Accurate and Convenient Energy Measurements for GPUs: A Detailed Study of NVIDIA GPU's Built-In Power Sensor

Zeyu Yang
Dept. of Engineering Science
University of Oxford
zeyu.yang@eng.ox.ac.uk

Karel Adamek
Dept. of Engineering Science
University of Oxford
karel.adamek@eng.ox.ac.uk

Wesley Armour
Dept. of Engineering Science
University of Oxford
wes.armour@oerc.ox.ac.uk

***Abstract*—GPU has emerged as the go-to accelerator for HPC workloads, however its power consumption has become a major limiting factor for further scaling HPC systems. An accurate understanding of GPU power consumption is essential for further improving its energy efficiency, and consequently reducing the associated carbon footprint. Despite the limited documentation and lack of understanding, NVIDIA GPUs' built-in power sensor is widely used in energy-efficient computing research. Our study seeks to elucidate the internal mechanisms of the power readings provided by nvidia-smi and assess the accuracy of the measurements. We evaluated over 70 different GPUs across 12 architectural generations, and identified several unforeseen problems that can lead to drastic under/overestimation of energy consumed, for example on the A100 and H100 GPUs only 25% of the runtime is sampled. We proposed several mitigations that could reduce the energy measurement error by an average of 35% in the test cases we present.**

***Index Terms*—High performance computing, Green computing, Energy consumption, Energy measurement, Power measurement.**

I. Introduction

GPUs are now critical components in computer systems, with NVIDIA as the leading player in the GPU market. Six out of the ten most powerful supercomputers in the TOP500 List [34] use NVIDIA GPUs, together they consist of 82,240 GPUs. NVIDIA holds over 80% market share in the discrete desktop GPU [8] and over 90% in the data center GPU market [27]. NVIDIA's market influence is further highlighted by its 2 trillion dollar market cap achieved in February 2024.

Under the widespread usage of GPUs, a glaring issue that emerges is the extensive power consumption. Data centers were estimated to consume 1% of global electricity [19] and contributed 0.3% of emissions [15] in 2018. Frontier, the first and only exascale supercomputer, consumes 23 MW of power. This highlights the critical need for energy efficiency.

A significant workload for GPUs is Machine Learning. GPT-4 was speculated to be trained on 25,000 A100 GPUs for 100 days [36]. This translates to 21GWh of electricity used by the GPUs alone. ML workloads accounted for 10-15% of Google's total energy usage from 2019 to 2021 [26]. While the proportion seems constant over time, Google's total energy consumption has been growing steadily by 20% each year, from 12,750 GWh in 2019 to 22,289 GWh in 2022 [11]. With AI/ML becoming ever more popular, energy consumption will continue to grow. Moreover, the computational power required for a widespread autonomous vehicle adoption would require as much electricity as all current data centres combined [33].

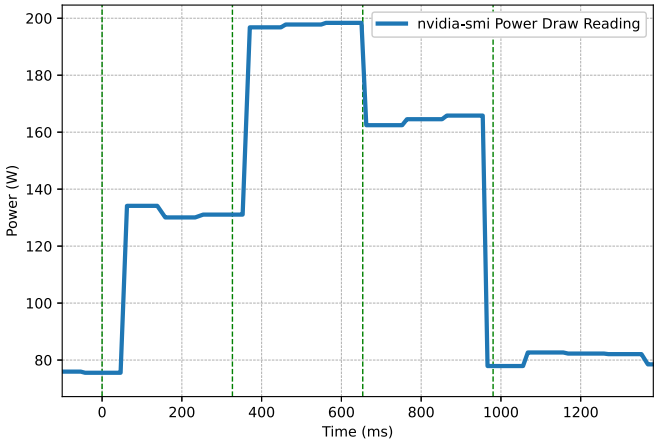

Fig. 1. nvidia-smi can report drastically different power draw for the same CUDA Kernel, ranging from 80W to 200W. This is because nvidia-smi does not fully capture the power information on some GPUs. The figure shows a CUDA program that runs for 325ms on an A100 GPU. The kernel is executed 4 times, and the green dotted line indicates the beginning of each iteration.

Higher energy efficiency yields significant advantages across various domains: lower overall energy use, reduced peak power draw, and decreased heat output. In data centers these reductions translate into lower operational costs for both power and cooling expenses, and consequently, a smaller carbon footprint. Simpler power and cooling solutions facilitate easier performance scaling, while lower operating temperatures boost device stability and longevity. For portable devices, improved energy efficiency can mean greater compute power and extended battery life in a more compact design, improving user experience [39]. Possibilities also opens for deployment in previously untenable scenarios due to power limitations.

Nonetheless, the prerequisite for enhancing energy efficiency is an accurate grasp of energy consumption. Research indicated that optimising algorithms based on inaccurate en-

SC24, November 17-22, 2024, Atlanta, Georgia, USA
979-8-3503-5291-7/24/$31.00 ©2024 IEEE

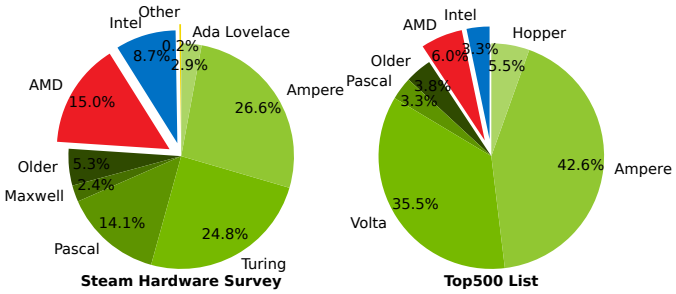

Fig. 2. GPU market share among Steam users [37] and Top500 List [34]. NVIDIA dominates both the Gaming and HPC market. Moreover, older architectures like the 'Turing', 'Volta', and 'Pascal' GPUs, launched between 2016 and 2018, still contribute a significant portion to the total market share.

ergy data can paradoxically lead to an 84% increase in energy usage [10]. Thus, to foster research and development in energy efficiency, an easily accessible, convenient, reliable, and precise method for measuring power is critically needed.

NVIDIA GPUs' onboard power sensors have the potential to fulfill these criteria. Yet, a significant gap in understanding their internal mechanisms hampers researchers' confidence in using data obtained from them. For example, as shown in Fig. 1, nvidia-smi reports drastically different power draw readings for running the same CUDA kernel.

This paper aims to investigate critical questions about the onboard sensors: What is being measured (over what time and which components)? How is it measured? How frequently are the readings being taken? And most importantly, how accurate are these measurements? Some of our key findings includes:

- The error in nvidia-smi's power draw is $\pm$5% as opposed to $\pm$5W claimed by NVIDIA. On modern GPUs capable of drawing 700W this could lead to a $\pm$30W of over/underestimation. For a data centre with 10,000 GPUs, this would lead to an extra $1 million in electricity cost yearly.
- On the A100 and H100 GPUs only 25% of the runtime is sampled for power consumption, during the other 75% of the time, the GPU can be using drastically different power and nvidia-smi and results presented by it are unaware of this. The situation with the Grace Hopper Superchip is even more pronounced, with the GPU sampling 20%, and the CPU sampling merely 10% of the runtime.
- Naively measure energy consumption using nvidia-smi could underestimate by on average 39.3% and up to 68.6%. We proposed several mitigations that could bring this error down to the 5% intrinsic power measurement electronic component error.

Despite being a 2 trillion-dollar company, NVIDIA has shown reluctance to invest in improving the power measurement accuracy of their products, with newer GPUs having increasingly worse measurement capability, and misleading documentation remaining outdated for a decade. We hope this paper would urge NVIDIA to adopt improved hardware components, rectifying driver issues, providing access to accurate power data, and thoroughly documenting these features; On the other hand, we encourage researchers to adopt good measurement practice and document their measurement methodology when using nvidia-smi. Our investigation seeks to lay the groundwork for a standardised, reliable, and precise approach to power measurement, thereby advancing research and development in the vital area of energy efficiency.

II. Background and Motivation

In this section, we offer background information on electronically how is power being delivered to and measured on the GPU. We evaluate existing methods of power and energy measurement and review various studies that have utilised NVIDIA GPU's onboard power sensors.

A. GPU Power Delivery

PCIe form factor GPUs are powered via two sources. The majority of the power is supplied from the PC Power Supply Unit (PSU) via power cables at 12 volts. The most common are the 6 and 8-pin PCIe power cable, rated for 75W and 150W respectively. As modern GPUs became more power hungry, it is common to see cards with several of these connectors. The newest generation GPU uses the 12-pin PCIe gen 5 power connector, rated for 600W. Most of the data centre (Tesla) GPUs use the 384W rated 8-pin EPS connector instead. The other power source is the PCIe x16 slot itself, capable of supplying 75W, of which 10W is supplied via the 3.3V rail.

B. Ways to measure power

a) External Power Meters: Power is the product of Voltage and Current. While Voltage can be measured by an Analog-to-Digital converter (ADC), Current measurement needs additional steps to convert current to a Voltage representation. One widely used method in low voltage applications is to pass the current through a shunt resistor with low resistance and measure the voltage across it. Using this method, researchers have developed several power meters [4], [17], [30]. NVIDIA themselves have a product called the "Power Capture Analysis Tool", as a part of their reviewer toolkit [31], but it's not for sale. We purchased a similar product by ElmorLabs called "Power Measurement Device" [9] to serve the purpose.

b) Predictive Models: Researchers developed power models that predict GPU energy consumption using a combination of architectural parameters, code analysis and performance counters. Leng et al. [18] proposed a configurable, cycle-level model based on a bottom-up methodology from micro-architectural component parameters. Kandiah et al. [16] expanded upon the work to account for modern GPU architectures and Dynamic Voltage and Frequency Scaling (DVFS). Researchers also tried to use Machine Learning techniques: Nagasaka et al. [20] trained a linear regression model with performance counters as the independent variables, and power consumption as the dependent variable. Barun et al. [5] built theirs based on random forests from 189 different kernels.

c) GPU on-board power measurement: NVIDIA GPUs have evolved from estimation-based to measurement-based onboard power tracking. As detailed in an NVIDIA patent [7], older and less expensive models estimate power consumption by monitoring activity signals, linked to the number of active

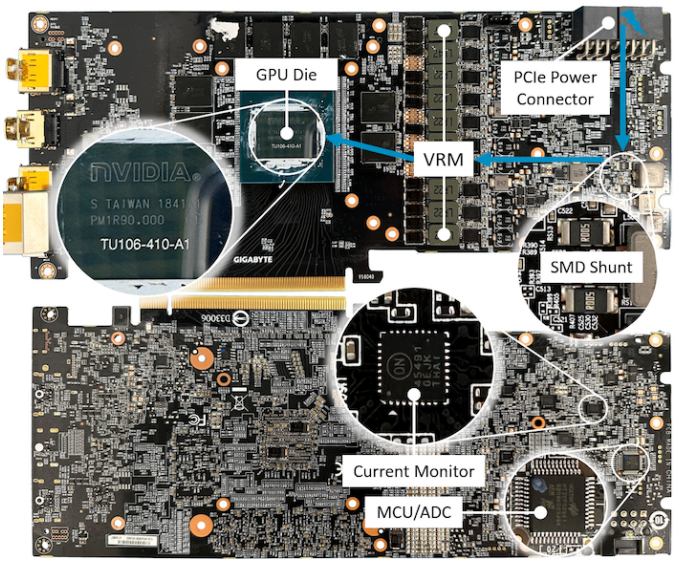

Fig. 3. Front (above) and back (below) image of the PCB of the Gefore RTX 2060 Super GPU (by Gigabyte), as shown by the chip model TU106 on the GPU die. The 5mΩ SMD shunt resistors are located on the front, whereas the ON Semiconductor NCP45491 Current Monitor IC and the HOLTEK HT32F52241 32-Bit Arm Cortex-M0+ MCU is located at the back. The blue colored arrows illustrate the flow of eccentricity from the 6/8-pin PCIe power connector, through the shunt resistors, to the VRMs and reaches the GPU die.

flip-flops, from different GPU functional blocks. Newer and higher-end models employ a measurement-based approach using the shunt resistor method.

We disassembled several GPUs to examine the PCB. Fig. 3 shows the PCB of a RTX 2060S GPU that we disassembled. We observed the existence of shunt resistors and current monitor ICs on GPUs from 'Kepler' onward. For GPUs from the 'Fermi' architecture and some other less expansive cards (e.g. Quadro K620) we found no evidence of those components.

C. nvidia-smi

nvidia-smi (NVIDIA System Management Interface) is a command-line utility included in the GPU device driver. It is used for monitoring, managing, and querying GPU statistics. Running the command with the '–help-query-gpu' flag will show all the querying options available.

For power consumption data, on drivers released before March 30, 2023, there was only one option:

1) "<power.draw> The last measured power draw for the entire board, in watts."

However, on drivers that were released after that, two new options were added. The documentation is also modified with more rather confusing information. The 3 options now are:

1) "<power.draw> The last measured power draw for the entire board, in watts. On Ampere or newer devices, returns average power draw over 1 sec. On older devices, returns instantaneous power draw."
2) "<power.draw.average> The last measured average power draw for the entire board, in watts. Only available on Ampere (except GA100) or newer devices."
3) "<power.draw.instant> The last measured instant power draw for the entire board, in watts."

Furthermore, NVIDIA claims all these readings are "accurate to within +/- 5 watts[1]". We will investigate and comment on each of the claims made in NVIDIA's documentation.

While the NVIDIA Management Library (NVML) [23] also provides access to power readings, the documentation is outdated, lacking details on those new power draw options. Nvidia-smi uses NVML internally and output the same data.

D. Advantages of on-board power measurement

The on-board sensors come built into PCB, eliminating extra cost and installation complexities. Power readings are easily accessible via nvidia-smi or NVML, which comes with the driver, bypassing the need for sudo privileges, and complex data conversions, as is necessary for measuring CPUs via Intel RAPL. The hardware and software interface is standard across NVIDIA product lines, ensuring comparability of data between GPUs.

E. Work that used nvidia-smi/NVML to measure power/energy

Researchers have employed power data from nvidia-smi/NVML to develop GPU modelling tools and enhance the energy efficiency of specific algorithms or applications: Kandiah et al. [16] proposed a GPU power model for energy consumption simulation and prediction. Arunkumar et al. [2] investigated the energy efficiency for future multi-module GPUs; Adamek et al. [1] reduced the energy consumption of the Fast Fourier Transform (FFT) algorithm on GPUs using Dynamic Voltage and Frequency Scaling (DVFS); White et al. [38] optimized the energy efficiency of Radio Astronomy signal processing by mixed-precision computing; Henderson et al. [12] studied energy consumption and carbon emission of Machine Learning; Jahanshahi et al. [13] characterised energy efficiency of multi-GPU ML inference servers, and Yu et al. [40] proposed a resource management strategy to save the energy consumption of cloud scale inference services.

In the seven studies reviewed, five merely mentioned using nvidia-smi or NVML for power/energy measurements without detailed methodology. The other two offered some specifics but assumed nvidia-smi as a uniform sampler. Any error margin cited quoted the 5W from NVIDIA's documentation. This situation highlights that researchers may have placed excessive trust in nvidia-smi, overlooking the critical importance of measurement methodology in their analyses.

F. Different measurement needs: Power vs Energy

Average and maximum power consumption is needed when designing the power delivery and cooling for a system, whether it be a data center or a embedded device. Energy consumption over a period of time is needed for people interested in cost of the computation, for example the operating cost of a data center and the battery life of a mobile device. We will evaluate NVIDIA GPU's on-board power sensor and nvidia-smi's ability in both measurement scenarios.

[1]The official documentation (https://docs.nvidia.com/deploy/nvidia-smi/index.html) is very outdated (last updated 2013, as time of writing). For more updated documentation query nvidia-smi by running: `nvidia-smi --help-query-gpu | grep -A 1 ''power.draw''`

TABLE I
LIST OF GPU TESTED

Architecture	Tesla (Data Center)	#	Quadro (Pro W/S)	#	GeForce (Gaming)	#
Hopper	H100	10				
	GH200 480GB	1				
Ada					RTX 4090	1
Ampere	A100 PCIE-40G	4	RTX A6000	10	RTX 3090	5
	A100 PCIE-80G	4	RTX A5000	1	RTX 3070 Ti	1
	A100 SXM4-40G	2				
	A10	1				
Turing			RTX 8000	4	TITAN RTX	4
					RTX 2080 Ti	1
					RTX 2060 S	1
					GTX 1650 Ti	1
Volta	V100 SXM2-16G	4				
	V100 PCIE-16G	1				
Pascal	P100 PCIE-16G	5			TITAN Xp	1
					GTX 1080ti	1
					GTX 1080	1
Maxwell 2.0	M40	1			TITAN X	1
Maxwell 1.0			K620	1	GTX 745	1
Kepler 2.0	K80	1				
Kepler 1.0	K40	1				
Fermi 2.0	M2090	1				
Fermi 1.0	C2050	1				

III. EQUIPMENT AND EXPERIMENTAL SETUP

A. GPU Selection

All other related works individually tested a limited number of GPUs [3], [6], [10], [14], [32], making it difficult to generalise their results for all GPU models. While it's impossible to test all of the different GPU models, our aim was to include GPUs from each different architecture generation, product line-ups, form-factors and manufacturers.

We tested over 25 different GPU models, and over 70 different GPUs. These GPUs span across all 12 architectural generations that supports power management from 'Fermi' to 'Hopper', 3 product lines (Tesla, Quadro, and GeForce), and different form factors (PCIe, SXM and Mobile). Multiple different cards and form-factors were tested for key GPU models: 5 RTX 3090s, 10 H100s and 10 A100s were tested. Among the 10 A100s, 2 of them were the SXM4-40GB variant, 4 were PCIe-80GB, and 4 were PCIe-40GB; among the 5 RTX 3090s, 4 were from Dell and 1 was from EVGA. In addition, we investigated the Grace Hopper Superchip. All the GPUs tested are shown in Table. I.

B. The Power Measurement Device (PMD)

The Power Measurement Device, positioned between the power supply and GPU, channels all GPU currents through its current sensing shunt resistors. A separate PCIe riser is needed to measure power supplied via the PCIe x16 slot, which disconnects the power rails from the motherboard while allowing data flow, and replaces it with a 4-pin EPS connector.

The voltage and current are quantized by a 12-bit (4096-levels) ADC. The voltage range is 0-31V (0.00757V per level), and the current range is 0-200A (0.0488A per level). For a normal operating scenario of 12V and 12.5A, the resolution is 1586 levels for voltage and 256 levels for current. The voltage is rated to $\pm$0.1V and current is rated to $\pm$0.5A. We've measured the shunt resistors to be within 1% using a benchtop multimeter. We've also calibrated the PMD voltage and current measurement using the multimeter that has a 0.5% error on voltage and 1% error on current, hence 1.12% error on power.

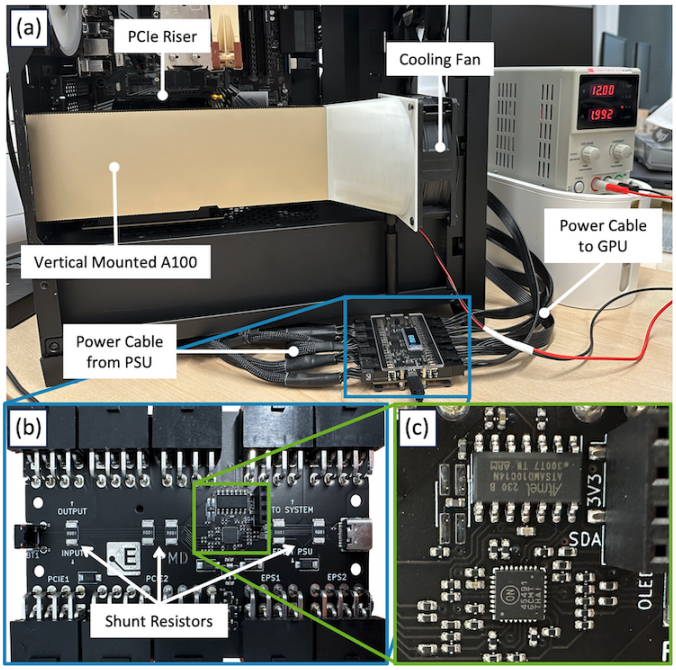

Fig. 4. (a) The test-bench PC with the A100 GPU installed. (b) The PMD PCB, and the location of the 1mΩ shunt resistors. (c) Zoom in on the PCB, showing the same NCP45491 used on many GPUs and a Atmel ARM MCU.

The provided software updates power measurements at 10 Hz and is Windows-compatible only. We developed our own data logger to configure the PMD to send raw data at a baud rate of 921,600 to the host PC for later processing, achieving a 5 kHz sampling frequency.

C. Measurement Setup

Fig. 4a shows the measurement setup for a A100 GPU as an example. The GPU needs to be vertically mounted with a PCIe extension cable to be able to install the PCIe riser PCB. A 24W fan is used with a air duct to cool the passively cooled data centre cards. The test-bench PC has a Intel Core i7-9700K CPU, 64GiB of DDR4 memory, an 850W PSU form Corsair, and runs Ubuntu 22.04 LTS Operating System. All the GPUs that we had physical access to were tested on this test-bench.

D. Benchmark load

The benchmark load is designed to induce high and low GPU power consumption states in a square wave pattern, with the amplitude (power draw), frequency and number of cycles be able to be precisely controlled. This serves as a stress test for probing nvidia-smi's internal mechanisms. The low power state is achieved by a timed sleep, with the duration defining the state's length, while the high power state is achieved through a CUDA kernel performing a data-dependent chain of vector Fused Multiply-Add (FMA) operations. The computation duration is linear to the length of the FMA chain (Fig. 5). To control the duration of this high power

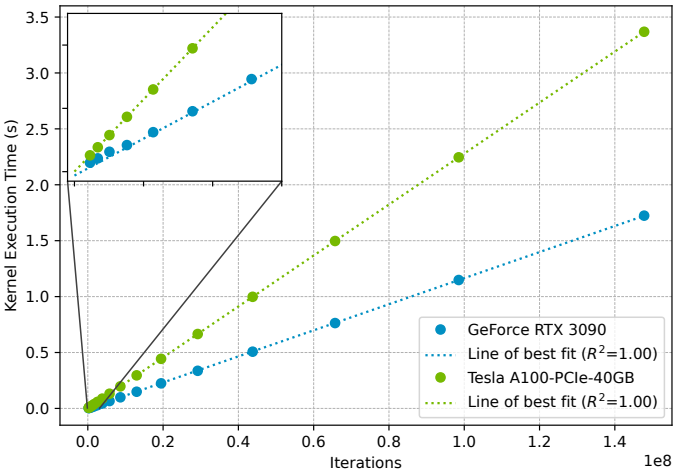

Fig. 5. The number of iterations and kernel execution time demonstrates a linear relationship on both RTX 3090 and A100. Each marker represents a tested iteration count, and the dotted line illustrates the line of best fit. Both models exhibit an R-squared value of 1.000, indicating a perfect fit. The slope of the line is then used to control the duration of the high power state.

state, linear regression was used to determine the gradient between the time measured for a set of arbitrary chain lengths. Amplitude control is achieved by activating different amounts of streaming multiprocessors (SM) of the GPU by setting the number of blocks to be a fraction of the total SM count of the GPU. Fig 8 shows the different power draw levels can be accurately controlled. The code snipped is shown in Listing 1.

```
__global__ void kernel(float *x, int niter) {
    int tid = threadIdx.x + blockDim.x * blockIdx.x;
    #pragma unroll
    for (int i=0; i<niter; i++) {
        x[tid] = x[tid] * 2 + 2;
        x[tid] = x[tid] / 2 - 1;
    }
}
int main(int argc, const char **argv) {
    int nblocks, nthreads, nsize;  float *d_x;
    cudaDeviceProp devProp = getDeviceProperties();
    nblocks = devProp.multiProcessorCount * PERCENT;
    nthreads = devProp.maxThreadsPerBlock;
    if (nblocks < 1) nblocks = 1;
    nsize = nblocks * nthreads;
    // cudaMalloc d_x array on GPU with nsize
    kernel<<<nblocks,nthreads>>>(d_x, DELAY*SCALE);
    cudaDeviceSynchronize();
    usleep(DELAY*1000);
}
```

Listing 1. Benchmark load CUDA code. The CUDA kernel conducts a chain of vector multiplications and additions. Each operation depends on the data from the previous one, ensuring sequential execution. The size of the vector is determined by the product of maximum threads per block and a streaming multiprocessor (SM) count. The number of threads (nthreads) is set to max threads per block, and number of blocks (nblocks) is set to the SM count.

IV. Power Measurement Investigation & Results

We've conducted three experiments to investigate the internal mechanisms of nvidia-smi, with the focus on power measurement. We ran the experiments on the 70+ GPUs we've selected, and present comprehensive and exhaustive results.

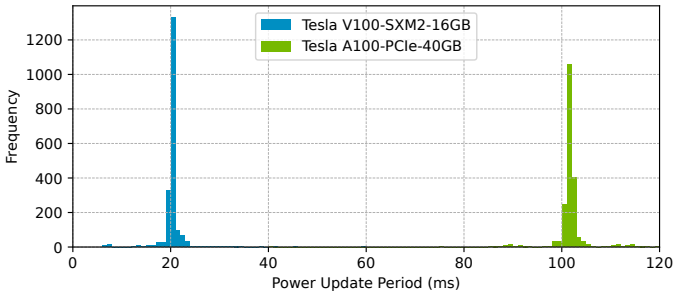

Fig. 6. Histogram of the measured power update period for the V100 and A100 GPU. While there are fluctuations, majority of the update period is at 20ms for V100, and 101ms for A100.

A. Sampling Frequency / Power update Frequency

When querying nvidia-smi, there is the option to specify a querying frequency/period in units of milliseconds. However, it was observed that the power draw reading remains constant for several query samples before updating. We call this the Power Update Frequency/Period. To test this, we ran the benchmark load for a short period of time, with a 20ms square wave period to ensure variating activity on the GPU. Then, the length of each time period where the power draw reading remained the same were counted. We take the median value as the Power Update Period of the GPU. Fig. 6 shows the Power Update Period of a tested V100 and A100 GPU.

If the reported value is simply an sample of the instantaneous power draw, this 10-50Hz sampling frequency is orders of magnitudes slower than the activities on the GPU, typically at a clock frequency of GHz. This indicates that majority of the information would not be captured. This issue is further investigated later in the chapter.

B. Transient Response

The transient response of the nvidia-smi's power measurement was then tested. The step function is generated by the benchmark load with a single period and a duration of 6 seconds. The rise time, defined as the time taken for the power draw to rise from 10% to 90% of max power draw, is measured. When the PMD is available (for the cards we have physical access to), the steady state error are also measured.

The analysis of rise time and delay is crucial for quantifying nvidia-smi's power reading responsiveness to fluctuations in actual power consumption. The steady-state error analysis effectively decouples power readings from time-domain disturbances (e.g. delay and averaging), enabling accurate measurement of the inherent error in power draw values.

a) Rise time and delay: Fig. 7 shows the 4 different kinds of response we observed. The 1st case is the actual power consumption rise nearly instantly, and the nvidia-smi power consumption follows at the next power update interval. The 2nd case is that the actual power consumption takes several hundred milliseconds to rise, however the nvidia-smi power reading is still updated at the next power update interval. In contrast, in the 3rd case, nvidia-smi clearly lags behind the actual power draw, with a linear growth over 1 second. The 4th case is a logarithmic growth over 200 milliseconds.

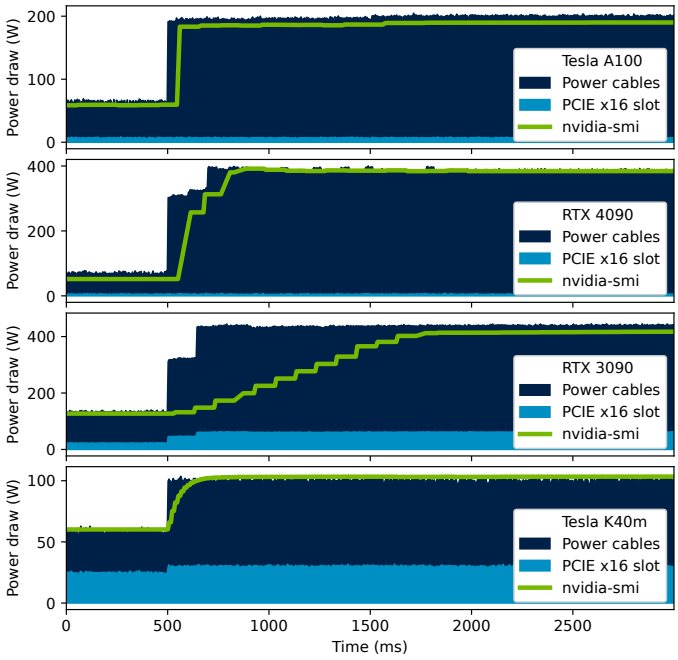

Fig. 7. The 4 different kinds of transient response observed. The benchmark load began execution at 500ms.

The first 2 cases correspond to the <power.draw.instant> option in nvidia-smi, which represents "The last measured instant power draw". Case 3 corresponds to the <power.draw.average> option, which is "The last measured average power draw over 1 sec". The logarithmic growth shown in Case 4 is the <power.draw.instant> option, however only observed on older GPUs of the 'Kepler' and 'Maxwell' generation.

These various responses imply that for some GPUs, if a short program is executed, the measured power are likely to be the activity happened before the program was executed.

b) Steady State Error: The steady state error measures the error when the GPU reaches a constant power draw. Different levels of power consumption were tested from idle to maximum. Fig. 8 shows the result from a RTX 3090. The gradient deviates from 1, indicating the error is proportional, rather than the flat ±5W NVIDIA claimed. We performed this experiment on all the GPUs we had physical access to, and plotted the results in Fig. 9. The tested GPUs, including identical/different models from the different/same manufacturers, showed no clear trends for any specific model or manufacturer.

The error is likely determined by a combination of manufacturer's component quality choice (magnitude of tolerance), and random error within the tolerance. In the majority of the cases the error is within ±5%, which disagrees with NVIDIA's claim that the power draw is with in ±5W. A percentage error makes more sense since the shunt resistor that measures the current will have a resistance tolerance in percentage. Modern GPUs like the H100 has a TDP of 700W, under max load a 5% error leads to a 35W error. If one trusted NVIDIA's documentation, 30W of power draw may be under/overestimated for a single GPU. For data centers with tens of thousands of GPUs, the magnitude of the error would be very significant.

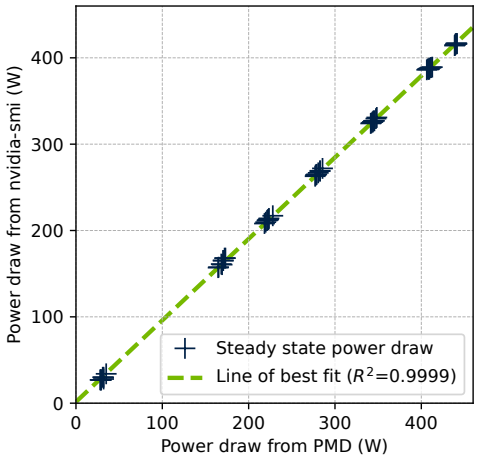

Fig. 8. RTX 3090 steady state power draw from nvidia-smi plotted against PMD. 8 repetitions were tested for each of the 7 different power draw levels: Idle, 1%, 20%, 40%, 60%, 80% and 100% of total SM count, each corresponds to the 7 clusters of points in the plot. The middle 5 clusters are roughly equally spaced apart. The Idle cluster is further apart since its on a lower GPU pstate, and the 100% cluster is closer due to the power limit at 420W.

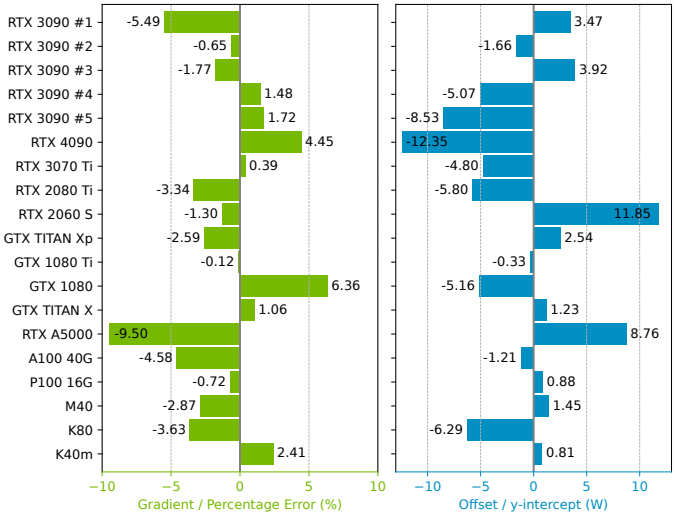

Fig. 9. Gradient and offset of the steady state error of each GPU tested as shown in Fig. 8. In some cases the gradient and offset are in opposite direction, work against each other to reduce the overall error to some extent.

Despite a deviation from a gradient of one, the relationship remains perfectly linear for each GPU itself. When optimizing programs on the same GPUs, the direction of power efficiency improvement will be correct, but with a minor magnitude error.

C. Boxcar Averaging window

To assess if the reported power draw is an instantaneous sample or averaged over the past power update period, we can conduct tests with benchmark loads set to square wave periods (consists of both high and low phases) matching the power update interval. If the power draw reported is instantaneous, we'll see distinct high and low power levels. Conversely, if it's an average, the power draw will be constant, midway between the high and low levels.

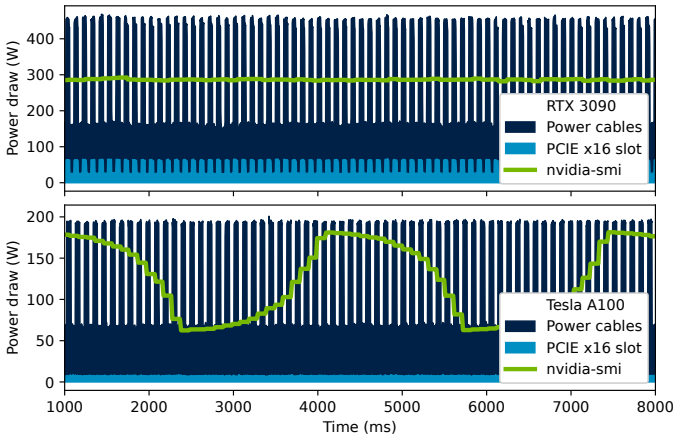

Fig. 10. Running the square wave period with period of 100ms on RTX 3090 and A100. The shaded area ("Power cables" and "PCIE x16 slot") shows the actual 5kSa/s power draw measured by PMD, whereas the green line shows nvidia-smi's reported power. On RTX 3090 nvidia-smi's reported power stays flat in the middle, indicating the boxcar averaging window equals to the power update period. On A100 the nvidia-smi's reported power swings up and down, indicating the boxcar window is a fraction of the power update period.

Fig. 10 shows the results of this test on a RTX 3090 and a A100. On the RTX 3090, nvidia-smi's power draw is constant at 285W, indicating that the reported power draw value is an average of the past power update period. This is essentially a Boxcar averaging over time, whereas the size of the boxcar is the power update period. On the other hand, power draw for the A100 swing between 65W and 180W. The existence of intermediate values indicates that the nvidia-smi's power is not an instantaneous sample, however the boxcar averaging window is not the entire power update frequency either. The best explanation would be that the boxcar window is a fraction of the power update period. We found out that the period of the generated square wave load deviated slightly from exactly 100ms, thus created an aliasing effect between the GPU activity and nvidia-smi's update frequency, hence the fluctuation. Moreover, the starting point of the fluctuation is random among multiple runs because nvidia-smi starts measuring at boot time, and there is no way for the user to control the starting time of the averaging process.

To measure the boxcar averaging window, a inverse approach is employed. A model that emulates the boxcar averaging behaviour is created, taking nvidia-smi's power data and a window size as input. For each nvidia-smi sample timestamp, the average power of the specified window from the PMD data is calculated. The experiment is performed as follows:

1) Set benchmark load square wave period to a fraction of the power update period to create more aliasing effect.
2) Run benchmark load for 9 seconds and collect nvidia-smi and PMD power data.
3) Reconstruct an emulated nvidia-smi data using the model.
4) Discard the first second of data, and normalise both original and emulated nvidia-smi data to only compare the shape.
5) Construct a loss function that calculates the MSE between the original and emulated nvidia-smi power data.
6) Minimise the loss function using Nelder-Mead, with initial power window set as half of the power update frequency.

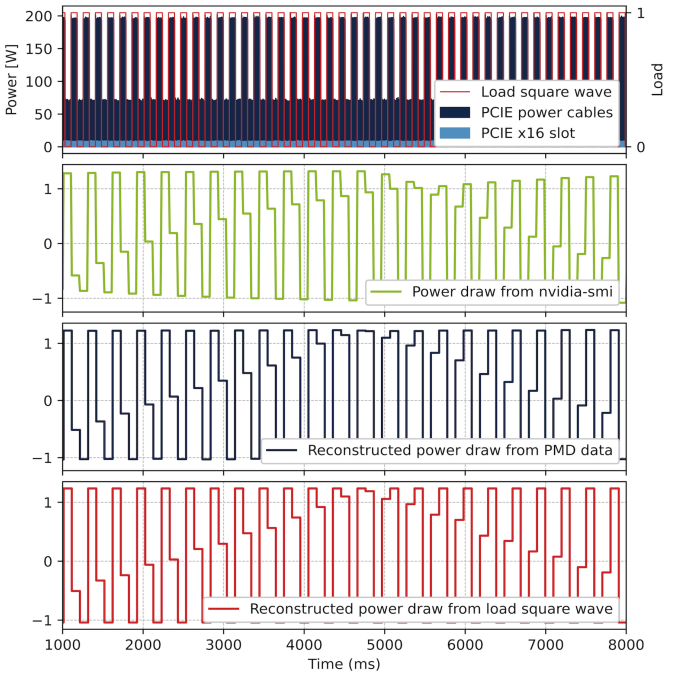

Fig. 11. Running the benchmark load with period equal to 154ms on A100. Top plot shows the power draw data from PMD and the square wave load. The following 3 plots show the original nvidia-smi power data, emulated nvidia-smi data from PMD data, and emulated nvidia-smi data from square wave.

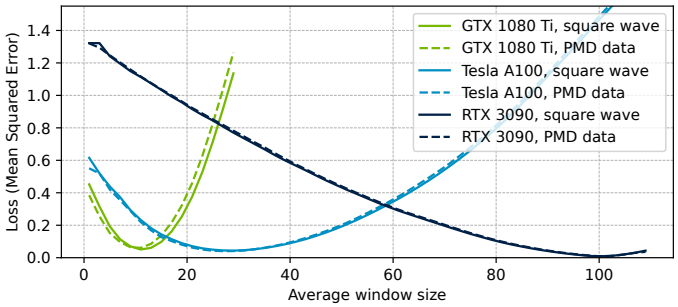

Fig. 12. Loss function of 3 representative GPUs. Minima is the same for emulation of nvidia-smi by either PMD data or Benchmark Load square wave, thus the experiment can be performed on GPUs without PMD attached.

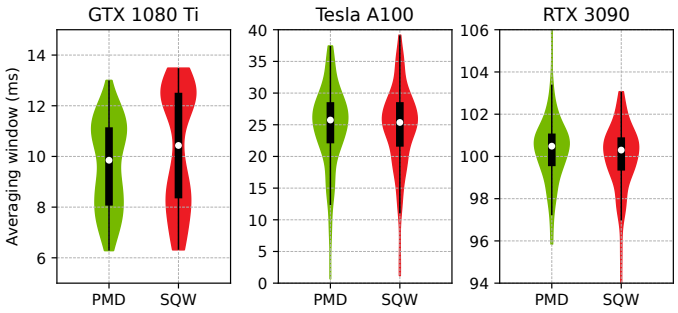

Fig. 13. Violin plot of the experiment results from PMD and Square wave (SQW) data. The white dot shows the median, thick black bar shows the interquartile range, and the thin black line shows the upper and lower adjacent value. Standard Deviation of the PMD/Square wave result for the three GPUs are 1.6/2.4, 3.3/3.2, and 1.2/1.3 respectively.

Fig. 11 shows a particular run of the experiment on the A100. Since the actual power draw reflects the square wave load very well, using the square wave to reconstruct the nvidia-smi power data is nearly the same as using PMD data. This

makes the experiment able to run on any GPU without the need of the PMD device. Fig 12 shows the loss function of a particular run of 3 GPUs: GTX 1080 Ti, RTX 3090 and A100. The loss function suggests that the boxcar averaging window of GTX 1080 Ti is 10ms out of the 20ms power update frequency, A100 is 25/100ms, and RTX 3090 is 100/100ms.

The above experiment is performed 32 times each for 6 different fractions: 2/3, 3/4, 4/5, 6/5, 5/4, and 4/3 of the power update period. Fig. 13 shows the distribution of the results for the three GPUs. The noise in the data collected and outcome of the optimisation caused the results to deviate slightly.

D. Overall Power Measurement Results

The above 3 experiments: Power Update Frequency, Transient Response, and Averaging Window, were performed on all 70 GPUs we've tested. For the 20 GPUs we had physical access to we also investigated nvidia-smi's behaviour across different driver versions. Furthermore, as mentioned in the Background section, to ensure there were no bias:

- Multiple cards were tested for the same model.
- Cards from multiple manufactures were tested.
- Card in different form-factors were tested.
- Same card in different host machines were test.

The manufacturers of the GPU selection includes EVGA, PNY, GiGABYTE, Dell Alienware and Founder Edition cards. 5 RTX 3090s from 2 different manufacturers were tested in different host machines. 10 different H100 PCIE cards were tested, 8 from our university cluster and 2 from a cloud provider. Ten different A100 cards were tested, 2 were the SXM4-40GB version, 4 were the PCIE-40GB version, and 4 were immersion cooled PCIE-80GB cards. A GTX 1650Ti laptop was also tested.

Overall, the error exists in 2 domains, power and time. Error in the power domain is largely due to the intrinsic tolerances of shunt resistors. Error in the time domain comes from the strange behaviours of nvidia-smi's internal data processing.

- Power: The error in power measurements are proportional (roughly $\pm 5\%$) rather than a flat value (± 5W claimed by NVIDIA). On modern GPUs capable to drawing 700W this could lead to a ± 30W of over/underestimation. Comparing the power consumption of different tasks on the same GPUs is still reliable thanks to the near perfect linear relationship. However, since the error margin is random, comparing results from different GPUs would be erroneous. The 30W of additional error may seem insignificant, but can accumulated in computing systems with thousands of GPUs, although statistically the error could (but not guaranteed to) average out for large quantities.
- Energy: Calculated as the product of power and time, energy is not only subject to power measurement errors but also to complications in the time domain arising from nvidia-smi's internal processing. These time-related issues are complex, and we aim to analyse them in the following chapter.

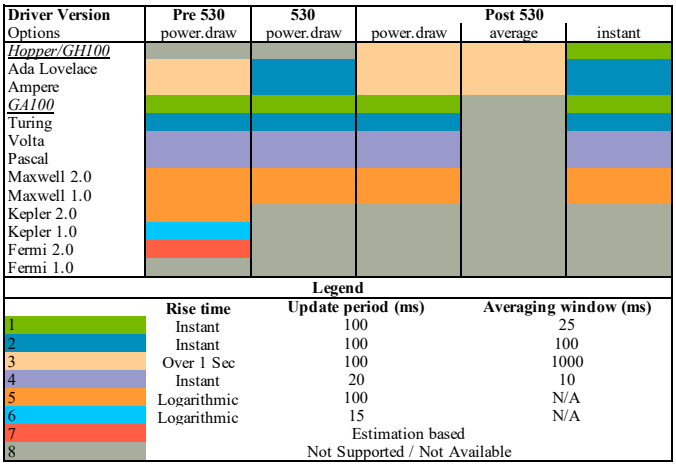

Fig. 14. Results summarised from all tested GPUs with different driver versions. The 'Fermi' generation GPUs either didn't support power measurement or was estimation based. 'Kepler' and 'Maxwell' generation GPUs show a logarithmic growth like transient response, which the power update frequency are different. 'Volta' and 'Pascal' GPUs all have instant rise time, 20ms power update period and 10ms averaging window. 'Turing' GPUs have instant rise time, 100ms update period and 100ms averaging window. The situation becomes more complicated for 'Ampere' and newer generations. A100 cards have a 25ms averaging window for all driver versions. For the rest of other 'Ampere' GPUs and the 'Ada Lovelace' GPUs, on drivers released before March 2023, the averaging window is the past 1 second. On driver version 530 the averaging window changed to 100ms, and after that its switch back to 1 second, whereas the 100ms version is the newly added instant power draw option. Lastly, for the H100, the instant option gives a averaging window of 25ms, while the average and normal option gives an average over 1 second.

V. Energy Measurement Investigation & Results

This section aims to find out the error if naively using nvidia-smi to measure energy, explore a good measurement practice, and verify our results on real world workloads.

A. Exploring Measurement Practice when using nvidia-smi

As seen in the previous section, NVIDIA GPU's on-board power sensor has multiple source of error. Furthermore, these errors are totally inconsistent across different GPU models and driver versions. With the vague documentation from NVIDIA, one may very likely to naively using nvidia-smi to measure energy, as shown in the background section. In reality, one should first understand the behaviour of their GPU, either reference our chart in Fig. 14 or run our benchmark (available on GitHub[2]), and then mitigate the errors accordingly.

Common practice when making measurements is to conduct multiple repetitions of the measurement and take the mean. Undertaking too few repetitions may lead to larger error, while too many cause wasted time and energy. We aim to find out the necessary amount of repetitions for an accurate measurement.

We explore three distinct cases: when the averaging period is the same, longer or shorter than the power update period. The first two cases are examined on the RTX 3090 with the instant and average power draw options. The third case is tested on the A100. For each case we test a short, medium and long benchmark load, corresponding to 25%, 100% and 800% of the power update period. The reason for keep using the

[2]https://github.com/JimZeyuYang/GPU_Power_Benchmark

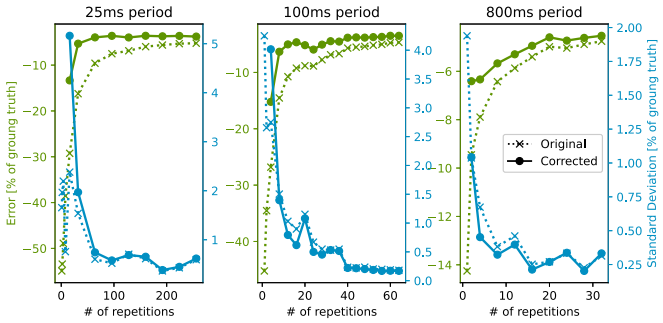

Fig. 15. Case one: The averaging window same as power update period. Results are tested on a RTX 3090 with the <power.draw.instant> option. In this case the averaging window and power update period are both 100ms.

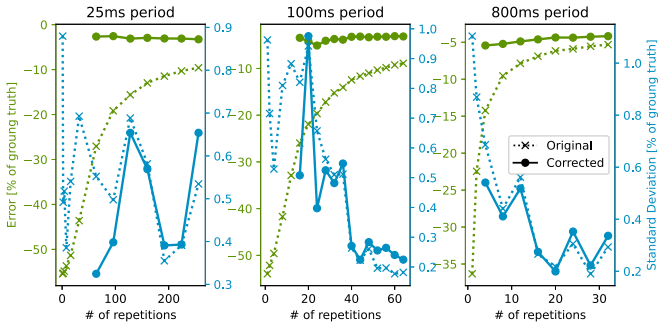

Fig. 16. Case Two: The averaging window is longer than the power update period. Results are tested on a RTX 3090 with the <power.draw.average> option. In this case the averaging window is 1000ms and the power update period is 100ms.

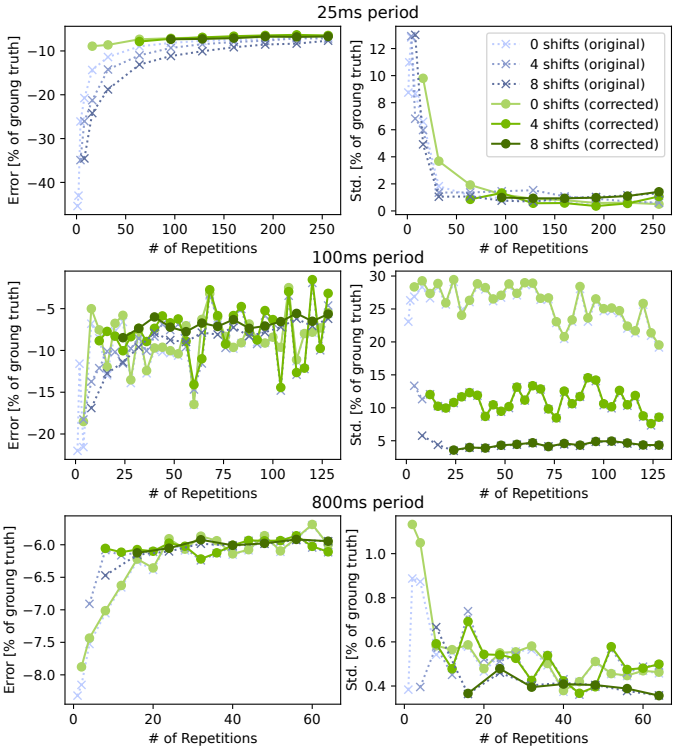

Fig. 17. Case Three: The averaging window is longer than the power update period. Results are tested on a A100 with the <power.draw.instant> option. In this case the averaging window is 25ms out of the 100ms power update period.

benchmark load is because of its controllable nature and the challenges its repeated high/low power states pose to the power sensor, thereby representing a worst-case scenario. We tested a varying number of repetitions for each workload, conducting 32 trials for each, with a random 0-1 second delay in between.

a) Case 1 (Fig. 15): The dotted line illustrate the untouched nvidia-smi results from integrating power over the kernel execution period. Fewer repetitions significantly underestimated the power consumption. As repetition counts increased, the mean error decreased to around -5%, align with the GPU's power draw error margin. The standard deviation also diminished, converging to about 0.5%. This indicates that increasing repetitions improves measurement accuracy to match the hardware's inherent accuracy and enhances precision. However, excessive repetitions don't further improve accuracy or precision and result in wasted time and energy.

The solid line represents energy measurements corrected with insights from the previous section. We excluding initial data within the GPU's 250-millisecond rise time. Given the 100ms averaging window and update period, we shifted the collected power draw data by 100ms to reflect the GPU activities from 100ms earlier. These correction showed that accurate measurements require fewer repetitions and achieve precision more quickly.

b) Case 2 (Fig. 16): This case is highly relevant for most user in upcoming years as its the default setting on most 'Ampere', 'Ada', and 'Hopper' GPUs. A similar trend was observed as for Case 1, where the accuracy and precision increased as the number of repetitions increased. However, due to the gradual ramp-up of power consumption, a much greater number of repetitions are required for the accuracy to increase and converge. Yet, by discarding the initial 1250ms (250ms rise time+1s average time) of data, accuracy levels comparable to Case 1 can be attained with even fewer repetitions.

c) Case 3 (Fig. 17): On H100 and A100, nvidia-smi reports an average of the past 25ms every 100ms, resulting in a 75% information loss. On 'Volta' and'Pascal' GPUs, 50% of information is lost due to the 10/20ms reporting interval. We suggest a strategy that uses controlled delays to shift the 'phase' of GPU activity, allowing different workload segments to be captured within the measurement window.

For a set number of repetitions, i.e. 64, a "0 shifts" test implies 64 straight repetitions. A "4 shifts" test means inserting a 25ms delay after every 16 repetitions, while "8 shifts" means a 25ms delay after every 8 repetitions. We discard the first 100ms of data to account for the GPU's rise time.

In the 25ms load test, activities aligned with the averaging window, showing accuracy and precision similar to earlier cases. For 100ms and 800ms loads, only a quarter of the activity was captured, leading to significant random deviations in measured versus actual power. For the 100ms load, the standard deviation of measurement error reached up to 30%. Introducing controlled 25ms delays, either 4 or 8 times, significantly reduced this deviation to below 5% and stabilised the mean error, enhancing measurement accuracy and precision.

Upon analysing results from three distinct scenarios, we

conclude on the following measurement good practice:

1) Execute the target program for 32 consecutive iterations or until a minimum runtime of 5 seconds is reached. If information loss exists due to a small averaging window, insert 8 controlled delays evenly spaced within the repetitions.
2) Perform separate trials with randomised delay in between.
3) Post process the data by discarding repetitions executed during rise time, and shift data to sync with GPU activity.

B. Benchmark Selection

We aimed to examine the accuracy of using nvidia-smi to measure the energy consumption in real-world workloads. We selected nine benchmarks representing a variety of applications, detailed in Table. II. These were configured to cover a range of execution times and energy consumption levels. The the A100 GPU the execution time taken of these workloads range from 0.69ms to 682ms.

TABLE II
SELECTED BENCHMARKS

Source	Benchmark	Application
CUDA Library	CUBLAS	Linear Algebra
	CUFFT	Signal Processing
	nvJPEG	Image Compression
Domain Specific	Stereo Disparity	Computer Vision
	Black-Scholes	Computational Finance
	Quasi-random Gen	Monte Carlo, etc
MLPerf	ResNet-50	Image Classification
	RetinaNet	Object detection
	Bert	Natural Language Processing

C. Energy Measurement Results

We measured the energy consumption of nine different workloads using a naive method of just running the workload once and take nvidia-smi's reported numbers as granted, and the above-proposed measurement practice for each case. The results were compared with energy calculated using PMD data, and the errors are depicted in Fig. 18. Error in the energy measurement from the naive method highly varies up to 69% randomly among the different workloads and cases. Nonetheless, by following the good practice, the error can be reduced to around 5% across the board. Moreover, the standard deviation of the good practice error across all cases and workloads is 0.25%, indicating that the proposed measurement practice is stable across different GPUs and workloads. Furthermore, these reduced error align with the power measurement error of each GPU. This indicates that errors in the time domain caused by the behaviours of nvidia-smi have been corrected. Despite this, it is impossible to mitigate the error caused by the random component resistance tolerance of a physical shunt resistor on the GPU's PCB.

VI. THE GRACE HOPPER SUPERCHIP EVALUATION

The GH200 Grace Hopper Superchip, NVIDIA's latest flagship, combining a powerful Hopper GPU and a 72-core Arm based Grace CPU in the same package, leveraging a high-bandwidth NVLink interconnect [21]. A key highlight is its

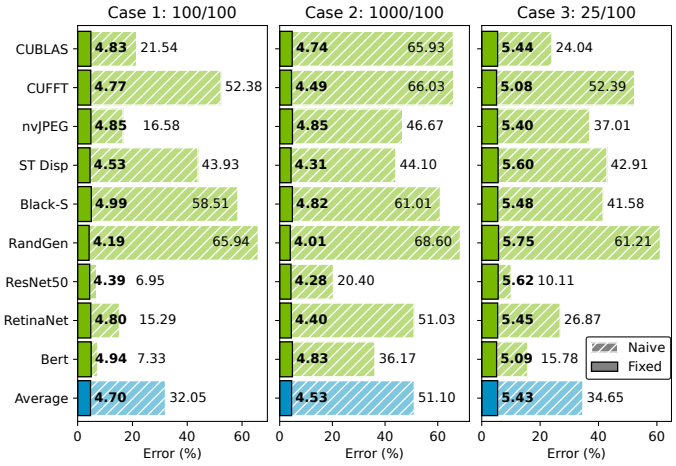

Fig. 18. The error in energy of the 9 selected benchmarks measured from naive approach and good practice compared to using the PMD for the 3 different cases: averaging window is the same as, longer than, or shorter than the power update period. On average, following the good practice can reduce the measurement error by 34.38%, from 39.27% to 4.89%. The standard deviation of the reduced error for each case are 0.25%, 0.28% and 0.21%

power efficiency, attributed to the Arm CPU and LPDDR5X memory. The University of Bristol has secured £220 million to acquire 5,448 GH200 chips [35], and Germany is set to purchase 24,000 GH200 for the JUPITER supercomputer [22]. Given its claimed power efficiency and global demand, it's necessary to assess GH200's power measurement capabilities.

First of all, we found discrepancies between the 'Average' and 'Instant' power query options in nvidia-smi, with the 'Instant' readings consistently exceeding 'Average' under various conditions. Suspecting that 'Instant' might encompass additional system components, we conducted an experiment involving separate and simultaneous CPU and GPU loads, as depicted in Fig. 19. The findings indicate that the 'Instant' measurement, ostensibly representing only GPU, actually reflects the combined power draw of the GPU and CPU.

After evaluating both the GPU and CPU power draw using our benchmark suite, it was observed that both readings update every 100ms. However, the measurement window encompasses only the last 20ms for GPU activity and 10ms for CPU activity. Consequently, 80% of GPU activity and 90% of CPU activity are overlooked. This is arguably another step backwards from the already substantial 75% oversight in power activity measurement found in the A100 and H100 GPUs. Without the insights provided by our benchmark suite, one might assume continuous power monitoring, whereas, in reality, only a fraction (10%) of the activity is being measured.

With the Grace CPU, NVIDIA has introduced power sensors accessible via the ACPI interface to record the average power every 50ms across different GH200 domains [24]. As shown in Fig. 19, we also collected power data from ACPI. The results generally align with those obtained from nvidia-smi, albeit with an anomalously flat power draw profile punctuated by discrete 'noise' fluctuations, exhibiting swings exceeding 100W. Without additional information, it difficult to definitively ascertain the methodology behind ACPI.

Overall, the power measurement capabilities of the Grace

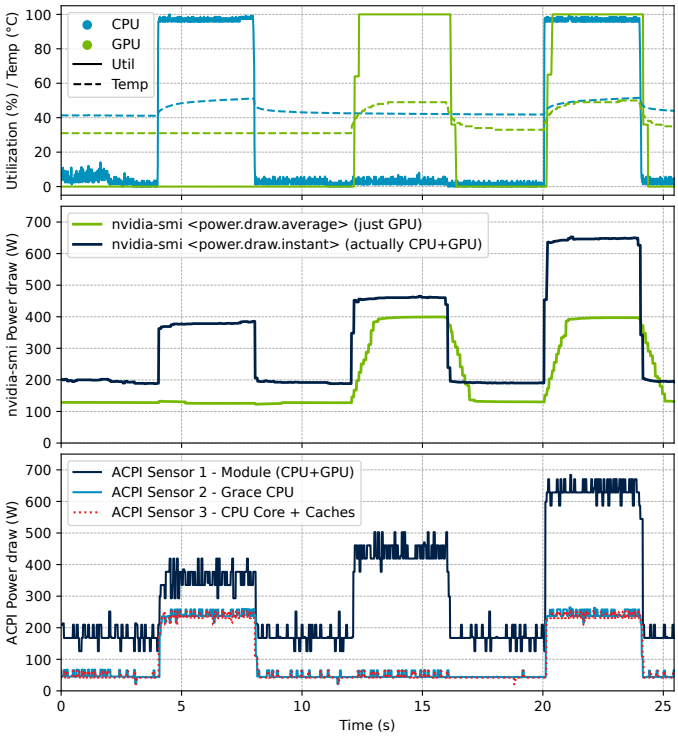

Fig. 19. Experiment of running a CPU and a GPU separately and then simultaneously. TOP: the utilisation and temperature of the CPU and GPU, indicating when the CPU, GPU and both are running. MIDDLE: Power draw output of nvidia-smi. The <Average> power is doing what it should do, but the <Instant> power reacts to both CPU and GPU activity, indicating its a measurement of the whole GH200. BOTTOM: Power draw data of ACPI. The overall shape seems to follow nvidia-smi, however the waveform is overall extremely flat while having seemingly discrete fluctuations over 100W.

Hopper Superchip leave much to be desired. Issues range from misleading power query options to even less reliable 'part-time' power measurement compared to its predecessors, the A100 and H100 GPUs. Additionally, the ACPI power draw data exhibits 'noise' with fluctuations exceeding 100W, further complicating accurate power consumption analysis.

VII. Related Work

Despite the wide usage, research on NVIDIA GPU's on-board power sensors has been limited and outdated. Burtscher et al. [6] were among the first to investigate the GPU on-board power sensors. They tested 2 n-body algorithms on 3 Tesla K20c/m/x GPUs, discovered that the sampling period is 15ms, and the power draw is distorted. We made the same observation on the K40 GPU. They modelled the distortion as capacitor charging, and proposed a method to correct it. Aslan et al. [3] performed a similar experiment on 2 embedded GPUs: Jetson TX2 (Pascal) and AGX Xavier (Volta), and observed the same "capacitor charging" distortion.

Fahad et al. [10] compared the on-board power sensors against a WattsUp Pro external power meter located at the mains plug socket that measures the power of the entire computer. They tested a GEMM and a 2D-FFT load on a K40 and a P100 GPU, and found the error of the energy obtained by NVML can be up to 73% of the external power meter.

Jay et al. [14] also compared a range of software tools that helps the power measurement by interface with NVML internally, i.e. CarbonTracker [12]. They tested several benchmarks on a DGX server with 8 V100s, and compared the results against a Omegawatt external power meter [25] sampling at 1Hz. The focus of this work is rather on qualitatively comparing the functionalities the different software tools.

The GPUs analysed in these studies often included only a few models like the 7-year-old V100 and the 12-year-old K20, and the workload tested were also lacking modern tasks such as AI. In contrast, our research encompasses over 70 GPUs across all product lines, form factors, and 12 architectural generations. The workloads we tested, as shown in Table II, covers a wide set of modern tasks including MLPerf [28].

Furthermore, these works generally assessed apparent measurement results superficially and did not delve into the underlying mechanisms, leading to largely speculative conclusions with limited practical utility. Our study comprehensively reverse-engineered the entire measurement workflow. This enabled us to accurately assess measurement accuracy across different scenarios and offer concrete guidelines for effective use of nvidia-smi in power measurement.

VIII. Conclusion

In this paper, we investigated the mechanisms of NVIDIA GPUs' on-board power sensor. We designed a suite of micro-benchmarks to characterise the behaviour of nvidia-smi, and tested on over 70 GPUs. We also assessed the accuracy of its power and energy measurements against an external power meter. Subsequently, we proposed a measurement standard for ensuring accurate and precise measurements using nvidia-smi. The micro-benchmark is available on GitHub[3], alongside a spreadsheet detailing our GPU test results, which will be updated with future GPU releases. We hope that with a robust understanding of the NVIDIA GPU's on-board power sensor's mechanisms and verified accuracy will empower researchers to use it confidently for accurate power measurements, promoting further research in energy efficient computing on GPUs.

Acknowledgements

WA and KA would like to acknowledge the support received from the STFC Grant (ST/W001969/1). WA would like to acknowledge support from the EPSRC Grant (EP/T022205/1). The authors would like to acknowledge the use of the University of Oxford Advanced Research Computing (ARC) facility in carrying out this work [29]. The authors would like to acknowledge ElmorLabs for their assistance in the use of their product. ChatGPT (Open AI, https://chat.openai.com) was utilized for editing and enhancing the grammar and clarity of this manuscript, as well as for generating portions of the code used in this research.

[3]https://github.com/JimZeyuYang/GPU_Power_Benchmark

References

[1] K. Adámek, J. Novotnỳ, J. Thiyagalingam, and W. Armour, "Efficiency near the edge: Increasing the energy efficiency of ffts on gpus for real-time edge computing," *IEEE Access*, vol. 9, pp. 18 167–18 182, 2021.

[2] A. Arunkumar, E. Bolotin, D. Nellans, and C.-J. Wu, "Understanding the future of energy efficiency in multi-module gpus," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019, pp. 519–532.

[3] B. Aslan and A. Yilmazer-Metin, "A study on power and energy measurement of nvidia jetson embedded gpus using built-in sensor," in *2022 7th International Conference on Computer Science and Engineering (UBMK)*. IEEE, 2022, pp. 1–6.

[4] D. Bedard, M. Y. Lim, R. Fowler, and A. Porterfield, "Powermon: Fine-grained and integrated power monitoring for commodity computer systems," in *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*. IEEE, 2010, pp. 479–484.

[5] L. Braun, S. Nikas, C. Song, V. Heuveline, and H. Fröning, "A simple model for portable and fast prediction of execution time and power consumption of gpu kernels," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 18, no. 1, pp. 1–25, 2020.

[6] M. Burtscher, I. Zecena, and Z. Zong, "Measuring gpu power with the k20 built-in sensor," in *Proceedings of Workshop on General Purpose Processing Using GPUs*, 2014, pp. 28–36.

[7] H. Cha, R. J. H. III, J. A. Robinson, S. J. Treichler, and A. U. Diril, "Power estimation based on block activity," U.S. Patent 8 060 765, nov 2011, [Online]. Available: http://www.freepatentsonline.com/8060765.html.

[8] R. Dow. (2023) Aib shipments climb in q2 2023, with unit sales increasing from quarter to quarter. Accessed: 2023-11-08. [Online]. Available: https://www.jonpeddie.com/news/aib-shipments-climb-in-q2-2023-with-unit-sales-increasing-from-quarter-to-quarter/

[9] ElmorLabs, "Elmorlabs pmd-usb (power measurement device with usb)," 2023, accessed: 2023-07-26. [Online]. Available: https://www.elmorlabs.com/product/elmorlabs-pmd-usb-power-measurement-device-with-usb/

[10] M. Fahad, A. Shahid, R. R. Manumachu, and A. Lastovetsky, "A comparative study of methods for measurement of energy of computing," *Energies*, vol. 12, no. 11, p. 2204, 2019.

[11] Google, "Google environment report 2023," 2023. [Online]. Available: https://www.gstatic.com/gumdrop/sustainability/google-2023-environmental-report.pdf

[12] P. Henderson, J. Hu, J. Romoff, E. Brunskill, D. Jurafsky, and J. Pineau, "Towards the systematic reporting of the energy and carbon footprints of machine learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 10 039–10 081, 2020.

[13] A. Jahanshahi, H. Z. Sabzi, C. Lau, and D. Wong, "Gpu-nest: Characterizing energy efficiency of multi-gpu inference servers," *IEEE Computer Architecture Letters*, vol. 19, no. 2, pp. 139–142, 2020.

[14] M. Jay, V. Ostapenco, L. Lefèvre, D. Trystram, A.-C. Orgerie, and B. Fichel, "An experimental comparison of software-based power meters: focus on cpu and gpu," in *CCGrid 2023-23rd IEEE/ACM international symposium on cluster, cloud and internet computing*. IEEE, 2023, pp. 1–13.

[15] N. Jones *et al.*, "How to stop data centres from gobbling up the world's electricity," *Nature*, vol. 561, no. 7722, pp. 163–166, 2018.

[16] V. Kandiah, S. Peverelle, M. Khairy, J. Pan, A. Manjunath, T. G. Rogers, T. M. Aamodt, and N. Hardavellas, "Accelwattch: A power modeling framework for modern gpus," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, 2021, pp. 738–753.

[17] J. H. Laros, P. Pokorny, and D. DeBonis, "Powerinsight-a commodity power measurement capability," in *2013 International Green Computing Conference Proceedings*. IEEE, 2013, pp. 1–6.

[18] J. Leng, T. Hetherington, A. ElTantawy, S. Gilani, N. S. Kim, T. M. Aamodt, and V. J. Reddi, "Gpuwattch: Enabling energy optimizations in gpgpus," *ACM SIGARCH Computer Architecture News*, vol. 41, no. 3, pp. 487–498, 2013.

[19] E. Masanet, A. Shehabi, N. Lei, S. Smith, and J. Koomey, "Recalibrating global data center energy-use estimates," *Science*, vol. 367, no. 6481, pp. 984–986, 2020.

[20] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, and S. Matsuoka, "Statistical power modeling of gpu kernels using performance counters," in *International conference on green computing*. IEEE, 2010, pp. 115–122.

[21] NVIDIA, "NVIDIA Grace Hopper Superchip Data Sheet," https://resources.nvidia.com/en-us-grace-cpu/grace-hopper-superchip, accessed: 2024-02-21.

[22] NVIDIA, "Nvidia grace hopper superchip powers jupiter, defining a new class of supercomputers to propel ai for scientific discovery," https://nvidianews.nvidia.com/news/nvidia-grace-hopper-superchip-powers-jupiter-defining-a-new-class-of-supercomputers-to-propel-ai-for-scientific-discovery, November 2023, accessed: 2024-02-21.

[23] Nvidia, "Nvml api reference guide," 2023, accessed: 2023-07-31. [Online]. Available: https://docs.nvidia.com/deploy/nvml-api/index.html

[24] NVIDIA, "Grace Hopper Superchip Performance Tuning Guide," https://docs.nvidia.com/grace-performance-tuning-guide.pdf, Feb 2024, accessed: 2024-02-21.

[25] OmegaWatt, "Omegawatt," accessed: 2023-07-27. [Online]. Available: https://mv.omegawatt.fr/en/index.php

[26] D. Patterson, J. Gonzalez, U. Hölzle, Q. Le, C. Liang, L.-M. Munguia, D. Rothchild, D. R. So, M. Texier, and J. Dean, "The carbon footprint of machine learning training will plateau, then shrink," *Computer*, vol. 55, no. 7, pp. 18–28, 2022.

[27] S. Rau, "Worldwide pc, workstation, and server discrete graphics processing unit market shares and market forecast, 3q22," IDC, Pivot Table US48434522, Dec. 2022, accessed: 2023-11-08. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=US48434522

[28] V. J. Reddi, C. Cheng, D. Kanter, P. Mattson, G. Schmuelling, C.-J. Wu, B. Anderson, M. Breughe, M. Charlebois, W. Chou, R. Chukka, C. Coleman, S. Davis, P. Deng, G. Diamos, J. Duke, D. Fick, J. S. Gardner, I. Hubara, S. Idgunji, T. B. Jablin, J. Jiao, T. S. John, P. Kanwar, D. Lee, J. Liao, A. Lokhmotov, F. Massa, P. Meng, P. Micikevicius, C. Osborne, G. Pekhimenko, A. T. R. Rajan, D. Sequeira, A. Sirasao, F. Sun, H. Tang, M. Thomson, F. Wei, E. Wu, L. Xu, K. Yamada, B. Yu, G. Yuan, A. Zhong, P. Zhang, and Y. Zhou, "Mlperf inference benchmark," 2019.

[29] A. Richards, "University of oxford advanced research computing," 2015. [Online]. Available: https://doi.org/10.5281/zenodo.22558

[30] J. W. Romein and B. Veenboer, "Powersensor 2: A fast power measurement tool," in *2018 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 2018, pp. 111–113.

[31] S. Schneider, "Nvidia reviewer toolkit for graphics performance," 2020, accessed: 2023-07-26. [Online]. Available: https://www.nvidia.com/en-gb/geforce/news/nvidia-reviewer-toolkit/

[32] S. Sen, N. Imam, and C.-H. Hsu, "Quality assessment of gpu power profiling mechanisms," in *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2018, pp. 702–711.

[33] S. Sudhakar, V. Sze, and S. Karaman, "Data centers on wheels: emissions from computing onboard autonomous vehicles," *IEEE Micro*, vol. 43, no. 1, pp. 29–39, 2022.

[34] TOP500.org, "Top500 list - november 2023," 2023, accessed: 2023-11-19. [Online]. Available: https://www.top500.org/lists/top500/list/2023/11/

[35] TOP500.org, "Unprecedented £225m investment to create uk's most powerful supercomputer in bristol," Nov. 2023, accessed: 2024-02-22. [Online]. Available: https://www.bristol.ac.uk/news/2023/november/supercomputer-announcement.html

[36] M. Treiber. (2024, July) The secrets of gpt-4 leaked. Accessed: 2024-03-30. [Online]. Available: https://www.ikangai.com/the-secrets-of-gpt-4-leaked/

[37] Valve, "Steam hardware & software survey: June 2023," 2023, accessed: 2023-07-25. [Online]. Available: https://store.steampowered.com/hwsurvey/

[38] J. White, K. Adamek, and W. Armour, "Cutting the cost of pulsar astronomy: Saving time and energy when searching for binary pulsars using nvidia gpus," *arXiv preprint arXiv:2211.13517*, 2022.

[39] Z. Yang, A. B. Clark, D. Chappell, and N. Rojas, "Instinctive real-time semg-based control of prosthetic hand with reduced data acquisition and embedded deep learning training," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5666–5672.

[40] J. Yu, J. Kim, and E. Seo, "Know your enemy to save cloud energy: Energy-performance characterization of machine learning serving," in *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2023, pp. 842–854.

Appendix: Artifact Description/Artifact Evaluation

Artifact Description (AD)

I. Overview of Contributions and Artifacts

A. Paper's Main Contributions

C_1 Nvidia-smi updates a new power reading every 20 to 100 milliseconds.

C_2 Some GPUs responds to a change in power draw at the next power update cycle, whereas others slowly ramp up over a second.

C_3 The error in nvidia-smi's power draw is $\pm 5\%$ as opposed to $\pm$5W claimed by NVIDIA.

C_4 On the A100 and H100 GPUs only 25% of the runtime is sampled for power consumption, during the other 75% of the time, the GPU can be using drastically different power and nvidia-smi and results presented by it are unaware of this. The situation with the Grace Hopper Superchip is even more pronounced, with the GPU sampling 20%, and the CPU sampling merely 10% of the runtime.

C_5 Naively measure energy consumption using nvidia-smi could underestimate by on average 39.3% and up to 68.6%. We proposed several mitigations that could bring this error down to the 5% intrinsic power measurement electronic component error.

B. Computational Artifacts

All https://zenodo.org/doi/10.5281/zenodo.13313030

Artifact ID	Contributions Supported	Related Paper Elements
A_1	C_1	Figures 6, 14
A_2	C_2	Figures 7, 14
A_3	C_3	Figures 7-9, 14
A_4	C_4	Figures 1, 10-14, 19
A_5	C_5	Figures 18 Table II

II. Artifact Identification

A. Computational Artifact A_1

Relation To Contributions

This artifact is the benchmark that tests the GPU's sampling frequency of NVIDIA GPU's on-board power sensors, resulted in contribution [C_1].

Expected Results

Power update frequency should be 100ms for all GPUs apart from "Volta" and "Pascal" generations that is 20ms.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is 2 min.

Artifact Setup (incl. Inputs)

Hardware: A PC or Server that has at least one NVIDIA GPU with power measurement capabilities (Fermi 2.0 generation or newer) is required.

Software:

- Ubuntu 22.04 LTS
- NVIDIA CUDA Compiler (11.8 or newer)
- NVIDIA GPU Driver (any)
- Conda (23.3.1)
- Python (3.10.9)
- Pandas (1.5.2)
- MatPlotLib (3.7.1)

Datasets / Inputs: No Datasets or Inputs required.

Installation and Deployment: Clone the repository from GitHub, build the conda environment from the yml file, and run the tested script following the provided guide.

Artifact Execution

The experiment workflow consists of: running the microbenchmark generating a square wave load with period of 20ms for 10 seconds, and logging nvidia-smi's power readings and timestamps for the duration; finding the durations that consecutive power reading samples remains the same value; take the median of the durations as the power update period of the GPU, and plotting the durations as a histogram for visualisation, as shown in Figure 6 in the paper.

Artifact Analysis (incl. Outputs)

The data processing workflow involves removing consecutively repeated entries in the logged power draw data, calculating the difference of each of the remaining timestamps to get a list of durations, taking the median of the durations as the resulting power update frequency of this GPU, and plotting the durations as a histogram for visualisation, as shown in Figure 6 in the paper.

B. Computational Artifact A_2

Relation To Contributions

This artifact is the benchmark that tests the NVIDIA GPU on-board power sensors reaction to a change in the power draw level, i.e. the transient response, results in contribution [C_2].

Expected Results

The response time is extremely inconsistent among different GPUs and driver versions, as detailed in Fig. 14 in the paper.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is 10min (together with artifact A3).

Artifact Setup (incl. Inputs)

Hardware: A PC or Server that has at least one NVIDIA GPU with power measurement capabilities (Fermi 2.0 generation or newer) is required.

Software:

- Ubuntu 22.04 LTS
- NVIDIA CUDA Compiler (11.8 or newer)
- NVIDIA GPU Driver (any)
- Conda (23.3.1)
- Python (3.10.9)
- Pandas (1.5.2)
- MatPlotLib (3.7.1)

Datasets / Inputs: No Datasets or Inputs required.

Installation and Deployment: Clone the repository from GitHub, build the conda environment from the yml file, and run the tested script following the provided guide.

Artifact Execution

The experiment workflow consists of: running the microbenchmark to generate one period of a square wave load to represent a unit-step function with a period of 3 seconds, and a amplitude of 100% of the GPU's TDP, and logging nvidia-smi's timestamp and power draw; finding the "rise time" by calculating the time it take to rise from 10% to 90% of the final value, and plot the power draw over time for visualisation, as shown in Figure 7 in the paper. This process is repeated 8 times and an average rise time is taken. Then the same workflow is repeated for setting the benchmark to run at 80%, 60%, 40%, 20%, and 1% of the TDP, each repeating 8 times. Finally a average is taking for all the different power draw levels as the rise time of this GPU.

Artifact Analysis (incl. Outputs)

The data processing workflow involves: finding the time it take for the power draw to rise from 10% to 90% of the final value, taking the average of the rise time of the 8 repetitions for each power draw level, and then taking the average of the different power draw levels as the rise time of this GPU.

C. Computational Artifact A_3

Relation To Contributions

This artifact is the benchmark that tests the accuracy of NVIDIA GPU's on board power sensor, results in contribution [C_3].

Expected Results

The expected error should be around +/-5%, rather than the +/-5W claimed by NVIDIA. Depends on the exact GPU the error may be slightly larger or smaller, the deviations are random among different cards, models, and manufactures.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is 10min (together with artifact A2).

Artifact Setup (incl. Inputs)

Hardware: A PC or Server that has at least one NVIDIA GPU with power measurement capabilities (Fermi 2.0 generation or newer) is required. A Power Measurement Device (PMD, https://www.elmorlabs.com/product/elmorlabs-pmd-usb-power-measurement-device-with-usb/) is required to externally capture the power consumption of the GPU.

Software:

- Ubuntu 22.04 LTS
- NVIDIA CUDA Compiler (11.8 or newer)
- NVIDIA GPU Driver (any)
- Conda (23.3.1)
- Python (3.10.9)
- Pandas (1.5.2)
- MatPlotLib (3.7.1)

Datasets / Inputs: No Datasets or Inputs required.

Installation and Deployment: Clone the repository from GitHub, build the conda environment from the yml file, and run the tested script following the provided guide.

Artifact Execution

The experiment workflow is similar to artifact A2, and the experiment is performed together with artifact A2, except the interest is now on when the power draw reached a steady state after the transient. The workflow consist of: the same 8 repetitions for each of the 6 different power draw levels; for each repetition taking an average power draw from 2 second to 3 second for both nvidia-smi's and PMD's readings; visualising the data by plotting the power draw from nvidia-smi against those from PMD, as shown in Figure 8 in the paper; and performing linear regression to find nvidia-smi's power draw error.

Artifact Analysis (incl. Outputs)

The data processing workflow involves taking an power draw average between second 2 to second 3 for each measurement source and each experiment repetition; plotting the the power draw from nvidia-smi against those from PMD; and performing linear regression to find nvidia-smi's power draw error.

D. Computational Artifact A_4

Relation To Contributions

This artifact is the benchmark that found out on certain GPUs only a small portion of the activity on the GPUs are captured for the power consumption reading, results in contribution [C_4].

Expected Results

On the A100 and H100 GPUs only 25% of the runtime is sampled for power consumption. On Grace Hopper Superchip, the GPU is sampling 20%, and the CPU is sampling merely 10% of the runtime.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is 30min.

Artifact Setup (incl. Inputs)

Hardware: A PC or Server that has at least one NVIDIA GPU with power measurement capabilities (Fermi 2.0 generation or newer) is required.

Software:

- Ubuntu 22.04 LTS
- NVIDIA CUDA Compiler (11.8 or newer)
- NVIDIA GPU Driver (any)
- Conda (23.3.1)
- Python (3.10.9)
- Pandas (1.5.2)
- SciPy (1.10.0)
- MatPlotLib (3.7.1)

Datasets / Inputs: No Datasets or Inputs required.

Installation and Deployment: Clone the repository from GitHub, build the conda environment from the yml file, and run the tested script following the provided guide.

Artifact Execution

The experiment workflow consists of running the microbenchmark to generate a square wave load for 9 seconds, with a period of 2/3, 3/4, 4/5, 6/5, 5/4, and 4/3 of the GPU's power update period, each repeating 64 times, and collect power draw and timestamps from nvidia-smi; discarding the first and last second of data, and normalising the data to zero mean unit variance; constructing a mimicking function that mimics nvidia-smi's box-car averaging behaviour from the square waveform, with the box-car averaging window/duration as an variable; constructing a loss function by calculating the Mean Squared Difference between the actual and mimicked nvidia-smi output; minimising the error function using Nelder-Mead method to find the actual box-car averaging window; visualising the result by plotting the averaging window got from all the repetitions in a violin plot, as shown in Figure 13 of the paper; and taking the average as the GPU's averaging window.

Artifact Analysis (incl. Outputs)

The data processing workflow involves, for each repetition: discarding the first and last second of data, and normalising the data; construct a mimicking function; construct a MSE error function; and minimise the error function. Then taking the average of all the repetitions as the averaging window of this GPU.

E. Computational Artifact A_5

Relation To Contributions

This artifact compares the energy measurement accuracy of naively using nvidia-smi to measure a given workload against following our proposed measurement good practice gained from all the previous insights, results in contribution [C_5].

Expected Results

Naively measure energy consumption using nvidia-smi should on average underestimate by 39.3% and up to 68.6%. Following our proposed measurement practice should bring this error down to the 5% intrinsic power measurement error.

Expected Reproduction Time (in Minutes)

The expected computational time for this artifact is 10-min

Artifact Setup (incl. Inputs)

Hardware: A PC or Server that has at least one NVIDIA GPU with power measurement capabilities (Fermi 2.0 generation or newer) is required. A Power Measurement Device (PMD, https://www.elmorlabs.com/product/elmorlabs-pmd-usb-power-measurement-device-with-usb/) is required to externally capture the power consumption of the GPU.

Software:

- Ubuntu 22.04 LTS
- NVIDIA CUDA Compiler (11.8 or newer)
- NVIDIA GPU Driver (any)
- Conda (23.3.1)
- Python (3.10.9)
- HuggingFace-hub (0.16.4)
- Pandas (1.5.2)
- Pytorch (2.0.1)
- TorchVision (0.15.2)
- SciPy (1.10.0)
- MatPlotLib (3.7.1)

Datasets / Inputs: No Datasets or Inputs required.

Installation and Deployment: Clone the repository from GitHub, build the conda environment from the yml file, and run the tested script following the provided guide.

Artifact Execution

The experiment workflow consists of three parts: measuring a ground truth energy consumption using the PMD, measuring a 'naive' result using nvidia-smi, and a 'mitigated' result by following the proposed measurement practice using nvidia-smi. Then a percentage error is calculated against the ground truth for both nvidia-smi results. This process is done for all 9 representative workloads outlined in Table II in the paper.

Measuring the ground truth involves running the workload 16 times; logging the timestamps and PMD power draw data; integrating the power draw over time to get energy consumption, and taking an average of the 16 repetitions as the ground truth power consumption. Measuring the 'naive' power consumption involves running the workload 3 times; logging the timestamps and nvidia-smi power draw data; integrating the power draw over time to get energy consumption, and taking an average of the 3 repetitions as the naive power consumption. The procedure for measuring the 'mitigated' power consumption is described in detail in the paper.

Artifact Analysis (incl. Outputs)

The data processing workflow consists of integrating the power draw over time to get energy consumption for each repetition; taking the average of the repetitions; calculating the percentage error against the ground truth; and plotting the error in a bar chart for visualisation as shown in Figure 18 in the paper.

Artifact Evaluation (AE)

A. Computational Artifact A_1

Artifact Setup (incl. Inputs)

1) Make sure the NVIDIA driver and CUDA toolkit are installed. Check you have them installed by running `'nvidia-smi'` and `'nvcc --version'` respectively. If not install, you could follow NVIDIA's official guide: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/.
2) Make sure conda is installed for bulding the environment.
3) Clone the repository, build and activate the environment:

```
git clone https://github.com/JimZeyuYang/
    GPU_Power_Benchmark.git
cd GPU_Power_Benchmark
conda env create -p ./env -f bin/project_env.yml
source activate ./env
```

Artifact Execution

This artifact runs as a pre-requisite for all other artifacts. To run this you could run artifact 2 for example by:

```
./source/benchmark.py -b -p -e 1
```

The `'benchmark.py'` script process the arguments and create a instant of the object `'GPU_pwr_benchmark'` in the `'GPU_pwr_benchmark.py'` file. The `'__init__'` function of the `'GPU_pwr_benchmark'` object calls the `'_find_pwr_update_freq'` function, which performs this experiment as described in the AD appendix.

Artifact Analysis (incl. Outputs)

The determined power update period will be printed to the standard stream (stdout)/terminal command line. A histogram of the measured periods, akin to Fig. 6 in the paper, can be found at `'./results/GPU_NAME_run_#X/Preparation/find_pwr_update_freq/power_update_freq.jpg'`, where 'GPU_NAME' will be the name of your GPU, and 'run_#X' will be the Xth number of time the experiment has been ran (i.e. the first time you run the code the file name could be `'NVIDIA_GeForce_RTX_3090_run_#0'`).

B. Computational Artifact A_2

Artifact Setup (incl. Inputs)

Same as for Artifact A_1

Artifact Execution

You can run this experiment by running:

```
./source/benchmark.py -b -p -e 1
```

The `'benchmark.py'` script will run the `'run_experiment'` function of the `'GPU_pwr_benchmark'` object, which will run the experiment as described in the AD appendix. Then the `'benchmark.py'` script will run the `'process_results'` function of the `'GPU_pwr_benchmark'` object, which will process and plot the results as described in the AD appendix.

Artifact Analysis (incl. Outputs)

The determined rise-time of the power transient response will be printed to the standard stream (stdout). A plot akin to Fig. 7 of the paper can be found at `'./results/GPU_NAME_run_#X/Experiment_1/X%_load/rep_#X/result.jpg'`. Note that the power measured by the PMD (The dark blue filled part of the plot in Fig. 7) will not show when a PMD is not present.

C. Computational Artifact A_3

Artifact Setup (incl. Inputs)

Same as for Artifact A_1

Artifact Execution

Same as for Artifact A_2, although will only run when a PMD is present.

Artifact Analysis (incl. Outputs)

The determined percentage error of the on-board power sensor compared against the PMD will be printed to the standard stream (stdout). A plot akin to Fig. 8 in the paper can be found at `'./results/GPU_NAME_run_#X/Experiment_1/power.draw.average_power_draw_comparison.jpg'`.

D. Computational Artifact A_4

Artifact Setup (incl. Inputs)

Same as for Artifact A_1

Artifact Execution

You can run this experiment by running:

```
./source/benchmark.py -b -p -e 2
```

The `'benchmark.py'` script will run the `'run_experiment'` function of the `'GPU_pwr_benchmark'` object, which will run the experiment as described in the AD appendix. Then the `'benchmark.py'` script will run the `'process_results'` function of the `'GPU_pwr_benchmark'` object, which will process and plot the results as described in the AD appendix.

Artifact Analysis (incl. Outputs)

The determined box-car averaging window will be printed to the standard stream (stdout). A plot akin to Fig. 10 can be found at `'./results/GPU_NAME_run_#X/Experiment_2/load_Y_ms/rep_#0-31/result.jpg'`, where Y is half of the power update period of the GPU. A plot akin to Fig. 11 can be found at `'./results/GPU_NAME_run_#X/Experiment_2/load_Z_ms/rep_#0-31/result.jpg'`, where `Z` are those 2/3, 3/4, 4/5, 6/5, 5/4, and 4/3 of the GPU's power update period. For the first 5 repetitions (0-4) a loss function is plot, akin to Fig. 12 in the paper. Only the first 5 are plotted in order to save time. A plot akin to Fig. 13 in the paper can be found at `'./results/GPU_NAME_run_#X/Experiment_2/history_length_GPU_NAME.jpg'`.

E. Computational Artifact A_5

Artifact Setup (incl. Inputs)

Same as for Artifact A_1

Artifact Execution

You can run this experiment by running:

```
./source/benchmark.py -b -p -e 5
```

The `'benchmark.py'` script will run the `'run_experiment'` function of the `'GPU_pwr_benchmark'` object, which will run the corresponding 9 different workloads in the `'test'` folder as described in the AD appendix. You may need to recompile these workloads on your own machine by running `'make clean && make`. When a PMD is not present the ground truth energy consumption will not be measured.

Artifact Analysis (incl. Outputs)

The measured energy consumption of 'naive' and 'proposed' will be printed to the standard stream (stdout). Fig. 18 of the paper were plotted using these results.