the temporal DVFS granularity in clock cycles, *traffic* threshold, and *burst* threshold.

MIX**: heterogeneous independent accelerators.** This case study involves ten accelerators (2 FFT2D, 2 D-FILTER, 2 IMAGE-WARP, 2 PFA_INTERPOLATION1 and 2 PFA_INTERPOLATION2) that run continuously and concurrently. Fig. 4 collects results in terms of execution time and energy savings for all policies, whose labels are reported on the horizontal axis. Within each group, the bar color corresponds to the application of the policy for a given number of VF domains: e.g. the first bar (dark orange) of the group labeled *pn0* corresponds to the application of policy PN to all accelerators that are part of a single VF domain; in contrast the fourth bar (violet) of this group corresponds to the application of the same policy to every accelerator, each stand-alone in a dedicated VF domain. Since *pn0* keeps the same fixed VF pair (1V, 1GHz), the different behaviors are due to the fact that the decision of turning off an accelerator can be taken only when all accelerators in a VF domain are ready to be turned off. Each group of the DVFS policies PT and PB has a fifth bar (green) that corresponds to the application of policy PL on top of the case of the fourth bar (violet). In this case, the DVFS daemon activation interval is 10ms. The height of every bar is normalized to the baseline, which is policy *pn0* applied with one single VF domain.

The results of Fig. 4 do not show a clear trend as we increase 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 and performance are penalized. On the other hand, the trend for increasing spatial granularity is clear: more VF domains yield usually a performance 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 smallest execution time (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 shows 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 16.7ms, which is the time allowed to the Ethernet interface for collecting statistics. 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 chart shows that policy *pb25* modulates the energy dissipation during the execution. Interestingly, however, the variation of the energy over time is 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. The last chart of Fig. 5 confirms the benefits of activating policy PL. All areas shrunk 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, accelerators spend less time waiting for a transaction to complete. The result is an energy saving of up to 85% of the baseline or 50% with respect to the same policy without software supervisor.

12-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 green 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

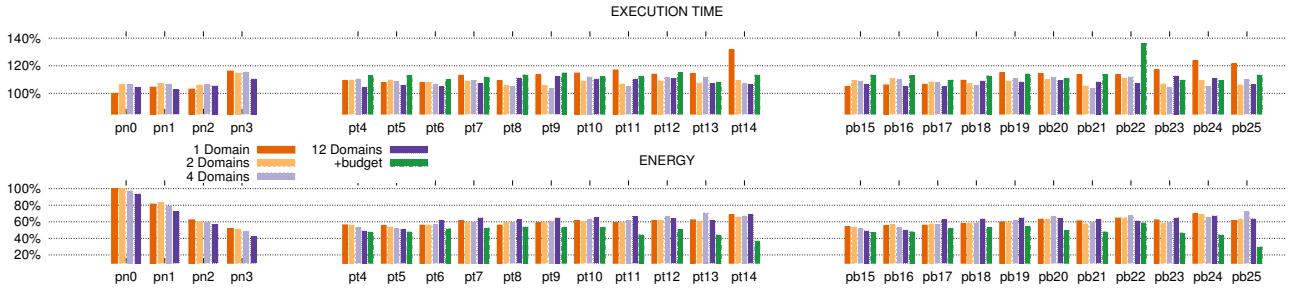

Figure 6: Design 12-FFT2D: normalized execution time and energy savings across varying VF-domain count and DVFS policies.

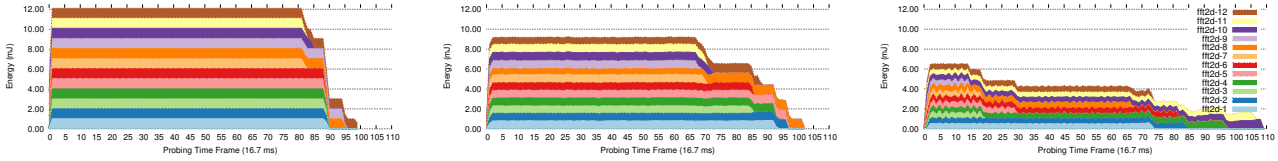

Figure 7: Design 12-FFT2D: energy breakdown over time for the policies *pn0* (left), *pt14* (center) and *pt14* with PL (right).

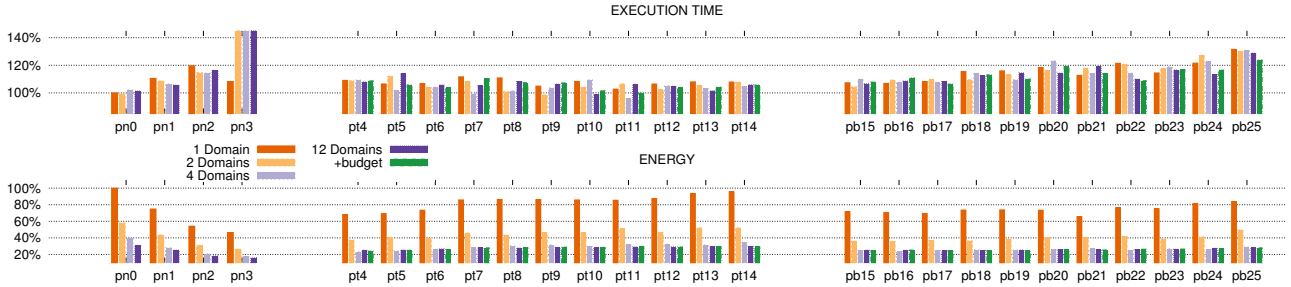

Figure 8: Design WAMI-APP: normalized execution time and energy savings across varying VF-domain count and DVFS policies.

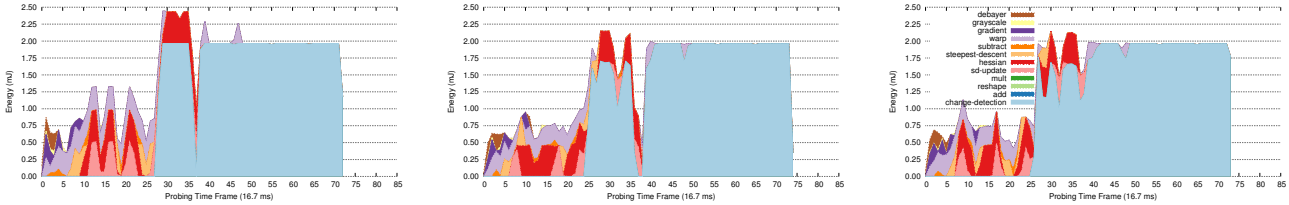

Figure 9: Design WAMI-APP: energy breakdown over time for the policies *pn0* (left), *pt14* (center) and *pt14* with PL (right).

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 other 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 processed 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 a software layer [23]. 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 is also the longest-running accelerator.

5. RELATED WORK

DVFS is a well-studied technique for reducing the power consumption of SoCs, especially in the context of processors [16, 20] or application scheduling [22, 25]. Indeed, this solution is becoming more and more relevant in SoC design to increase energy efficiency [19]. For example, Bogdan et al. [6] propose a control approach to optimize power management on an NoC-based multicore architecture starting from the analyses of the target application task graph. However, there is a lack of research examining the effects of time-space fine-grained power management on accelerator-based SoCs subject to highly variable workloads. There are different analytical studies on voltage regulators to analyze dynamics and overheads. Park et al. [24] propose a model to analyze the overhead of DC-DC converters, while Jevetic et al. [18] analyze switched-capacitor converters for fine-grained DVFS of many-core systems. Wang et al. present an analytical study and comparison of on-chip and off-chip regulators [36]. 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 DVFS granularity and showed its applicability to embedded systems with many accelerators.

Kim et al. [21] 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 [25]. DVFS 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 [13]. However, they do not apply global techniques to coordinate the scaling in case of multiple concurrent accelerators under complex full-system scenarios. Kornaros and Pnevmatikatos [22] propose an FPGA emulation of DVFS, but do not consider the effects of fined-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 DVFS, 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 DVFS 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.

Acknowledgments. This work is supported in part by DARPA PERFECT (C#: R0011-13-C-0003), the NSF (A#: 1219001), and C-FAR (C#: 2013-MA-2384), an SRC STARnet center.

7. REFERENCES

[1] ANDERSEN, T. M., ET AL. A feedforward controlled on-chip switched-capacitor voltage regulator delivering 10W in 32nm SOI CMOS. In *ISSCC Digest of Technical Papers* (Feb. 2015), pp. 22–26.

[2] ANGIOLINI, F., ET AL. Contrasting a NoC and a traditional interconnect fabric with layout awareness. In *DATE* (Mar. 2006), pp. 124–129.

[3] ARVIND. Simulation is passé; all future systems require FPGA prototyping. *Keynote Address at Embedded System Week (ESWEEK)* (Oct. 2014).

[4] BARKER, K., ET AL. *PERFECT (Power Efficiency Revolution For Embedded Computing Technologies) Benchmark Suite Manual*, December 2013. http://hpc.pnnl.gov/projects/PERFECT/.

[5] BHATTACHARJEE, A., ET AL. Full-system chip multiprocessor power evaluations using fpga-based emulation. In *ISLPED* (Aug 2008), pp. 335–340.

[6] BOGDAN, P., ET AL. An optimal control approach to power management for multi-voltage and frequency islands multiprocessor platforms under highly variable workloads. In *NOCS* (May 2012), pp. 35–42.

[7] BORKAR, S., ET AL. The future of microprocessors. *Communication of the ACM 54* (May 2011), 67–77.

[8] BURTON, E. A., ET AL. FIVR – fully integrated voltage regulators on 4th generation Intel Core SoCs. In *APEC* (Mar. 2014), pp. 16–20.

[9] CARLONI, L. P. From latency-insensitive design to communication-based system-level design. *Proceedings of the IEEE 103*, 11 (Nov. 2015), 2133–2151.

[10] CHANG, L., ET AL. A fully-integrated switched-capacitor 2:1 voltage converter with regulation capability and 90% efficiency at 2.3A/mm2. In *VLSI symp.* (June 2010), pp. 55–56.

[11] COTA, E., ET AL. An analysis of accelerator coupling in heterogeneous architectures. In *DAC* (June 2015), pp. 1–6.

[12] DALLY, W. J., ET AL. Route packets, not wires: on-chip interconnection networks. In *DAC* (June 2001), pp. 684–689.

[13] DASIKA, G., ET AL. DVFS in loop accelerators using BLADES. In *DAC* (June 2008), pp. 894–897.

[14] DIBENE, J. T., ET AL. A 400A fully integrated silicon voltage regulator with in-die magnetically coupled embedded inductors. In *APEC* (Feb. 2010).

[15] ESMAEILZADEH, H., ET AL. Dark silicon and the end of multicore scaling. In *ISCA* (June 2011), pp. 365–376.

[16] HERBERT, S., ET AL. Exploiting process variability in voltage/frequency control. *IEEE Trans. VLSI Systems 20*, 8 (Aug. 2012), 1392–1404.

[17] HOROWITZ, M. Computing's energy problem (and what we can do about it). In *ISSCC Digest of Technical Papers* (Feb. 2014), pp. 10–14.

[18] JEVTIC, R., ET AL. Per-core DVFS with switched-capacitor converters for energy efficiency in manycore processors. *IEEE Trans. on VLSI Systems 23*, 4 (Apr. 2015), 723–730.

[19] KARNIK, T., ET AL. Power management and delivery for high-performance microprocessors. In *DAC* (June 2013), pp. 1–3.

[20] KAXIRAS, S., ET AL. *Computer Architecture Techniques for Power-Efficiency*, 1st ed. Morgan and Claypool Publishers, 2008.

[21] KIM, W., ET AL. System level analysis of fast, per-core DVFS using on-chip switching regulators. In *HPCA* (Feb. 2008), pp. 123–134.

[22] KORNAROS, G., ET AL. Dynamic power and thermal management of noc-based heterogeneous mpsocs. *ACM Trans.s on Reconfigurable Technology and Systems 7*, 1 (Feb. 2014), 1–26.

[23] MANTOVANI, P., GUGLIELMO, G. D., AND CARLONI, L. P. High-level synthesis of accelerators in embedded scalable platforms. In *ASP-DAC* (Jan. 2016), pp. 204–211.

[24] PARK, J., ET AL. Accurate modeling and calculation of delay and energy overheads of dynamic voltage scaling in modern high-performance microprocessors. In *ISLPED* (Aug. 2010), pp. 419–424.

[25] RANGAN, K. K., ET AL. Thread motion: Fine-grained power management for multi-core systems. In *ISCA* (June 2009), pp. 302–313.

[26] RUSU, S., ET AL. A 22 nm 15-Core Enterprise Xeon Processor Family. *IEEE Journal of Solid-State Circuits 50*, 1 (Jan. 2015), 35–48.

[27] SALIHUNDAM, P., ET AL. A 2 Tb/s 6x4 Mesh Network for a Single-Chip Cloud Computer With DVFS in 45 nm CMOS. *IEEE Journal of Solid-State Circuits 46*, 4 (Apr. 2011), 757–766.

[28] SEMERARO, G., ET AL. Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling. In *HPCA* (2002), pp. 29–40.

[29] SIMUNIC, T., ET AL. Dynamic voltage scaling and power management for portable systems. In *DAC* (2001), pp. 524–529.

[30] STRANO, A., ET AL. A library of dual-clock FIFOs for cost-effective and flexible MPSoC design. In *SAMOS* (July 2010), pp. 20–27.

[31] STURCKEN, N., ET AL. A 2.5D integrated voltage regulator using coupled-magnetic-core inductors on silicon interposer. *IEEE Journal of Solid-State Circuits 48*, 1 (Jan. 2013), 244–254.

[32] TAYLOR, M. B. Is dark silicon useful? Harnessing the four horsemen of the coming dark silicon apocalypse. In *DAC* (June 2012), pp. 1131–1136.

[33] TIEN, K., ET AL. An 82%-efficient multiphase voltage-regulator 3D interposer with on-chip magnetic inductors. In *VLSI symp.* (June 2015), pp. 16–19.

[34] VENKATESH, G., ET AL. Conservation cores: reducing the energy of mature computations. In *ASPLOS* (Mar. 2010), pp. 205–218.

[35] WANG, N., ET AL. Ultra-high-Q air-core slab inductors for on-chip power conversion. In *IEDM* (Dec. 2014), pp. 15–17.

[36] WANG, X., ET AL. Characterizing power delivery systems with on/off-chip voltage regulators for many-core processors. In *DATE* (Mar. 2014), pp. 1–4.

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