2018 IEEE International Symposium on High Performance Computer Architecture

# GPGPU Power Modeling for Multi-Domain Voltage-Frequency Scaling

João Guerreiro, Aleksandar Ilic, Nuno Roma, Pedro Tomás
*INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal*
{*Joao.Guerreiro, Aleksandar.Ilic, Nuno.Roma, Pedro.Tomas*}*@inesc-id.pt*

***Abstract*—Dynamic Voltage and Frequency Scaling (DVFS) on Graphics Processing Units (GPUs) components is one of the most promising power management strategies, due to its potential for significant power and energy savings. However, there is still a lack of simple and reliable models for the estimation of the GPU power consumption under a set of different voltage and frequency levels.**

**Accordingly, a novel GPU power estimation model with both core and memory frequency scaling is herein proposed. This model combines information from both the GPU architecture and the executing GPU application and also takes into account the non-linear changes in the GPU voltage when the core and memory frequencies are scaled. The model parameters are estimated using a collection of 83 microbenchmarks carefully crafted to stress the main GPU components. Based on the hardware performance events gathered during the execution of GPU applications on a single frequency configuration, the proposed model allows to predict the power consumption of the application over a wide range of frequency configurations, as well as to decompose the contribution of different parts of the GPU pipeline to the overall power consumption.**

**Validated on 3 GPU devices from the most recent NVIDIA microarchitectures (Pascal, Maxwell and Kepler), by using a collection of 26 standard benchmarks, the proposed model is able to achieve accurate results (7%, 6% and 12% mean absolute error) for the target GPUs (Titan Xp, GTX Titan X and Tesla K40c).**

## I. Introduction

During the past decade, Graphics Processor Units (GPUs) have suffered many evolutions, transitioning from real-time graphics processors to high performance accelerators for general-purpose applications, with particular application in Deep Learning [1]. Because GPU architectures are able to achieve both high arithmetic throughput and high memory bandwidth [2], they are ideal to accelerate data parallel applications. GPUs are nowadays a staple in many high-performance computing (HPC) systems, confirmed by their usage in 72 of the most recent TOP500 HPC systems list.

One common challenge of GPU-accelerated systems regards the power and energy constraints. Despite their potential to high performance computing, GPU devices consume considerable amounts of power, even on underused components. This issue is often addressed by Dynamic Voltage and Frequency Scaling (DVFS), which consists on scaling the voltage and frequency of the GPU components according to the requirements of the executing applications and leading to significant power and energy savings [3], [4], [5], [6].

However, to efficiently apply these power management techniques, an accurate model is required to predict how the power consumption scales when different GPU frequency/voltage configurations are applied. Previous works have showed that applications that utilize the GPU resources differently have their performance and power consumption scale in distinct ways when DVFS is applied [7], [8], [9], [10]. Hence, to accurately characterize the GPU power consumption when executing any given application, it is necessary to analyze the usage pattern of the multiple GPU components. Other research works have proposed GPU power models [11], [12], [13], that predict the power consumption only at a fixed GPU configuration, *i.e.* not predicting the consequent changes in power consumption caused by DVFS. More recent works partially tackle this problem [14], [15], by focusing on power prediction at different frequency configurations, although achieving non-negligible accuracy errors (ranging from 10% up to 24%). Nevertheless, none ot these approaches considers the non-linear scaling of the GPU voltage with the operating frequency.

In accordance, this paper main contribution is a new approach to estimate GPU power consumption across an ample range of frequency and voltage configurations for the multiple GPU domains (core and memory). This is done by carefully crafting a set of 83 CUDA microbenchmarks, exercising the different components of real GPUs. The average GPU power consumption during the execution of each microbenchmark is measured for all frequency/voltage levels, while a collection of performance events is measured only at a reference configuration, allowing a clear understanding of how the power consumption changes with DVFS and how each microbenchmark exploits the underlying GPU.

With this information, a power model for the considered GPU device is estimated using an iterative heuristic algorithm that relies on statistical regression. Based on the observed GPU components utilization rates, the model allows the prediction of the power consumption of each component, as well as estimating how the voltage scales with their operating frequency. Once the model is created, it is possible to characterize the power consumption of any GPU application for all frequency and voltage configurations, by measuring the performance events during its execution at a single configuration.

Beyond DVFS prediction, the proposed model can also be used in other scenarios, such as in providing an estimate of

2378-203X/18/$31.00 ©2018 IEEE
DOI 10.1109/HPCA.2018.00072

the total and/or per-component power consumption for short-lived kernels or even in devices without embedded power sensor; or provide insights on the most influential factors of GPU power consumption, useful during application optimization.

The proposed GPU power model was extensively validated with a collection of applications from standard benchmarks (Parboil [16], Rodinia [17], Polybench [18] and CUDA SDK [19]), by using real GPU devices from the three most recent NVIDIA microarchitectures (Pascal, Maxwell and Kepler). The proposed model achieves accurate results, on a frequency range of up to $2\times$ change in core frequency and $4\times$ change in memory frequency, with average errors of about 7%, 6% and 12% for the Pascal, Maxwell and Kepler devices, respectively. This is a significant improvement over the accuracy offered by previous state-of-the-art models and low-level simulators, with the benefit of also running faster than the latter. Accordingly, the most significant contributions of this paper are the following:

- a microbenchmark suite that stresses the GPU components that are the most relevant to the GPU power consumption, as well as the full disclosure of the performance events that characterize the utilization of the GPU components;
- a novel DVFS-aware GPU power model, able to predict the GPU power consumption (decoupling it at the level of each GPU component) at different frequency and voltage configurations by using performance events gathered at a single configuration — to the best of our knowledge, this is the first truly DVFS-aware power model, by being able to estimate how GPU voltage scales with the operating frequency on modern GPU devices[1];
- validation of the proposed GPU power model with standard benchmarks on commercially available GPU devices from multiple architectures, including the most recent Pascal microarchitecture.

The rest of this paper is organized as follows. Section II motivates the presented work. Section III details the proposed DVFS-aware power model and Section IV presents the proposed microbenchmark suite. Section V presents the experimental results obtained to validate the proposed model. Section VI overviews the related work and Section VII concludes the manuscript.

## II. Background and Motivation

Since GPU devices started being used as massively parallel general-purpose accelerators, their microarchitecture observed several incremental changes. Nonetheless, common design principles are usually observed, such as their modular design and structure (giving rise to both mobile and high-performance devices).

[1]The complete source code (microbenchmark suite and a tool to construct the DVFS-aware GPU power consumption model) is publicly available at: https://github.com/hpc-ulisboa/gpupowermodel.

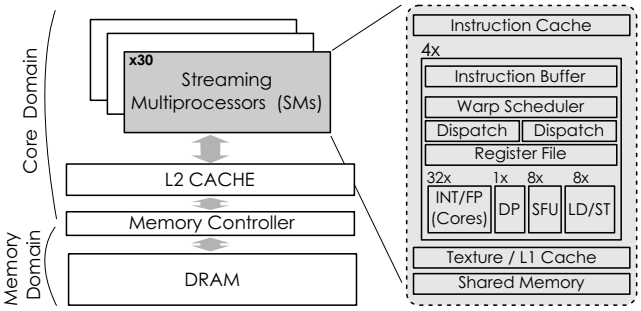


Figure 1: Block diagram of NVIDIA's Titan Xp GPU.

Another common characteristic across most GPU generations (illustrated in Figure 1 for a Titan Xp GPU) is the existence of independent frequency domains, such as the *core* (or graphics) *domain*, clocked at $\mathrm{f_{core}}$ and the *memory domain*, clocked at $\mathrm{f_{mem}}$ and which affects only the device memory (DRAM) bandwidth.

By applying DVFS to exploit the independent frequency domains, it is possible to adapt the performance of the GPU components to the requirements of the application under execution and attain energy savings [3], [4], [5], [6]. However, optimizing the GPU configuration (*i.e.* the frequency and voltage levels of both core and memory domains) is a non-trivial problem [20], [9], [10], as it requires an accurate estimation of both the execution time and average power consumption and how they change when the GPU configuration is modified.

Generally, the power consumption of a GPU device can be decomposed in the sum of the power consumptions of the multiple architectural components [21], with the power of each component ($\mathrm{C_k}$) being associated with its peak power consumption and with how an application stresses such component during its execution ($\mathrm{Power(C_k)} \propto \mathrm{Utilization(C_k)}$).

### A. Power consumption and DVFS

Gonzalez *et al.* [22] and Butts *et al.* [23] proposed the power models presented in Equations 1 and 2:

$$\mathrm{Power_{Dynamic}} = \mathrm{a} \cdot \mathrm{C} \cdot \mathrm{V}^2 \cdot \mathrm{f}, \tag{1}$$

$$\mathrm{Power_{Static}} = \mathrm{V} \cdot \mathrm{N} \cdot \mathrm{K_{design}} \cdot \hat{\mathrm{I}}_{\mathrm{leak}}, \tag{2}$$

where $a$ denotes the average utilization ratio, $C$ the total capacitance, $V$ the supply voltage, $f$ the operating frequency and $N$ the number of transistors in the chip design. $K_{design}$ is a constant factor associated with the technology characteristics and $\hat{I}_{leak}$ is a normalized leakage current for a single transistor, which depends on the threshold voltage. These two power models can be used to describe how the dynamic and static components scale with the frequency and voltage of their respective hardware elements. However,

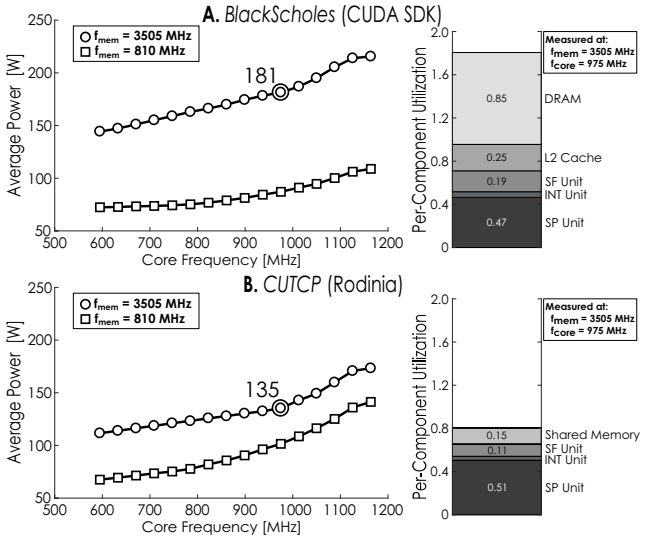



Figure 2: DVFS impact on the power consumption of two applications on the GTX Titan X GPU. On the right side is presented the utilization of the GPU components during the application execution (GPU frequencies set to $\mathrm{f_{core}} = 975$ MHz and $\mathrm{f_{mem}} = 3505$ MHz).

although they give a valuable insight on the impact of DVFS, it is usually impossible to accurately measure these two components separately, let alone determine the model individual parameters. Consequently, other approaches to model the GPU power consumption are required.

Additionally, while it is common for GPU manufacturers to provide tools to dynamically scale the frequency of each GPU domain, there is usually no way of knowing how the voltage is scaling. In fact, while in some previous NVIDIA generations (like the Fermi), the GPU voltage scaled linearly with the frequency of the cores [4], in Maxwell GPUs it is possible to scale the frequency of the cores without changing the voltage [10].

### B. DVFS impact on GPU power consumption

Each GPU application has its unique characteristics, such as the algorithm, the data types, the used operations, as well as the size of the input data, the dimensions of the grid of threads, *etc*. These characteristics determine how the different GPU components are used during the application execution. Furthermore, depending on how the applications exercise the GPU components, the effects of DVFS on the total GPU power consumption can vary between applications — the effects also depend on the architectural characteristics of each utilized component (see Equations 1 and 2).

Figure 2 presents an example of such a scenario, where the *BlackScholes* and *CUTCP* benchmarks were executed on a NVIDIA GTX Titan X GPU across multiple frequency and voltage configurations. Figure 2 also presents the utilization of the main GPU components, represented as the ratio of the achieved and peak theoretical throughputs of the component. As it can be seen, the two applications present very different utilization rates of the GPU components, which results in the distinct power consumption levels of 181W and 135W at the default frequency configuration of the GPU ($\mathrm{f_{core}} = 975$ MHz and $\mathrm{f_{mem}} = 3505$ MHz).

Additionally, it can also be seen that the variation of the power consumption when the memory frequency is decreased is much higher for the *BlackScholes* benchmark, mainly because of its greater DRAM utilization: when the memory frequency decreases from 3505 MHz to 810 MHz, the power consumption decreases by 52% (from 181W to 87W). On the other hand, the power consumption of the *CUTCP* benchmark decreases only by 24% (from 135W to 102W). Regarding the core frequency scaling, it can be seen that the GPU power cannot be represented as a simple linear function of the core frequency, as suggested in the power models proposed in previous works [12], [14], due to the implicit voltage scaling (see also Equations 1 and 2 and observe the non-linear behaviour of the power consumption in Figure 2).

From these observations, it is clear that an accurate DVFS-aware power model is needed, to characterize the relationship between the utilization of the GPU components, their runtime power consumption and how they change when the frequency/voltage of the GPU domains are scaled.

This is a gap that this research aims to close, by proposing an iterative heuristic algorithm based on statistical regression to model both the uncertainties of the GPU components and how their voltage scales with the frequency of each domain, creating an accurate power consumption model of the GPU. Through extensive microbenchmarking of the several GPU components, it is possible to estimate the parameters of the power consumption model. Once the model is constructed, one can predict the total and/or per-component power consumption of a new (unseen) application on any frequency/voltage configuration, by measuring its performance events at a single configuration.

## III. DVFS-Aware power model

### A. Power consumption model

The proposed DVFS-aware power model assumes the decomposition of the GPU power consumption across its internal components. This is done by considering that the components may operate under different frequency and voltage domains, such that:

$$\mathrm{P_{GPU}} = \sum_{k=1}^{\mathrm{N_{V\text{-}F}}} \mathrm{P(D}_k), \tag{3}$$

where $\mathrm{N_{V\text{-}F}}$ represents the number of independent voltage/frequency (V-F) domains and $\mathrm{P(D}_k)$ represents the

power consumption of each domain ($D_k$). The power of each domain ($D_k$) is defined as follows:

$$P(D_k) = \alpha_0 V_k + {V_k}^2 f_k(\alpha_1 + \sum_{i=1}^{N_C} \gamma_i \cdot U_i) \quad (4)$$

where $V_k$ and $f_k$ represent the specific voltage and frequency of the $D_k$ domain, $N_C$ is the number of GPU components operating under domain $D_k$ and $U_i \in [0,1]$ is their respective average utilization rate. The coefficients $\alpha_0$, $\alpha_1$, $\gamma_1$,...,$\gamma_{N_C}$ represent a set of hardware-specific parameters, associated to the characteristics of the underlying architecture, such as component total capacitance and leakage resistance. Hence, the proposed power model comprises 3 different terms: 1) $\alpha_0 V_k$, corresponding to the static power of the domain (see Equation 2); 2) ${V_k}^2 f_k \cdot \alpha_1$, corresponding to the power consumption associated with that specific frequency and voltage level, independent of the component utilizations (*e.g.*, idle power of that V-F level); and 3) ${V_k}^2 f_k \cdot \gamma_i U_i$, corresponding to the dynamic power of component $i$ (see Equation 1).

To reduce the number of unknown parameters, Equation 4 can be normalized to a reference voltage ($V_R$):

$$\begin{aligned} P(D_k) &= \alpha_0 V_R \frac{V_k}{V_R} + \frac{{V_k}^2}{{V_R}^2} f_k(\alpha_1 V_R + \sum_{i=1}^{N_C} \gamma_i V_R \cdot U_i) \\ &= \beta_0 \bar{V}_k + \bar{V}_k^2 f_k(\beta_1 + \sum_{i=1}^{N_C} \omega_i \cdot U_i), \end{aligned} \quad (5)$$

where $\beta_0 = \alpha_0 V_R$, $\beta_1 = \alpha_1 V_R$, $\omega_i = \gamma_i V_R$ and $\bar{V}_k = \frac{V_k}{V_R}$. This formulation is particularly useful during the model estimation, since the normalized $\bar{V}_k$ is 1 at the reference configuration, making it simpler for this particular setup and providing the grounds for the initial estimation of the parameters (see Section III-D).

Although one can generally consider multiple V-F domains, in most modern GPU devices $N_{V\text{-}F} = 2$, corresponding to the core domain ($P_{core}$), which includes the L2 cache, and the memory domain ($P_{mem}$), *i.e.* $P_{GPU} = P_{core} + P_{mem}$. By replacing Equation 5 for these two domains and by denoting with $N_{core}$ the number of components from the core domain whose power consumption is considered in the model, the following is obtained:

$$P_{core} = \beta_0 \bar{V}_{core} + \bar{V}_{core}^2 f_{core}(\beta_1 + \sum_{i=1}^{N_{core}} \omega_i U_i), \quad (6)$$

$$P_{mem} = \beta_2 \bar{V}_{mem} + \bar{V}_{mem}^2 f_{mem}(\beta_3 + \omega_{mem} U_{mem}). \quad (7)$$

Equations 6 and 7 show the distinctive approach of the proposed model when compared with the state-of-the-art [12], [14], since it considers multiple V-F domains and relies on a more accurate relationship between frequency scaling, voltage levels and power consumption.

### B. Hardware utilization metrics

To accurately determine the utilization parameters ($U_i$) in Equations 6 and 7, a set of metrics is defined, which consider the GPU hardware components with the greatest contribution to the power consumption variations, namely: integer (Int), single- and double-precision floating-point (SP/DP) and special-function (SF) units, shared memory, L2 cache and DRAM. Although it would be potentially beneficial to consider more components of the GPU architecture (*e.g.* L1 instruction and data caches, texture units, *etc.*), it is not easy to assess their real-time utilization, since NVIDIA does not disclose events or metrics to accurately describe their average utilization. Nonetheless, should such information be disclosed, one may easily consider other hardware units, in order to further improve the model accuracy. Moreover, as it will be described in Section III-C, even for the selected GPU components it was deemed necessary to rely on undisclosed events, by performing extensive experimental testing to uncover their potential meaning.

The utilization level of the considered GPU compute units, measured during the application execution, can be obtained by observing the number of executing warps and by comparing it to the number of warps that would execute if the units were always filled:

$$U_x = \frac{AWarps_x \cdot WarpSize}{ACycles \cdot UnitsPerSM_x}, x \in \{Int, SP, DP, SF\}, \quad (8)$$

where $AWarps_x$ is the number of warps executing on unit $x$ during the application execution, $ACycles$ is the number of cycles when there is at least one active warp on the SMs, $UnitsPerSM_x$ is the number of units of type $x$ on each SM and $WarpSize$ is the number of threads in a warp (a characteristic of the GPU device).

On the other hand, the utilization rate of the different memory hierarchy levels can be computed by looking at the achieved bandwidth at each level ($ABand$) and by comparing it with the corresponding peak bandwidth ($PeakBand$), such that:

$$U_y = \frac{ABand_y}{PeakBand_y}, y \in \{L2, Shared, DRAM\}. \quad (9)$$

### C. Architecture-specific events

Some of the metrics used to compute the utilization rate in the proposed model can be directly gathered from the publicly available device characteristics, such as the $UnitsPerSM$ for the Int, SP, DP and SF units, and the $WarpSize$. The DRAM and shared memory peak bandwidth can be calculated using the known device characteristics ($PeakBand = f \cdot \frac{Bytes}{Cycle}$, where f is the operating frequency of that memory level). The L2 cache peak bandwidth cannot be computed as trivially, as it was shown by numerous works [24], [25], [26]. Hence, it was experimentally determined with a set of specific L2 microbenchmarks (detailed in Section IV), specifically developed for this purpose.

Table I: Performance events required to compute the metrics used in the proposed power consumption model.

| Metric | Titan Xp | GTX Titan X | Tesla K40c |
|---|---|---|---|
| ACycles | *active_cycles* | | |
| $\text{ABand}_{\text{L2}}$ | *l2_subp*$_{\{0,1\}}$*_total_rd_sq** *l2_subp*$_{\{0,1\}}$*_total_wr_sq** | | *l2_subp*$_{\{0,1,2,3\}}$*_t_rd_sq** *l2_subp*$_{\{0,1,2,3\}}$*_t_wr_sq** |
| $\text{ABand}_{\text{Shared}}$ | *shared_ld_trans* *shared_st_trans* | | *l1_sh_ld_trans* *l1_sh_st_trans* |
| $\text{ABand}_{\text{DRAM}}$ | *fb_subp*$_{\{0,1\}}$*_rd_sectors* *fb_subp*$_{\{0,1\}}$*_wr_sectors* | | |
| $\text{AWarps}_{\text{SP/INT}}$† | W580, W581 | W361, W362 | W131, W134 W136, W137 |
| $\text{AWarps}_{\text{DP}}$† | W584 | W364 | W141 |
| $\text{AWarps}_{\text{SF}}$† | W560 | W359 | W133 |
| $\text{Inst}_{\text{INT}}$† | W831 | W504 | W205 |
| $\text{Inst}_{\text{SP}}$† | W829 | W502 | W203 |

* sq - sector_queries

† The prefix W stands for: 352321 for Titan Xp, 335544 for GTX Titan X and 318767 for Tesla K40c.

However, other required metrics depend on the average utilization of the GPU components, which also depend on the application characteristics (and therefore, need to be measured during their execution). Table I presents the set of performance events that were used to obtain the remaining parameters shown in Equations 8 and 9, collected using the NVIDIA CUPTI library. The events identified with a label correspond to the events disclosed by NVIDIA. The remaining events, identified with a numeric ID, were selected through an extensive experimental testing in order to assess their meaning. Furthermore, since some of the considered metrics (*e.g.* $\text{ABand}_{\text{DRAM}}$) depend on the values of multiple performance events (4 for this specific metric) an aggregation step needs to be conducted.

Moreover, since in the considered GPU devices the events related with the SP and Int units are combined into the same set of events (making them indistinguishable), the utilization of each of those components is determined by the ratio of instructions executed for each instruction type:

$$\text{AWarps}_{\text{z}} = \frac{\text{AWarps}_{\text{Int/SP}}\text{Inst}_{\text{z}}}{\text{Inst}_{\text{Int}} + \text{Inst}_{\text{SP}}}, \text{z} \in \{\text{Int}, \text{SP}\}. \quad (10)$$

### D. Model parameter estimation

The final step towards the definition of the proposed DVFS-aware power model corresponds to the determination of the unknown parameters $\mathbf{X} = [\ \beta_0, \beta_1, \beta_2, \beta_3, \omega_{\text{mem}}, \omega_1, \ldots, \omega_N\ ]$ and of the set of voltages $\bar{\mathbf{V}} = (\bar{\text{V}}_{\text{core}}, \bar{\text{V}}_{\text{mem}})$ associated with each frequency configuration (which are also considered as unknowns because general GPU drivers do not directly provide these values). Hence, a set of specifically developed microbenchmarks (described in Section IV) is used to stress the considered GPU components to better characterize their uncertainties. The set of measurements gathered during the execution of these microbenchmarks, *i.e.* the utilization rates and the power consumption at each V-F configuration, can then be used to estimate the parameters of the proposed model. Moreover, since there is a relation between the unknowns $\bar{\text{V}}_{\text{k}}$ and $\beta_i/\omega_i$ in the proposed model (Equations 6 and 7), a simple least squares regression cannot be used, as it leads to a non-full-rank optimization problem. Hence, an iterative optimization algorithm was devised to estimate such parameters, which works as follows:

1) Determine the initial value of the unknowns $\mathbf{X}$ by considering the reference frequency $\mathbf{F_1} = (\text{f}_{\text{core1}}, \text{f}_{\text{mem1}})$, where $\bar{\text{V}}_{\text{core1}} = \bar{\text{V}}_{\text{mem1}} = 1$. Consider also two additional configurations $\mathbf{F_2} = (\text{f}_{\text{core2}}, \text{f}_{\text{mem1}})$ and $\mathbf{F_3} = (\text{f}_{\text{core1}}, \text{f}_{\text{mem2}})$ and assume that their corresponding normalized voltage levels are also 1 ($\bar{\text{V}}_{\text{core2}} = \bar{\text{V}}_{\text{mem2}} = 1$). By using the measurements obtained at those three configurations, solve the following linear system using a least squares estimation:
$$\mathbf{X} = \arg\min_{\mathbf{X}} \sum_{\text{Microbench.} \in \mathbf{BA}} \left(\text{P}_{\text{meas.}} - \hat{\text{P}}\right)^2 \quad \text{s.t. } \bar{\text{V}}_{\text{core}} = \bar{\text{V}}_{\text{mem}} = 1, \quad (11)$$
where $\text{P}_{\text{meas.}}$ is the measured power consumption, $\hat{\text{P}} = \text{P}_{\text{core}} + \text{P}_{\text{mem}}$, as given by Equations 6 and 7, and $\mathbf{BA}$ is the set of microbenchmarks executed at the frequency configurations $\mathbf{F_1}, \mathbf{F_2}$ and $\mathbf{F_3}$.
2) By using the previously determined vector of parameters $\mathbf{X}$ — for each frequency configuration ($\text{f}_{\text{core}}$, $\text{f}_{\text{mem}}$) — use the measurements from the set of microbenchmarks to estimate the values of $\bar{\text{V}}_{\text{core}}$ and $\bar{\text{V}}_{\text{mem}}$, by solving the following problem:
$$\text{For each } \mathbf{F} = (\text{f}_{\text{core}}, \text{f}_{\text{mem}}) \text{ vector, solve :} \quad \bar{\mathbf{V}} = \arg\min_{\bar{\mathbf{V}}} \sum_{\text{Microbench.} \in \mathbf{BB}} \left(\text{P}_{\text{meas.}} - \hat{\text{P}}\right)^2 \quad \text{s.t. } \forall_{\text{f}_{\text{x1}} > \text{f}_{\text{x2}}} \bar{\text{V}}_{\text{x1}} \geq \bar{\text{V}}_{\text{x2}}, \text{x} \in \{\text{core}, \text{mem}\} \quad (12)$$
where $\bar{\text{V}}_{\text{xi}}$ is the voltage level associated with frequency $\text{f}_{\text{xi}}$ and $\mathbf{BB}$ is the set of microbenchmarks executed at frequency configuration ($\text{f}_{\text{core}}$, $\text{f}_{\text{mem}}$).
3) Considering the newly determined values for $\bar{\text{V}}_{\text{core}}$ and $\bar{\text{V}}_{\text{mem}}$, repeat step 1 to estimate the new values of $\mathbf{X}$, but using the measurements from all frequency levels, *i.e.* by extending $\mathbf{BA}$ to include the set of measurements taken for all microbenchmarks at all frequency configurations.
4) Iterate between steps 2 and 3 until convergence is achieved, or the maximum number of iterations is reached.

One benefit of the proposed methodology over previous studies is the ability to dynamically determine how the GPU voltage is scaling for each frequency configuration. Considering that part of the power consumption scales with the square of the voltage, it is important to have an informed knowledge of these values, in order to achieve an accurate model of the architecture. Hence, despite being impossible to measure the real-time voltage of each GPU domain in many

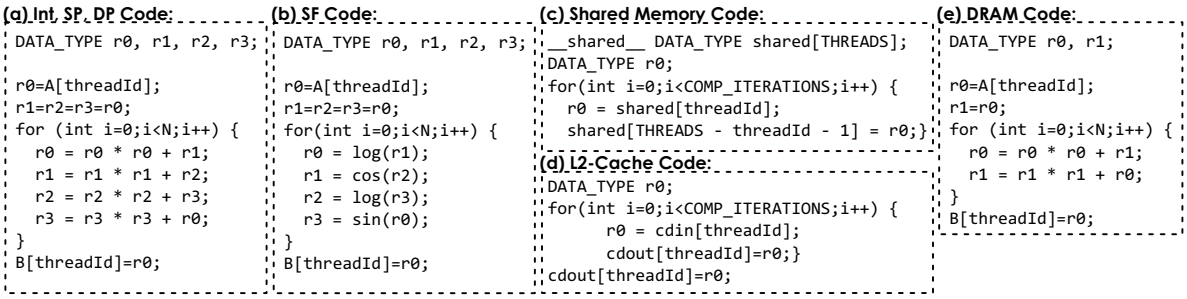



Figure 3: Example CUDA source code of some of the used microbenchmarks.

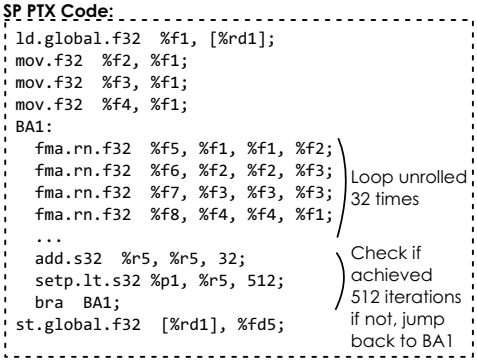



Figure 4: PTX source code of the microbenchmark stressing the single-precision floating-point units.

computing systems, the proposed methodology still allows achieving accurate power predictions, since no assumption is made on how the voltage scales with frequency. However, if there is a previous information regarding the voltage levels of each domain at any given frequency configuration, the proposed methodology can be simplified into a single execution of step 3, by utilizing the real voltage values.

### *E. Power consumption prediction*

With the model parameters determined, it is possible to predict how the voltage scales with the frequency of each GPU domain and to obtain the total power consumption of any executed application for the whole range of the device V-F configurations, by simply measuring its performance events on a single configuration. This allows a considerable decrease of the design search space, which is a highly valuable advantage when applying DVFS in real-time. Finally, the obtained power model also allows the decomposition of the power consumption into the partial consumptions of the several GPU components.

## IV. Microbenchmarking the GPU

To model the unknown characteristics of the underlying architecture, the proposed modelling methodology relies on in-depth microbenchmarking of specific GPU components. By creating a wide set of microbenchmarks, covering the several components of the GPU, it is possible to isolate their power consumption, enabling an accurate prediction of their contribution to the total GPU power consumption.

Figure 3 presents a subset of CUDA code examples from the developed microbenchmarks. To stress the main arithmetic units (Int, SP and DP), the microbenchmark presented in Figure 3a was developed, where the DATA_TYPE can be switched between *int*, *float* and *double*. Figure 4 presents the PTX code corresponding to the SP variation of the microbenchmark, where it can be seen that the FP operations make use of architectural registers. When executing this microbenchmark, each thread starts by initializing the values of 4 registers with data from the global memory. Afterwards, each thread executes a series of multiply and addition operations (using the PTX fused multiply-add instruction), until a number of N iterations is reached (N=512 in the example shown in Figure 4). The threads finalize by storing the computed value back to the global memory. By running the same code with different values of N, it is possible to characterize the impact of different instruction mixes to the GPU power consumption, by assigning different amounts of arithmetic operations per memory access (arithmetic intensity). As the value of N increases, more arithmetic instructions are executed for each pair of load/store instructions, resulting in increasingly higher levels of utilization of the corresponding arithmetic units and lower utilization levels of the memory hierarchy (DRAM and L2 cache).

Figure 3b presents the developed microbenchmark to stress the special-function units. The code is very similar to the previous arithmetic microbenchmarks, with the difference relying on using transcendental operations instead of a simple multiply and addition.

The microbenchmark presented in Figure 3c was developed to stress the memory subsystem, where each thread consecutively performs one load and one store to the shared memory. The load and store addresses are chosen in a way that minimizes the shared-memory bank conflicts for both loads and stores.

Due to the absence of publicly available information regarding the operation and structure of the L2 cache on NVIDIA GPUs, developing a microbenchmark to stress this

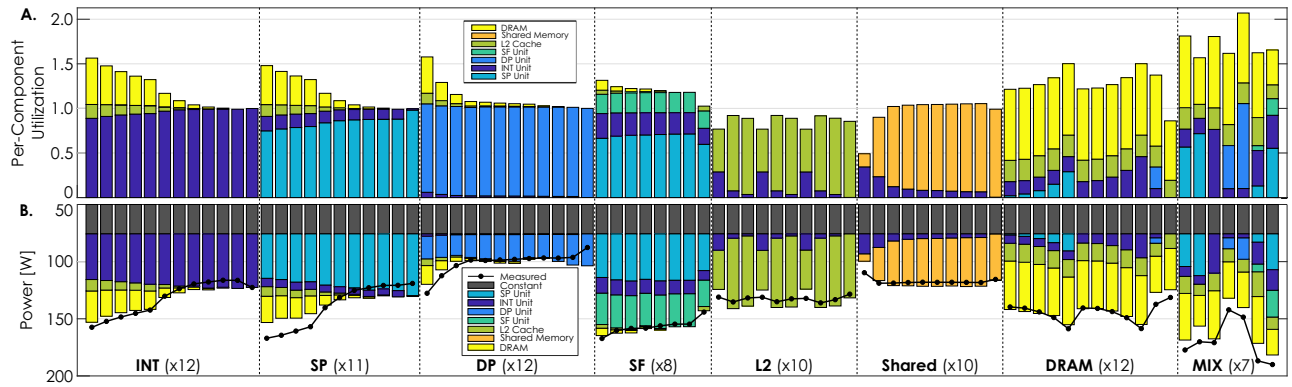



Figure 5: Per-component utilization rates and power breakdown of the microbenchmark suite on GTX Titan X at the default frequency ($f_{core} = 975$MHz and $f_{mem} = 3505$MHz).

component is not a trivial task. Hence, the considered microbenchmark (Figure 3d) is based on [26], where different access patterns are explored to characterize the L2 cache.

Figure 3e illustrates the microbenchmarks used to stress the GPU DRAM. While the structure is rather similar to the arithmetic microbenchmarks (Figure 3a), lower arithmetic intensities can be obtained by assigning a lower number of arithmetic instructions per loop and by choosing smaller values for N, which results in higher utilization of the DRAM (since the threads spend less time inside the SMs).

Finally, a set of microbenchmarks corresponding to a mix of the several used components was also considered, as well as a microbenchmark where the GPU is awaken with no executing kernel (Idle), resulting in the proposed suite of 83 microbenchmarks.

Figure 5A presents the utilization rate of the seven considered GPU components, obtained by executing the developed microbenchmarks on the GTX Titan X GPU (Maxwell) at the default frequency configuration. Looking at the first 11 microbenchmarks from the Integer collection, it is possible to see the effects of varying the arithmetic intensity (by increasing N, in Figure 3a). This can be observed with the gradual decrease of the utilization rate corresponding to both elements from the memory hierarchy (DRAM and L2 cache), and by the increase in the utilization of the Int units. The same behaviour can be observed for the SP, DP and SF microbenchmarks. Regarding the memory microbenchmarks, these successfully stress the corresponding memory element, with varying degrees of utilizations obtained again by varying the loop iterations in their source code. Overall, the results show that the proposed microbenchmark suite successfully accomplishes its design goal, *i.e.* in stressing the considered components.

Upon model construction, *i.e.* after estimating the unknown parameters in Equations 6 and 7, it is possible to predict the power consumption of each microbenchmark at the component-level. Figure 5B illustrates the per-component power breakdown of each microbenchmark, together with the total measured power consumption at the default frequency configuration. From these results, it can be observed that the proposed model is particularly accurate in predicting the power consumption on this set of applications (see Section V-B for a robust assessment of the results using an independent set of applications) and that the power consumption of the components follow their corresponding utilization rate. Additionally, it can also be observed that, for this V-F configuration, the constant portion of the power consumption, *i.e.* the terms from Equations 6 and 7 that do not depend on the components utilization, contribute with 84W to the total power consumption, and that the maximum contribution of the dynamic power is about 49% (achieved in one of the Mix microbenchmarks).

## V. Experimental Results

### A. Experimental setup

To validate the proposed model, three GPUs from the most recent NVIDIA microarchitectures (see Table II) were used as testing platforms on a Linux CentOS7 environment. While the Tesla K40c GPU has a single non-idle memory frequency level, the two other GPUs allow multiple configurations. For this reason, and since the Titan GPUs are more recent, the presented results will mostly focus on the GTX Titan X and Titan Xp.

The NVML library was used for monitoring and changing the operating frequencies of the GPU domains (while the voltage is automatically set). The real power measurements are also obtained using NVML, whose values are refreshed at an estimated 35ms period for the Titan Xp, 100ms for the GTX Titan X and 15ms for the Tesla K40c. Since many GPU benchmarks have very short execution times, which may result in misleading power measurements given the observed refresh rates, the kernels were repeatedly executed whenever necessary, to always reach an execution time of at least 1 second at the fastest GPU configuration (highest core

Table II: Summarized description of the used GPUs.

| | Titan Xp | GTX Titan X | Tesla K40c |
|---|---|---|---|
| **Base architecture** | Pascal | Maxwell | Kepler |
| **Compute capability** | 6.1 | 5.2 | 3.5 |
| **Memory frequencies** (MHz) | {5705, 4705}* | {4005, 3505, 3300, 810} | 3004 |
| **Core freq. range** (MHz) | [1911:582] | [1164:595] | [875:666] |
| Number of core freq. levels | 22 | 16 | 4 |
| **Default Mem. Frequency** | 5705 | 3505 | 3004 |
| **Default Core Frequency** | 1404 | 975 | 875 |
| **Threads per warp** | 32 | 32 | 32 |
| **Number of SMs** | 30 | 24 | 15 |
| **Memory Bus Width** | 48B | 48B | 48B |
| **Shared mem. banks** | 32 | 32 | 32 |
| **SP/INT Units/SM** | 128 | 128 | 192 |
| **DP Units/SM** | 4 | 4 | 64 |
| **SF Units/SM** | 32 | 32 | 32 |
| **TDP** (W) | 250 | 250 | 235 |

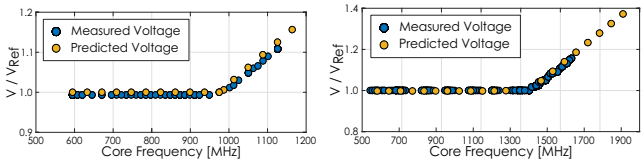
* NVIDIA driver does not allow setting the memory frequency to lower levels.

Table III: Standard benchmarks used to validate the proposed power model.

| Suite | Application Name |
|---|---|
| Rodinia [17] | *Streamcluster, Backprop, LUD, Gaussian, Hotspot, K-Means, ParticleFilter_naive, ParticleFilter_float, SRAD_v1, SRAD_v2* |
| Parboil [16] | *CUTCP, LBM* |
| Polybench [18] | *2MM, 3MM, FDTD-2D, SYRK, CORR, GEMM, GESUMMV, GRAMSCHM, SYRK_DOUBLE, 3DCONV, COVAR* |
| CUDA SDK [19] | *Blackscholes, ConjugateGradientUM, matrixMulCUBLAS* |

and memory frequencies). The power consumption of each kernel was computed as the average of all gathered samples. For benchmarks with multiple kernels the total power consumption was obtained by weighting the consumption of each kernel with its relative execution time. To guarantee the accuracy of the presented results, all benchmarks were repeated 10 times, with the presented values corresponding to the median value.

The results will be analysed by considering the whole frequency range in Section V-B. To create the model, the microbenchmark suite described in Section IV was executed on different frequency levels, with the performance events shown in Table I being measured only at the reference frequency levels. The algorithm proposed to estimate the power model converged in less than 50 iterations, corresponding to about 30 seconds on an Intel i7 4500U processor. To obtain a bias-free validation of the model, an independent collection of 26 applications from 4 benchmark suites was used (see Table III), with each application being executed only at the reference frequency configuration to measure the required hardware events. Since these applications were not used to estimate the model parameters, they allow showing the model robustness for new (unseen) applications. The




(a) GTX Titan X. (b) Titan Xp.

Figure 6: Measured vs. predicted core voltage.

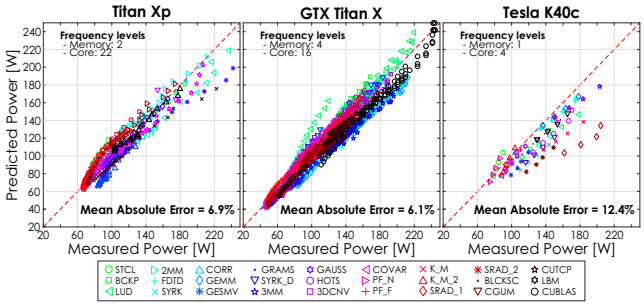



Figure 7: Power prediction for all V-F configurations, for the validation set of standard benchmarks (not used in the model construction).

accuracy of the predictions is validated using the power consumption measurements taken at all considered frequency levels (see Table II).

### B. DVFS-aware power model validation

***Voltage levels prediction:*** Even though the devised model assumes that both $V_{core}$ and $V_{mem}$ can scale with the changes in frequency of the two GPU domains (see Section II-B), from extensive experimental testing, no voltage differences were observed across the different memory frequency levels for the considered GPUs.

On the other hand, the results denote clear differences on the core voltage levels for the GTX Titan X and Titan Xp GPUs. Figure 6 presents the comparison between the predicted core voltage (obtained during the construction of the model), with the measured voltage for the Titan Xp and GTX Titan X. The real measured voltages were obtained using the NVIDIA Inspector and MSI Afterburner (third party Windows tools). However it was not possible to sweep through all core and memory frequency ranges, due to limitations of these tools, nor was it possible to verify the voltage levels on the Tesla K40c GPU. The results clearly show that there are two distinct regions for the core voltage when scaling the core frequency: i) a constant voltage region, for lower frequencies; and ii) after a specific frequency, the voltage starts increasing linearly with the frequency. By comparing the predicted and measured values, it can be observed that the devised model is accurate in predicting the core voltage, and in identifying the breaking point between the two distinct regions. The existence of these

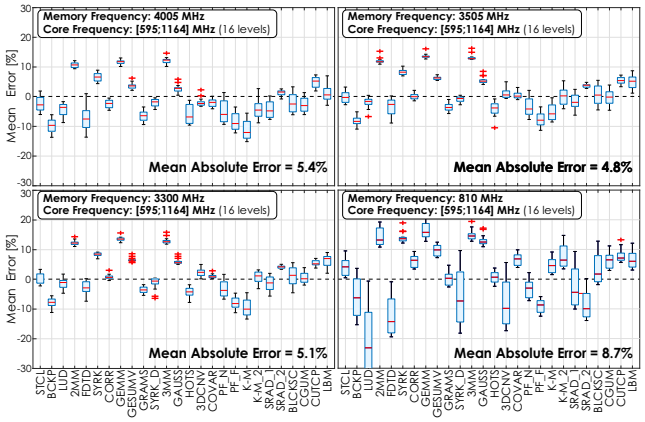
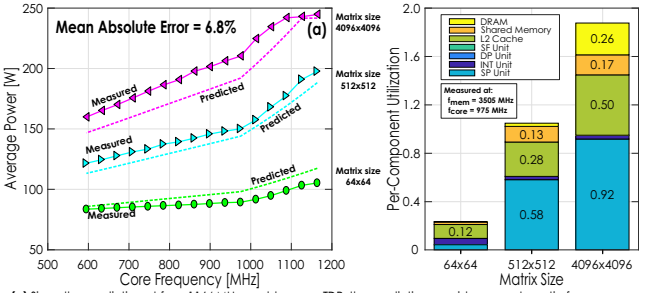



Figure 8: Prediction error obtained with the validation set of benchmarks on GTX Titan X. Each figure presents the prediction error for all *core* frequencies with a fixed *memory* frequency of all benchmarks.




(a) Since the prediction at $f_{core}$=1164 MHz would surpass TDP, the prediction considers an automatic frequency decrease to the closest frequency level ($f_{core}$=1126 MHz) that does not violate TDP.

Figure 9: Effects of varying the input matrices size for the *matrixMulCUBLAS* kernel, on the GTX Titan X.

two different scaling behaviours may significantly affect the power consumption of an application over the different core frequencies. Moreover, it should be highlighted that significant core voltage differences are predicted on the GTX Titan X across different memory frequencies (although it is not shown in the figure because such values could not be validated).

***Power consumption prediction with DVFS:*** Figure 7 presents the accuracy of the proposed power model for the validation benchmarks[2], for multiple core and memory V-F configurations. Since multiple frequency settings were taken into account, the range of obtained power values is large, *e.g.* going from 40W up to 248W on the GTX Titan X.

On the validation benchmarks the model achieves mean absolute errors of 6.9%, 6.0% and 12.4% for the three devices. The observed higher error on the Tesla K40c can be explained by a reduced accuracy of the hardware events when characterizing the utilization of the GPU components (using the undisclosed events presented in Table I). Nonetheless, the achieved prediction error is still considerably lower than the one achieved in previous works on the same microarchitecture, where the prediction error was 23.5% [14]. The architecture and existing performance events of the two other GPUs (GTX Titan X and Titan Xp) are very similar, resulting in similar accuracies of the power model. It is also worth noting that even by using the performance events measured only at the reference configuration, the model achieves accurate results when predicting the power consumption for a wide range of configurations. In particular, the results of the GTX Titan X show accurate predictions up to a frequency range of 4.3× for the memory frequency (3505MHz→810MHz) and 1.6× for the core frequency (975MHz→595MHz).

Figure 8 depicts the prediction error of the power model for the considered validation benchmarks for different memory frequencies of the GTX Titan X. When covering a frequency range of 2× for the core frequencies and 4× for the memory frequencies, a mean prediction error of 6.0% is still achieved, over all the V-F configurations. As one might expect, when predicting the power at the operating frequency furthest away from the reference configuration (where the performance events that were used to build the model were measured), the accuracy error slightly increases. This is seen by the 4.9% accuracy error when $f_{mem} = 3505$ MHz, while at $f_{mem} = 810$ MHz the accuracy error increases to 8.7%.

***Input data size:*** For a given kernel, the characteristics of the input data will determine how the different GPU components are stressed. For example, one kernel with small enough input data such that it fits in the L2-cache is expected to have different resource utilization than the same kernel with a much larger input data, as this will lead to an increase of DRAM accesses. Naturally, the different utilization patterns are taken into account in the model, resulting in distinct power consumptions.

Figure 9 presents the effects of varying the size of the (square) input matrices of the *matrixMulCUBLAS* kernel in the GPU power consumption and in the utilization of each GPU component. As it can be seen, with larger input data sizes, the utilization of the SP unit, L2-cache and DRAM increase, resulting in the presented rise of the GPU power consumption, which is predicted by the proposed model with a 6.8% average error.

***Decoupling the GPU power:*** Once the model is fully determined, it is possible to estimate the power consumption of each GPU component for any application. This power-breakdown can be particularly interesting for application optimization, since it provides the developers with crucial information about which components represent the main power consumption bottlenecks.

Figure 10 presents the utilization and power breakdown of the set of standard benchmarks for two V-F configurations on the GTX Titan X GPU. From the presented results, it can

[2]The validation benchmarks were not used in the construction of the model, to allow a robust assessment of the results.

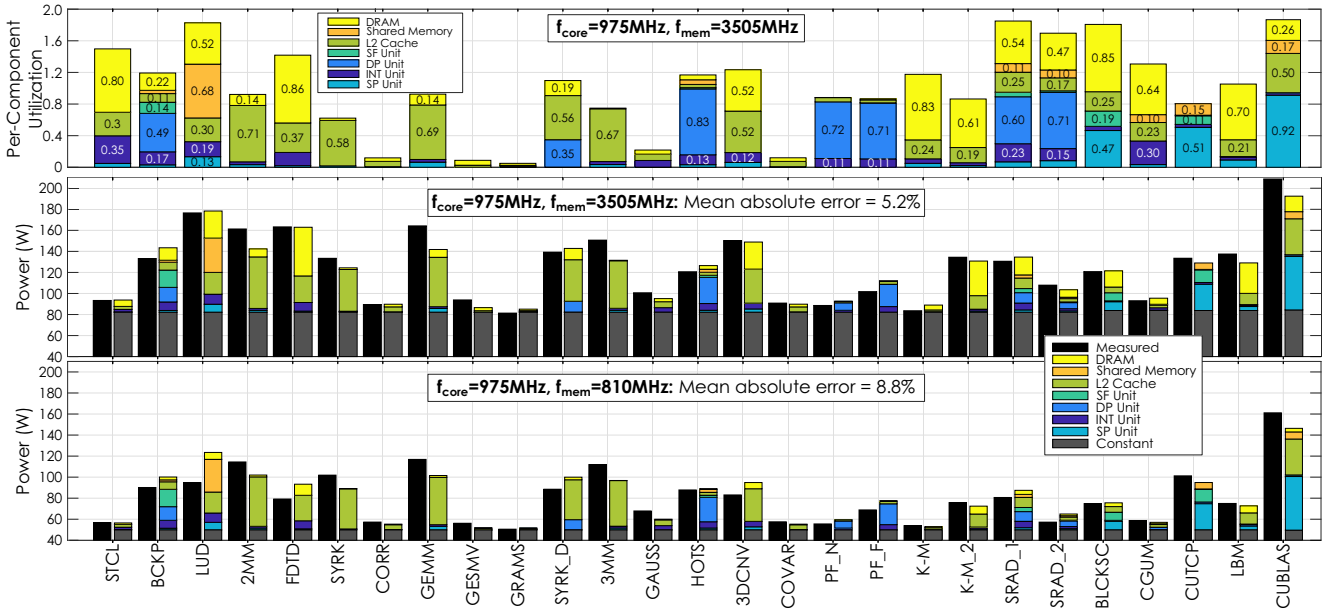



Figure 10: Power consumption breakdown on the real benchmarks, on the GTX Titan X GPU.

be observed that the group of validation benchmarks is rather representative, presenting large differences in the utilization levels of the different GPU components. Nonetheless, in both V-F configurations, it can be seen that a non-negligible portion of the power consumption is accounted for in the constant part (80W for the reference configuration and 50W for the low memory configuration), which aggregates the static power, the idle power of that frequency configuration, and the power consumptions of other non-modelled GPU components (due to the lack of informative counters). Naturally, between the two V-F configurations, the large variation in DRAM operating frequency leads to the observed large variation in the DRAM power consumption, while the power of the remaining components stays almost constant.

***Use cases:*** The proposed power consumption model can be applied on the following scenarios: **1) GPUs without sensor**, by using a previously built model (*e.g.* using external sensors) to provide an estimate of the total and/or per-component GPU power consumption (similarly to [27] from Intel). **2) Application analysis**, by using the per-component breakdown to assess the power bottlenecks of developing applications (alternative to the usual performance optimization); or even in a virtualization scenario (*e.g.* NVIDIA GRID system using Hyper-V execution [28]), where the model — constructed in the Hypervisor — could be provided to the guest VMs, allowing them to estimate their corresponding total and/or per-component power consumption (which they currently have no way of measuring). **3) DVFS management**, by facilitating the search for the optimal frequency state, as it allows estimating the power consumption at different frequency configurations without requiring exhaustive execution on all possible configurations as in [29]. **4) GPU hardware integration**, by implementing the proposed model in hardware (similarly to Intel RAPL [30]), where it would be able to take into account fine-grained V-F perturbations and potentially even non-SMU (System Management Unit) V-F adjustments.

## VI. Related Work

Initial attempts to model the GPU power consumption were focused on modelling the power at a fixed frequency/voltage configuration, neglecting the effect of DVFS [31], [32], [33], [34], [35], [36]. In particular, Nagasaka *et al.* [37] proposed a power consumption model for a Tesla GPU (GTX285) based on hardware performance events and on a statistical approach to find the correlation between the performance profiles and the GPU power consumption. They achieved 4.7% average prediction error, although they also stated that the approach was ineffective on more recent GPUs, namely those from the Fermi generation. Hong *et al.* also proposed a power model for a Tesla GPU (GTX280) [11] based on an analysis of both the binary PTX and of the pipeline, at runtime. The offline PTX analysis allows this model to attain highly accurate GPU power predictions, at the cost of being very GPU-specific. Hence, such an approach lacks the ability to make accurate predictions for different GPU architectures, or even for the same GPU at different core and memory configurations.

Song *et al.* used an artificial neural-network to train the GPU power consumption [13], achieving better prediction accuracy than other traditional regression based models. However, neural network approaches usually create output

models of high complexity, where it is often hard to extract its physical meaning.

Leng *et al.* integrated Hong's power model inside the GPGPU-Sim [38] simulator to form GPUWattch [12]. Supporting both NVIDIA's Tesla and Fermi GPU architectures, GPUWattch can estimate the cycle-level GPU power consumption during application execution. The considered model assumes that the power consumption of a GPU domain always scales linearly with its frequency[12, eq.6]. However, as it was previously shown (see Figure 2) this is not always the case, because of the non-linear behaviour of the voltage (see also Figure 6). Nath *et al.* also used GPGPU-Sim to create a performance model for DVFS, which could potentially be expanded to include a power model [39]. However, such approach requires adding logic to the GPU scoreboard, making it impossible to replicate on real hardware. This type of approaches has been deemed product-specific and difficult to apply on modern GPUs [14].

Abe *et al.* proposed DVFS-aware power regression models for GPUs from the NVIDIA's Tesla, Fermi and Kepler generations [14], which separate the GPU power consumption in core and memory domains, each proportional to their corresponding frequency and associated performance events. The models are estimated with linear regression by using measurements taken at 3 different core and 3 different memory frequencies. The proposed models achieved average prediction errors of 15% for the Tesla GPU, 14% for the Fermi GPU and 23.5% for the most recent Kepler GPU. However, the authors do not disclose which set of performance events are used in the model. Additionally, despite performing the power consumption decomposition in the core and memory domains, similar to the one herein proposed, the authors do not consider the non-linear scaling effects of the voltage.

Wu *et al.* studied how the performance and power consumption of an AMD GPU scale with core and memory frequency variations, as well as with different number of cores [15]. These authors group GPU applications into distinct clusters based on their characteristics, each representing a different performance/power scaling. Neural-network classifiers are used to characterize new applications, by predicting which scaling factor better represents an application. They achieve an average deviation of about 10% on the tested GPU device. However, the model accuracy is highly dependent on a set of fine-tuned parameters, such as the number of clusters, which makes it difficult to replicate on different architectures. More recently, in a follow-up [40], a technique was proposed to optimize the GPU energy efficiency by predicting the characteristics of upcoming kernels, based on recent execution history.

## VII. Conclusions and Future Work

This manuscript presented a new DVFS-aware GPU power model that is able to predict the power consumption of the several GPU components for any frequency/voltage configuration, by using the performance events gathered at a single configuration. The proposed approach makes use of an especially devised iterative algorithm that relies on statistical regression and is able to model not only the unknown characteristics of the underlying architecture, but also to accurately predict how the GPU voltage scales with the core/memory domain frequencies.

When used with a set of three different GPUs, representing the three most recent NVIDIA microarchitectures, the proposed DVFS-aware power model accurately estimates the power consumption with an average error of 7%, 6% and 12% on the Pascal, Maxwell and Kepler GPUs, respectively. Particularly, in the Maxwell (or Pascal) GPU, the model provides accurate results across a frequency range of $1.6\times$ ($2.4\times$) for the core and $4.3\times$ ($1.2\times$) for the memory frequencies.

The work herein presented opens the possibility for many different future directions, one of which is the implementation of the proposed DVFS-aware power model in real-time. This can be done by taking advantage of the iterative nature of many of the most common GPU applications, by measuring the performance events during the first call to a GPU kernel and then using the power prediction to determine the frequency/voltage configuration that best suits that kernel. Additionally, the proposed model can be used for the development of novel energy-aware GPU simulators and for the energy-optimization of GPU applications.

## Acknowledgment

This work was partially supported by Fundação para a Ciência e a Tecnologia (FCT), under grants SFRH/BD/101457/2014 and UID/CEC/50021/2013.

## References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and Others, "TensorFlow: a system for large-scale machine learning," in *Proc. Conf. Oper. Syst. Des. Implement. (OSDI)*, 2016.

[2] NVIDIA, "NVIDIA, CUDA C Programming Guide v9.0. 2017." 2017. [Online]. Available: http://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html

[3] S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip-multiprocessors," in *Proc. Int. Symp. Low Power Electron. Des. (ISLPED)*, 2007.

[4] X. Mei, L. S. Yung, K. Zhao, and X. Chu, "A measurement study of GPU DVFS on energy conservation," in *Proc. Workshop Power-Aware Comput. Syst. (HotPower)*, 2013.

[5] K. Ma, X. Li, W. Chen, C. Zhang, and X. Wang, "GreenGPU: A Holistic Approach to Energy Efficiency in GPU-CPU Heterogeneous Architectures," in *Proc. Int. Conf. Parallel Process. (ICPP)*, 2012.

[6] R. Ge, R. Vogt, J. Majumder, A. Alam, M. Burtscher, and Z. Zong, "Effects of Dynamic Voltage and Frequency Scaling on a K20 GPU," in *Proc. Int. Conf. Parallel Process. (ICPP)*, 2013.

[7] S. Zhuravlev, S. Blagodurov, and A. Fedorova, "Addressing shared resource contention in multicore processors via scheduling," in *Proc. Int. Conf. Archit. Support Program. Lang. Oper. Syst. (ASPLOS)*, 2010.

[8] S. Che, J. W. Sheaffer, M. Boyer, L. G. Szafaryn, Liang Wang, and K. Skadron, "A characterization of the Rodinia benchmark suite with comparison to contemporary CMP workloads," in *Proc. Int. Symp. Workload Characterization (IISWC)*, 2010.

[9] J. Guerreiro, A. Ilic, N. Roma, and P. Tomás, "Performance and Power-Aware Classification for Frequency Scaling of GPGPU Applications," in *Proc. Workshop Algorithms, Model. Tools Parallel Comput. Heterog. Platforms (HeteroPar)*, 2016.

[10] X. Mei, Q. Wang, and X. Chu, "A survey and measurement study of GPU DVFS on energy conservation," *Digital Communications and Networks*, vol. 3, no. 2, pp. 89–100, 2017.

[11] S. Hong and H. Kim, "An integrated GPU power and performance model," in *Proc. Int. Symp. Comput. Archit. (ISCA)*, 2010.

[12] J. Leng, T. Hetherington, A. ElTantawy, S. Gilani, N. S. Kim, T. M. Aamodt, V. J. Reddi, J. Leng, T. Hetherington, A. ElTantawy, S. Gilani, N. S. Kim, T. M. Aamodt, and V. J. Reddi, "GPUWattch: enabling energy optimizations in GPGPUs," in *Proc. Int. Symp. Comput. Archit. (ISCA)*, 2013.

[13] S. Song, C. Su, B. Rountree, and K. W. Cameron, "A Simplified and Accurate Model of Power-Performance Efficiency on Emergent GPU Architectures," in *Proc. Int. Symp. Parallel Distrib. Process. (IPDPS)*, 2013.

[14] Y. Abe, H. Sasaki, S. Kato, K. Inoue, M. Edahiro, and M. Peres, "Power and performance characterization and modeling of gpu-accelerated systems," in *Proc. Int. Parallel Distrib. Process. Symp. (IPDPS)*, 2014.

[15] G. Wu, J. L. Greathouse, A. Lyashevsky, N. Jayasena, and D. Chiou, "GPGPU performance and power estimation using machine learning," in *Proc. Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2015.

[16] J. A. Stratton, C. Rodrigues, I.-J. Sung, N. Obeid, L.-W. Chang, N. Anssari, D. Geng, W.-M. Liu, and W. Hwu, "Parboil: A Revised Benchmark Suite for Scientific and Commercial Throughput Computing," Center for Reliable and High-Performance Computing, Tech. Rep., 2012.

[17] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," in *Proc. Int. Symp. Workload Characterization (IISWC)*, 2009.

[18] L.-N. Pouchet, "Polybench: The polyhedral benchmark suite," *URL http//www.cs.ucla.edu/˜pouchet/software/polybench/*.

[19] NVIDIA, *NVIDIA, GPU Computing SDK.*, 2017. [Online]. Available: https://developer.nvidia.com/cuda-code-samples

[20] R. A. Bridges, N. Imam, and T. M. Mintz, "Understanding GPU Power: A Survey of Profiling, Modeling, and Simulation Methods," *ACM Comput. Surv. (CSUR)*, 2016.

[21] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: methodology and empirical data," in *Proc. Int. Symp. Microarchitecture (MICRO)*, 2003.

[22] R. Gonzalez, B. Gordon, and M. Horowitz, "Supply and threshold voltage scaling for low power CMOS," *IEEE J. Solid-State Circuits (SSC)*, vol. 32, no. 8, pp. 1210–1216, 1997.

[23] J. Butts and G. Sohi, "A static power model for architects," in *Proc. Int. Symp. Microarchitecture (MICRO)*, 2000.

[24] H. Wong, M.-M. Papadopoulou, M. Sadooghi-Alvandi, and A. Moshovos, "Demystifying GPU microarchitecture through microbenchmarking," in *Proc. Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, 2010.

[25] X. Mei and X. Chu, "Dissecting GPU Memory Hierarchy Through Microbenchmarking," *IEEE Trans. Parallel Distrib. Syst. (TPDS)*, vol. 28, no. 1, pp. 72–86, jan 2017.

[26] A. Lopes, F. Pratas, L. Sousa, and A. Ilic, "Exploring GPU performance, power and energy-efficiency bounds with Cache-aware Roofline Modeling," in *Proc. Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, 2017.

[27] J. Haj-Yihia, A. Yasin, Y. B. Asher, and A. Mendelson, "Fine-Grain Power Breakdown of Modern Out-of-Order Cores and Its Implications on Skylake-Based Systems," *ACM Trans. Archit. Code Optim. (TACO)*, vol. 13, no. 4, pp. 1–25, dec 2016.

[28] NVIDIA and Microsoft, "Graphics Accelerated Productivity For Every User, Any Application," 2016. [Online]. Available: http://images.nvidia.com/content/grid/pdf/microsoft-server-solution.pdf

[29] J. Guerreiro, A. Ilic, N. Roma, and P. Tomas, "Multi-kernel Auto-Tuning on GPUs: Performance and Energy-Aware Optimization," in *Proc. Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. (PDP)*, 2015.

[30] D. Hackenberg, R. Schone, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer, "An Energy Efficiency Feature Survey of the Intel Haswell Processor," in *Proc. Int. Parallel Distrib. Process. Symp. Workshop (IPDPSW)*, 2015.

[31] X. Ma, M. Dong, L. Zhong, and Z. Deng, "Statistical power consumption analysis and modeling for GPU-based computing," in *Proc. Workshop Power-Aware Comput. Syst. (HotPower)*, 2009.

[32] J. Chen, Bin Li, Ying Zhang, L. Peng, and J.-k. Peir, "Statistical GPU power analysis using tree-based methods," in *Proc. Int. Green Comput. Conf. and Workshops (IGCC)*, 2011.

[33] Y. Zhang, Y. Hu, B. Li, and L. Peng, "Performance and Power Analysis of ATI GPU: A Statistical Approach," in *Proc. Int. Conf. Networking, Archit. Storage (NAS)*, 2011.

[34] S. Ghosh, S. Chandrasekaran, and B. Chapman, "Statistical modeling of power/energy of scientific kernels on a multi-GPU system," in *Proc. Int. Green Comput. Conf. and Workshops (IGCC)*, 2013.

[35] J. Lim, N. B. Lakshminarayana, H. Kim, W. Song, S. Yalamanchili, and W. Sung, "Power Modeling for GPU Architectures Using McPAT," *ACM Trans. Des. Autom. Electron. Syst. (TODAES)*, vol. 19, no. 3, pp. 1–24, jun 2014.

[36] V. Adhinarayanan, B. Subramaniam, and W.-C. Feng, "Online Power Estimation of Graphics Processing Units," in *Proc. Int. Symp. Clust. Cloud Grid Comput. (CCGrid)*, 2016.

[37] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, and S. Matsuoka, "Statistical power modeling of GPU kernels using performance counters," in *Proc. Int. Green Comput. Conf. and Workshops (IGCC)*, 2010.

[38] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong, and T. M. Aamodt, "Analyzing CUDA workloads using a detailed GPU simulator," in *Proc. Int. Symp. Perform. Anal. Syst. Softw. (ISPASS)*, 2009.

[39] R. Nath and D. Tullsen, "The CRISP performance model for dynamic voltage and frequency scaling in a GPGPU," in *Proc. Int. Symp. Microarchitecture (MICRO)*, 2015.

[40] A. Majumdar, L. Piga, I. Paul, J. L. Greathouse, W. Huang, and D. H. Albonesi, "Dynamic GPGPU Power Management Using Adaptive Model Predictive Control," in *Proc. Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2017.