Exploration of LLM Workload Reliability based on di/dt Effects and Voltage Droops

Zhixing Jiang†, Justin Garrigus†, Allison Seigler†, Ethan Syed†,
Yan-Lun Huang†, Mehdi Sadi‡, Tawfik Rahal-Arabi‡, and Lizy Kurian John†
University of Texas-Austin†, Advanced Micro Devices, Inc.‡
zx.jiang@utexas.edu, justingarrigus@utexas.edu, aseigler@utexas.edu, ethansyed@utexas.edu,
yanlun25635@utexas.edu, mehdi.sadi@amd.com, Tawfik.Arabi@amd.com, ljohn@ece.utexas.edu

***Abstract*—Large language model (LLM) inference workloads have emerged as a critical reliability challenge for cloud GPU systems. Unlike traditional workloads, the highly structured execution of LLMs creates large power oscillations. These oscillations become a vulnerability when their frequency aligns with the resonant modes of a GPU's power delivery network (PDN), leading to excessive voltage droops and unreliable operation. In this work, we present the first comprehensive profiling of LLM-induced power oscillations, revealing that many workloads generate oscillatory patterns in the MHz range—critically aligning with typical GPU PDN resonant frequencies and leading to excessive voltage droops.**

To systematically investigate this phenomenon, we developed a novel stressmark framework that generates workloads with controllable, high-frequency power oscillations and voltage droops. Our evaluation shows that operating at a resonant frequency induces voltage droops up to 2× larger than conventional workloads, exceeding critical noise margins. Critically, we find that real LLM workloads operating even near these frequencies generate significant voltage droops greater than 100mV. Based on these findings, we propose a kernel staggering technique that mitigates this threat by shifting power oscillation frequencies away from resonance frequency, successfully reducing voltage droops and reducing reliability concerns. This work provides the first systematic understanding of LLM-PDN resonance and offers a practical solution to improve GPU reliability in AI cloud environments.

I. Introduction

In the era of hyperscale computing, AI workloads, particularly LLMs, have become a major driver of large-scale GPU deployments in cloud data centers [17], [19], [20]. These models, such as GPT-4 [43], which has 1.8 trillion parameters, require significant computational resources due to their large size and complexity. During inference and training, heavy computational demands can lead to reliability issues [5], [18], [44], [69], [70], such as silent data corruption (SDC) [6], [7], [16], [33], [53], [66], where errors in computation bypass error correction (ECC) mechanisms. These **errors** can stem from sudden voltage noise caused by abrupt or **periodic power fluctuations** during processing [2], [31], [32], [59]. Such instabilities can disrupt computations, requiring checkpointing and recovery to maintain progress [35], [36], [58], [65].

The computational patterns of LLM inference workloads inherently lead to **periodic power fluctuations** in GPUs due to their highly structured and repetitive workflow patterns. For example, the general matrix multiplication (GEMM) operations, which dominate the attention mechanisms in transformer layers, are executed through multi-level nested loops. These loops alternate between different workload intensity levels, leading to periodic changes in computational demand and creating characteristic power oscillations. These oscillatory power patterns, as analyzed in Section IV, are unique characteristics of LLM workloads and significantly impact GPU power dynamics, potentially introducing errors.

A key concern with the **periodic power fluctuations** in LLM inference workloads is their potential to cause power oscillations. If these oscillations occur at or near **the resonant frequency** of the GPU's power delivery network (PDN), they can exacerbate voltage noise and create significant reliability challenges [21], [47]. The resonant frequency is the natural frequency at which the PDN oscillates most efficiently when subjected to periodic disturbances, due to the interplay between its inductive and capacitive elements. Resonance noise arises when periodic fluctuations in current demand (di/dt) align with the PDN's natural resonant frequency, typically in the tens to hundreds of MHz range [23], [32]. This alignment amplifies voltage droops, potentially causing timing violations and errors. Such resonance-driven noise is problematic because repeated large drops can compromise chip reliability, as shown in Section II.

Modern semiconductor technology scaling introduces multiple critical factors that **exacerbate PDN vulnerability**: decreasing supply voltages, increasing frequencies, and increasing die sizes [56]. These scaling effects compound to create more challenging PDN design requirements and increased susceptibility to voltage droop events. Advanced technology nodes require progressively lower supply voltages, scaling from 1.8V in mature nodes to 0.9V in current technologies and toward 0.7V and less in future nodes [45], [55], [68]. This voltage scaling directly impacts PDN sensitivity, as the same absolute voltage droop magnitude represents an increasingly larger percentage of the available noise margin. Concurrent trends in increasing GPU operating frequencies further exacerbate PDN vulnerability. Earlier architectures such as Volta operated at boost clocks around 1400 MHz [39], while Ampere pushed this to approximately 1500 MHz [40], and Hopper further increased boost frequencies up to 1800 MHz in datacenter configurations [41]. Future generations continue to target even higher operating frequencies to sustain greater computational

throughput. Simultaneously, increasing die sizes [56] drive PDN resonant frequencies downward through larger on-die capacitance. These opposing trends create a convergence scenario where GPU operating frequencies and PDN resonant frequencies approach each other, increasing the probability that computational patterns and instruction execution cycles will match resonant modes and trigger voltage instabilities.

The widespread deployment of LLM in GPU clusters has introduced new reliability challenges that existing research has yet to address. While prior work has extensively studied voltage noise in CPUs and traditional GPU applications [31], [32], [49], [50], [51], [52], [59], the vulnerability of LLM to voltage droops remains unexplored, despite their prominence in datacenters. The di/dt effects caused by LLM workloads—particularly oscillatory patterns occurring near PDN resonant frequencies—can amplify GPU voltage noise, potentially leading to severe voltage droops, timing violations, and errors. Existing di/dt stress testing frameworks like AUDIT [29], which were designed for CPUs, cannot be directly applied to GPU architectures, and no voltage mitigation techniques exist for LLM-specific workloads.

To address these, this work makes three key contributions:

(1) We perform a comprehensive characterization of LLM power oscillations using models such as DeepSeekR1 and Gemma3, identifying intra-kernel oscillations in the MHz range. Oscillations of up to 150W are observed in low frequencies, but many oscillations up to 40W range are observed at frequencies close to the GPU's PDN resonance (Section IV).

(2) We present an automated di/dt stressmark generation framework using genetic algorithms that creates controllable oscillatory workloads capable of inducing voltage droops 2× larger than real LLM workloads, enabling systematic characterization of GPU vulnerability (Section V and Section VI).

(3) We propose a warp-level kernel launch staggering technique to mitigate resonance by unaligning kernel launches at the warp level, reducing the worst-case voltage droop by 20% without degrading computational throughput (Section VII).

II. Background

This section reviews GPU PDNs, the link between PDN resonance and voltage droops, and how oscillatory LLM workloads interact with these effects.

GPU PDN Architecture: Modern GPU PDNs consist of voltage regulator modules (VRMs), package-level inductance and resistance, on-die decoupling capacitors, and board-level bulk capacitors [47]. GPUs present unique challenges because of thousands of processing cores that can switch simultaneously, creating substantial current transients that stress the PDN [23], [32]. GPU PDNs are modeled using **ladder RLC networks** with cascaded RLC segments to capture parasitic effects across PCBs, package interconnections, bumps, and the die as shown in Figure 1.

Resonant Frequency Phenomena: For a single RLC segment, resonance occurs when inductive reactance (ωL) and capacitive reactance ($1/\omega C$) exactly cancel each other:

$$\omega_0 L = \frac{1}{\omega_0 C} \quad \Longrightarrow \quad f_0 = \frac{1}{2\pi\sqrt{LC}}. \tag{1}$$

Real-world resonant frequencies in GPU PDNs depend on the specific circuit configuration and component parameters. At the resonant frequencies, the PDN impedance becomes significantly higher compared to other frequencies, representing the PDN's inability to quickly respond to rapid current demands.

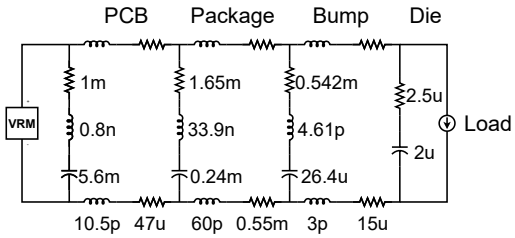


Fig. 1: GPU PDN circuit model with ladder RLC configuration

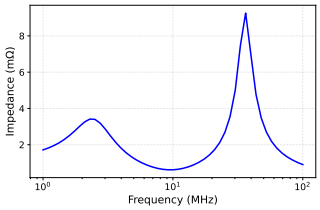


Fig. 2: PDN impedance vs. frequency showing first droop resonant frequency at 35 MHz and second droop resonant frequency at 1.5 MHz.

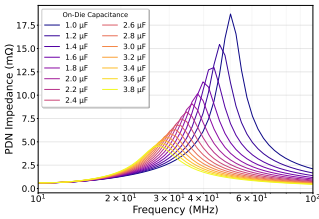


Fig. 3: Relationship between on-die capacitance and PDN resonant frequency, showing the impact of die size scaling.

For example analysis, previous PDN studies [8], [23], [32], [62] and Nvidia A100 specifications [3] are referenced to construct a modeled PDN, as shown in Figure 1. The PDN impedance model is derived using the scaling technique from Leng et al. [32], which adjusts PDN parameters based on the processor's peak thermal design power (TDP).

According to Leng et al., PDN scaling reflects the relationship between processor TDP and package impedance characteristics. In their analysis, they compared the Intel Pentium 4 (50-60 W TDP) to the GTX 480 GPU (200+ W TDP), scaling PDN parameters by 2× rather than the linear 4× TDP ratio due to nonlinear high-performance processor package impedance scaling.

Following this methodology, we scale the A100 GPU PDN model parameters by 4× (half of the 8× TDP ratio between A100's ∼400 W and Pentium 4's ∼55 W). The resulting PDN exhibits two resonant frequencies due to the ladder RLC configuration: 35 MHz (first droop resonant frequency) and 1.5 MHz (second droop resonant frequency), as shown in Figure 2.

Increasing die sizes introduce larger on-die decoupling capacitance, which shifts PDN resonant frequency characteristics toward lower frequencies according to $f_r = \frac{1}{2\pi\sqrt{LC}}$. As die sizes increase and on-die capacitance grows, primary resonant modes shift to lower frequencies that are more easily

excited by software workloads, creating higher susceptibility to voltage droop events.

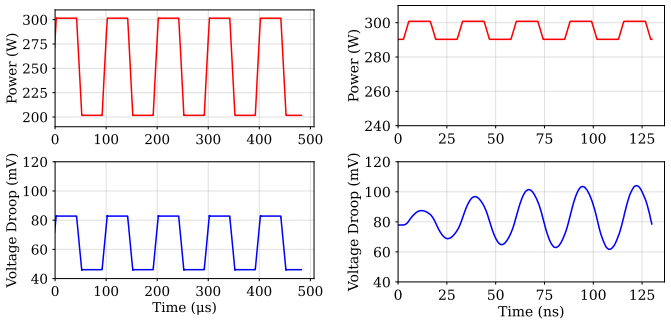
Figure 3 illustrates the inverse relationship between on-die capacitance and resonant frequency. The combination of reduced noise margins, lower resonant frequencies, and higher operating frequencies creates a compounding effect that demands careful PDN design consideration and comprehensive characterization methodologies.

Resonant Frequency Amplification and Voltage Droop Mechanism: The PDN voltage response is highly sensitive to the frequency of power oscillations, with resonant frequencies posing the greatest risk. Variations in GPU power consumption induce time-varying load currents, which interact with the PDN's impedance. Rapid changes in power consumption ($\frac{dP}{dt}$ and $\frac{d^2P}{dt^2}$) generate transient currents that excite the PDN's natural resonances.

When the frequency of these transients aligns with the PDN's resonant frequency, constructive interference amplifies voltage fluctuations. At resonance, the instantaneous voltage at the GPU load node can be expressed as:

$$V_{\text{load}}(\omega) \approx I_{\text{load}}(\omega) \times Z_{\text{PDN}}(\omega), \tag{2}$$

where $Z_{\text{PDN}}(\omega)$ is the frequency-dependent impedance of the PDN. Under resonant conditions, $Z_{\text{PDN}}(\omega)$ increases significantly, amplifying voltage droops. These amplified droops can exceed the GPU's noise margin, potentially leading to instability or degraded performance.

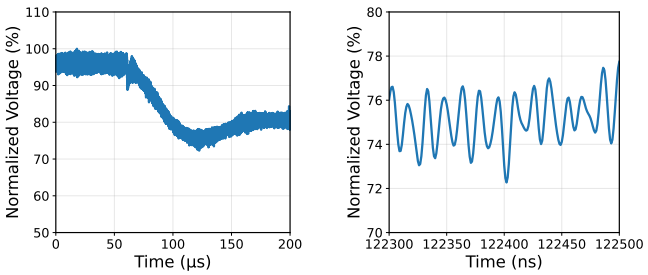


(a) (b)

Fig. 4: Comparison of PDN voltage response to power oscillations at different frequencies: (a) Non-resonant frequency at 10 kHz; (b) Resonant frequency at 35 MHz. The resonant case shows significantly larger voltage droop amplification. Voltage traces start 80 mV below nominal due to static IR droop from high GPU power usage.

We also apply ordinary differential equation (ODE) solvers to solve the relation by applying Kirchhoff's current and voltage laws (KCL, KVL), and power relations to find the current in terms of power, and to observe the Power-Voltage relationship. Figure 4 provides compelling experimental evidence from SPICE simulation of this frequency-dependent vulnerability through direct comparison of power-induced voltage perturbations. At non-resonant frequencies (10 kHz), substantial power variations producing approximately 100 W power differences result in manageable voltage responses, with droop limited to 45–80 mV beyond the static baseline.

At the PDN's resonant frequency (35 MHz), the response becomes catastrophic. A mere 10 W power difference—ten times smaller than the non-resonant case—induces voltage droops exceeding 105 mV within just four oscillation cycles. More critically, the voltage droop exhibits progressive escalation with each subsequent oscillation, demonstrating destructive resonant amplification. This escalating voltage degradation occurs because power oscillations at resonant frequencies create constructive interference with the PDN's natural response, establishing a positive feedback mechanism that amplifies small disturbances into system-threatening voltage instabilities.



(a) (b)

Fig. 5: Normalized GPU rail voltage measurements: (a) view in microseconds showing workload-induced droops, and (b) zoomed view (nanoseconds) around the largest droop, revealing up to 28% voltage reduction.

Data obtained from an industry GPU lab indicates that during execution of modern workloads, voltage droops can reach up to 28%, as shown in Figure 5, reinforcing the severity of resonance-driven instabilities under dynamic operating conditions. Due to the sensitive nature of this data, the droops have been normalized to the nominal supply voltage.

Observation 1

Voltage droops caused by a small power swing (e.g.: 10 W) at resonant frequency are higher and more detrimental than voltage droop caused by a larger power swing (e.g.: 100 W) at a non-resonant frequency.

When voltage droops occur, even though individual events last only nanoseconds, the extended runtime of workloads virtually guarantees that worst-case droop events will happen at some point during execution. Since designers cannot predict customer workloads, they must assume these maximum droops can occur at any time. A single droop causing timing violations can trigger system failure. To prevent this, GPUs employ static voltage guardbanding—raising nominal voltage across the chip. However, a 10% droop margin costs approximately 20% additional power due to quadratic voltage-power relationships. At datacenter scale, this translates to substantial impacts on thermal design, hardware provisioning, and operational costs. While this paper focuses on nanosecond-scale transients, our

techniques apply to analyzing droop resilience across multiple timescales.

III. Experimental Setup

Experiments are conducted on a DGX Linux server running Ubuntu 22.04.5 LTS with an AMD EPYC 7742 64-core processor operating at a maximum frequency of 2.25 GHz. The system features eight NVIDIA A100 GPUs, Ampere architecture, with 40GB memory capacity, at 1410 MHz clock frequency, and a TDP rating of 400 W. All experiments are run with a single NVIDIA A100 GPU, though we expect our results to generalize across multi-GPU environments as well.

Power consumption measurement employs a dual-methodology approach tailored to different frequency ranges and measurement granularities. For low-frequency oscillatory behavior, NVIDIA management library (NVML) provides direct hardware power readings with sampling capabilities appropriate for frequencies within the hardware measurement limitations. AccelSim [26] and AccelWattch [24] simulation frameworks are chosen for high-frequency power trace characterization at MHz level, which enables detailed power profiling beyond the temporal resolution constraints of hardware-based measurement tools.

Voltage droop analysis leverages Cadence Virtuoso for SPICE-based circuit simulation, implementing the complete PDN model depicted in Figure 1.

For this experimental setup, we focus exclusively on real-world LLM inference workloads rather than general-purpose benchmarks. Specifically, we utilize Hugging Face implementations of transformer-based models such as Gemma3 (27B parameters) [61] and DeepSeekR1 Distill Qwen (32B parameters) [4]. This layer-wise approach allows for direct comparison with our generated stressors, capturing the realistic power and voltage behaviors of modern LLM inference.

IV. LLM Power Profiling

LLM workloads exhibit inherent power instability during GPU execution, manifesting as rapid power fluctuations that may coincide with PDN resonant frequencies. Understanding these oscillatory patterns is crucial for characterizing the potential for LLM workloads to inadvertently trigger voltage droop events through resonance effects.

A. Multi-Granularity Oscillation Analysis

1) Hardware-Based LLM Flow Profiling: At the application level, we analyze complete LLM inference workloads using NVIDIA's **TensorRT** framework combined with NVML **hardware monitoring**. Power consumption profiles reveal substantial variations during execution phases across different computational stages.

As illustrated in Figure 6, LLM workloads during inference exhibit abrupt power transitions with significant noise characteristics. During inference execution, power variations reach up to 150W in larger models such as DeepSeekR1 and Google's Gemma3. However, NVML's temporal resolution limitations constrain detailed oscillatory analysis. NVML reports power

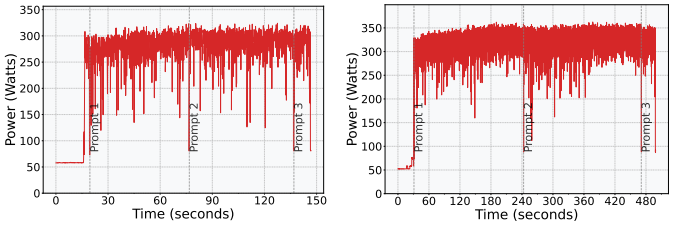


Fig. 6: Power fluctuations during inference across two different LLMs on Nvidia A100 GPU (a) Gemma3 27B; (b) DeepSeekR1 Distill Qwen 32B.

data at approximately 25 ms intervals, corresponding to a sampling frequency of roughly 40 Hz, while PDN resonant frequencies lie in the MHz range—orders of magnitude faster. This discrepancy highlights NVML's limitation in capturing high-frequency oscillations critical to PDN integrity analysis.

Observation 2

LLM inference workloads demonstrate inherent power consumption variability with magnitudes ranging from 50W to 150W during typical execution. The power traces exhibit frequent abrupt transitions, creating the possibility of di/dt effects and voltage droops.

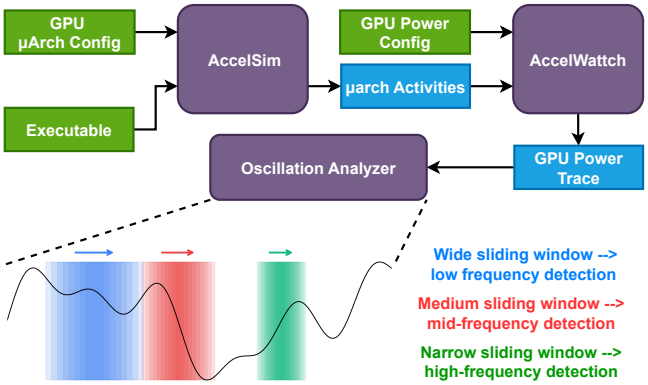
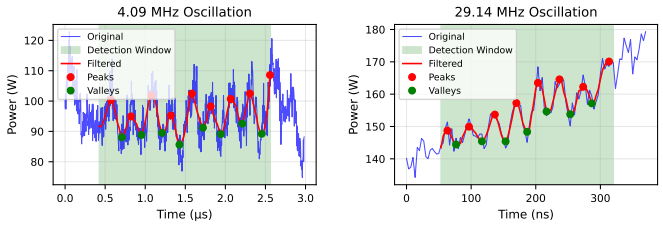


Fig. 7: Overview of the LLM Profiling Flow.

2) Simulation-Based Oscillation Analysis: To overcome hardware measurement limitations and capture high-frequency oscillatory patterns, we develop a comprehensive LLM profiling flow illustrated in Figure 7. The methodology begins with AccelSim, which receives a GPU microarchitecture configuration and a GPU program to generate and simulate detailed microarchitectural activity traces. The simulator integrates with AccelWattch and uses GPU power configuration parameters for power modeling.

Since A100-specific power configuration parameters are unavailable, we calibrate GV100 power configuration parameters in AccelWattch to minimize differences between simulation and hardware measurements for A100 analysis. While this affects absolute power magnitudes, the oscillatory patterns and trends remain representative, as our **primary**

objective focuses on characterizing oscillation behavior rather than precise power values. The resulting power traces are processed through a custom oscillation analyzer employing sliding window techniques to detect periodic patterns across different frequency ranges. The sliding window approach enables frequency-selective analysis—wider windows capture lower frequencies while narrower windows resolve higher-frequency oscillations. Our analysis specifically targets oscillations in the MHz range that overlap with PDN vulnerability frequencies.



(a) (b)

Fig. 8: Kernel-level oscillations observed in the Gemma3 model running on Nvidia A100 GPU: (a) at 4.09 MHz and (b) at 29.14 MHz.

Using the simulation-based profiling flow, we analyze two LLM models of varying complexity: Gemma3 and DeepSeekR1. Kernel-level analysis reveals oscillatory behavior that approaches PDN resonant frequency ranges, as demonstrated in Figure 8.

GEMM kernel execution demonstrates oscillatory patterns at 4.09 MHz as shown in Figure 8 (a). This oscillation likely originates from nested loop structures within GEMM implementations, including outer tile iteration loops processing matrix blocks at regular intervals, thread block scheduling patterns creating periodic resource contention, and memory coalescing behavior producing cyclic access patterns across memory banks.

A second oscillatory component appears at 29.14 MHz as shown in Figure 8 (b), approaching primary PDN resonant frequencies. This high-frequency oscillation may stem from instruction-level execution patterns such as arithmetic pipeline scheduling, fine-grained synchronization barriers within thread block execution, or tensor core scheduling patterns.

These kernel-level oscillations pose significant vulnerability due to their proximity to PDN resonant frequencies. Under Dynamic Voltage and Frequency Scaling (DVFS), variations in GPU clock frequency can align these oscillations with PDN resonant modes, triggering severe voltage droops that compromise system stability.

B. Comparative Kernel Analysis

We analyze oscillation patterns across different kernel types in both Gemma3 and DeepSeekR1 models using AccelWattch power modeling. Figure 9 shows detected oscillation frequencies for representative kernels including attention, GEMM, reduction, and elementwise, with indices representing kernel execution order within each model.

Most oscillations occur from range 10 MHz to 30 MHz, approaching PDN resonant frequencies and potentially amplifying voltage fluctuations. While most power oscillations remain around 10 W or less, some instances reach up to 40 W and persist for more than 6 oscillation cycles, representing particularly dangerous scenarios for power delivery systems.

Both LLM models—Gemma3 and DeepSeekR1—show similar oscillation frequency patterns in GEMM kernels, but the number and amplitude of oscillations vary between individual kernels. Different GEMM kernel types within transformer layers have different runtime characteristics [27]. For instance, in DeepSeekR1, the `020-gemm-3` kernel has more oscillation events than `043-gemm-6`, likely due to factors such as GEMM size, loop structure, or tiling strategy. In some cases, the oscillation amplitude exceeds 35 W. If such oscillations align with the resonant frequency of the PDN, they could contribute to noticeable voltage droop. As discussed in the background section II, a 10 W difference can have a significant impact on voltage droop. In addition to GEMM, other kernels like `016-attention-1` in Gemma3 also show oscillatory patterns, with a frequency near 30 MHz and more than six cycles observed. This frequency is close to typical PDN resonant bands.

Reduction kernels in both models generally produce fewer oscillations, often around 2 MHz, and with relatively small amplitude changes. Elementwise kernels show more varied behavior, with oscillations appearing in both the low MHz and tens of MHz ranges.

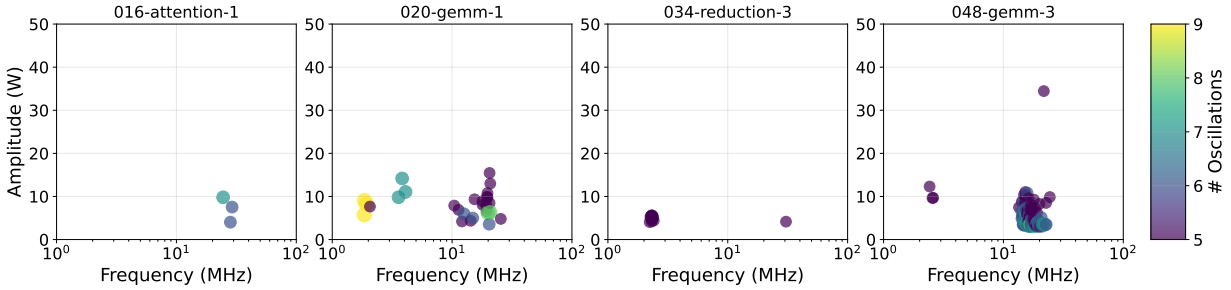
Observation 3

LLM kernel execution consistently generates intra-kernel power oscillations in the MHz frequency range across different model architectures and kernel types. This frequency range critically overlaps with typical GPU PDN resonance frequencies, creating a serious vulnerability to resonance-induced voltage instabilities during normal LLM inference operations.

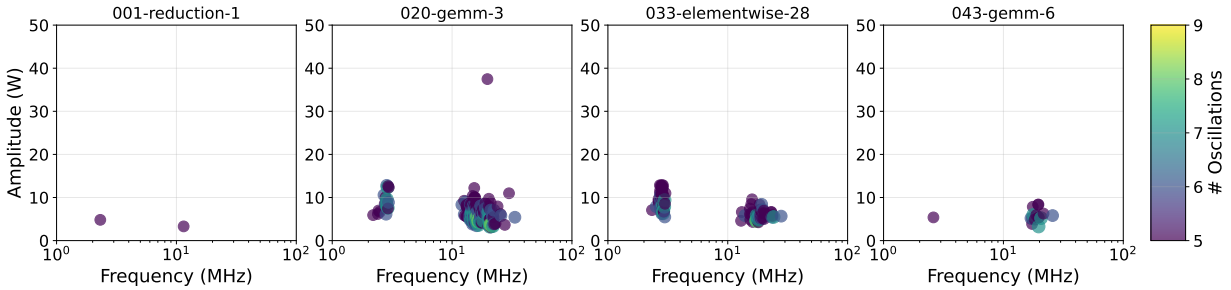
C. Impact of Sequence Length and Batch Size

Gemma3 was profiled end-to-end across varying sequence lengths and batch sizes in Figure 10. The y-axis reports max power, and the x-axis is the sequence length. Batch 1 shows the highest oscillation magnitude on average, likely because less resource utilization across the GPU leads to more pronounced power swings. Higher batch sizes have higher and more consistent utilization across the GPU, reducing average oscillation amplitudes.

Across sequence lengths, no clear monotonic trend in oscillation magnitude is observed. A likely cause is tiling: longer sequences are decomposed into similarly sized tiles (e.g., fixed block/tile shapes for GEMM/attention), so per-tile execution patterns—and their associated power oscillations—remain comparable. As a result, increasing sequence length mainly scales the number of tiles rather than changing per-tile oscil-



(a) Gemma3 27B



(b) DeepSeekR1 Distill Qwen 32B

Fig. 9: Comparison of power oscillation magnitudes across different kernels in (a) Gemma3 and (b) DeepSeekR1 models. Each subplot shows kernels labeled as "x-op-y" where x = global kernel index and y = per-layer operation index. The color indicates the amplitude of power swings detected in each kernel, representing the **magnitude of power fluctuations** rather than absolute power consumption levels.

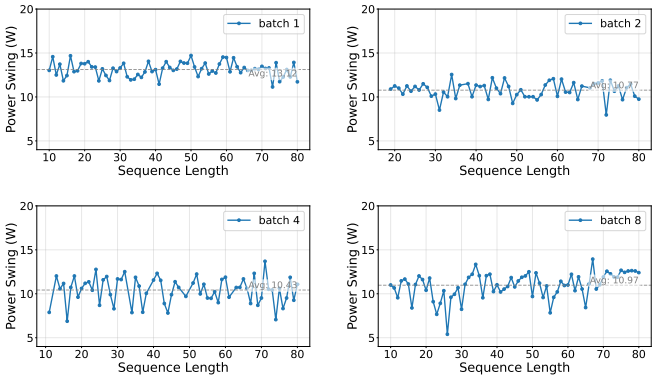


Fig. 10: Power Swing vs. Sequence Length across batch sizes for Gemma3 end-to-end inference. Horizontal gray dashed lines denote the overall average across sequence lengths.

latory behavior, yielding similar oscillation magnitudes across sequence settings.

V. GPU di/dt Stressmark Generation

In Section IV, the analysis revealed that oscillatory behaviors can emerge within LLM kernel execution patterns. However, directly using LLM kernels to study resonant behavior presents significant methodological challenges. LLM kernels provide **limited controllability** over when and how resonant oscillations occur, race conditions introduce substantial timing variability that makes sustained oscillations at precise resonant frequencies extremely difficult to achieve. These factors make it nearly **impossible** to conduct **controlled experiments** that can systematically trigger PDN resonance effects and di/dt stressors such as AUDIT [29] that are required for GPUs.

To systematically study oscillatory behavior in GPU systems under controlled conditions, this work develops a comprehensive **di/dt stressor generation framework** that creates **stable**, **controllable**, and **configurable** oscillatory workloads capable of generating precise oscillation patterns at any specified frequency with varying levels of di/dt stress intensity. Figure 11 illustrates the two-stage methodology for di/dt stressor generation and voltage droop analysis.

Stage 1 (Green Background): Stressor Generation and Power Characterization. The process begins by defining a comprehensive parameter space of knobs that control GPU execution characteristics. A search algorithm systematically explores this space, taking configuration inputs and previous results to generate new parameter combinations. The code generator translates these parameters into executable Compute Unified Device Architecture (CUDA) kernels with controlled oscillatory power patterns. Power consumption is measured in hardware using NVML, and valid stressors spanning the full spectrum of power oscillation magnitudes are stored in a stressor pool.

Stage 2 (Blue Background): Voltage Droop Analysis and Resonance Tuning. The collected stressors are fed into a simulator that converts power traces to current traces using PDN model equations. SPICE simulations of the GPU's RLC power delivery network generate voltage waveforms at the load node.

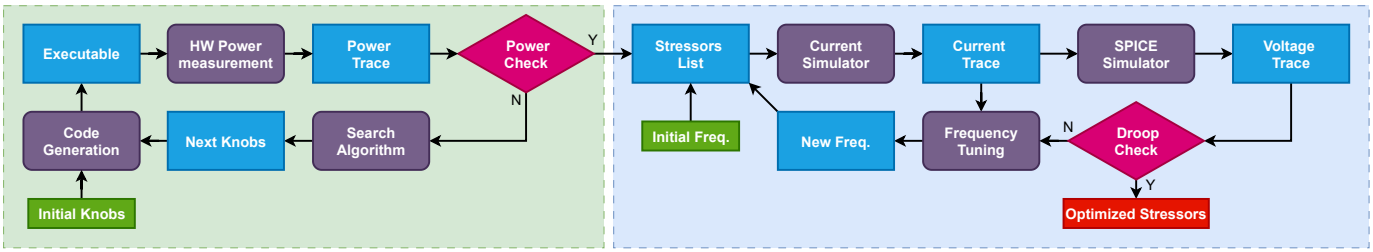


Fig. 11: Overview of di/dt Stressmark Generation.

A droop level check determines if the workload operates at the resonant frequency. If not, the workload frequency adjustment module modifies oscillation timing parameters to tune toward the PDN's resonant frequency. This process iterates until the maximum voltage droop amplitude is achieved.

By using this two-stage methodology, the framework produces a comprehensive library of stressors that operate at resonance and generate different levels of voltage droop. This approach enables systematic analysis of GPU vulnerability to resonance-induced voltage instabilities across varying stress intensities, providing controlled conditions to study phenomena that occur unpredictably in real LLM workloads.

A. Parameter Space Definition and Knob Configuration

To generate effective di/dt stressor workloads, it is essential to understand the computational resources available for power stress generation and define a comprehensive parameter space that controls GPU execution behavior. Modern GPUs contain multiple specialized execution pipelines designed for different types of operations, each with distinct power consumption characteristics and throughput capabilities.

Following the instruction classification approach established in prior GPU power analysis work [54], computational resources are categorized into eight distinct pipeline groups: FMA-Float, FMA-Integer, FMA-EP, ALU-Logic, ALU-Intrinsic, FP64, FP16, and XU. The definitions are in Table I. For each pipeline, four representative instructions are selected based on their frequency of occurrence in LLM workloads.

The parameter space for di/dt stressor generation encompasses 20 distinct knobs that control various aspects of GPU execution behavior, workload composition, and temporal characteristics. Table I summarizes these parameters, their value ranges, and their functional roles in stress generation.

The **execution configuration parameters** in Table I directly control GPU execution scale through GridDim and BlockDim (total active threads) and concurrency through Streams (parallel kernel execution contexts). **Pipeline instruction selection parameters** control computational workload composition. The eight instruction count parameters determine how many operations from each pipeline type are included in the generated kernel. Two array-based parameters provide additional control: pipeline depth arrays manage spacing between same-pipeline instructions, while pipeline ordering arrays determine execution sequencing. **Memory operation control parameters** manage different memory subsystem activation. Five normalized proportion parameters distribute accesses across global, shared, and constant memory operations, while the memory fraction parameter controls the compute-to-memory operation ratio. The **dependency management parameter** controls register dependency spacing, affecting resource reuse efficiency and sustained high-power execution phases necessary for di/dt stress generation.

TABLE I: Parameter space configuration for di/dt stressor generation with 20 knobs.

Knob Name	Range	Definition
Execution Configuration		
GridDim	108-216	Number of thread blocks
BlockDim	256-1024	Threads per block
Streams	1-6	Number of parallel CUDA streams
Pipeline Instruction Selection		
FMA-Float Count	0-5	Fused Multiply Add (FMA) floats
FMA-Int Count	0-5	FMA integer instruction count
FMA-EP Count	0-5	FMA extended precision count
ALU-Logic Count	0-5	ALU logic instruction count
ALU-Intrinsic Count	0-5	ALU intrinsic instruction count
XU Count	0-5	Special function unit instruction count
FP16-Vec Count	0-5	FP16 vector instruction count
FP64 Count	0-5	FP64 instruction count
Pipeline Depth[8]	1-10	Spacing for same pipeline instructions
Pipeline Ordering[8]	0-7	Execution order of pipelines
Memory Operation Control		
Shared Mem Load Prop	0.0-1.0	Shared memory load proportion
Shared Mem Store Prop	0.0-1.0	Shared memory store proportion
Global Mem Load Prop	0.0-1.0	Global memory load proportion
Global Mem Store Prop	0.0-1.0	Global memory store proportion
Constant Load Prop	0.0-1.0	Constant memory load proportion
Memory Fraction	0.0-0.5	Fraction of memory operations
Dependency Management		
Min Dep Distance	1-20	Minimum dependency distance

B. Template-Based Code Generation

The code generator (Fig. 11 left) translates knobs from the search algorithm into CUDA kernels with controlled oscillatory power. Using pipeline instruction parameters, it generates a PTX sequence. PTX, an assembly-like language for NVIDIA GPUs, provides low-level control over pipelines, memory transactions, and synchronization. The generator activates pipelines based on instruction count, depth, and order, while memory parameters manage load/store distribution and dependency controls register allocation and spacing.

The generated instruction mix is embedded into a template kernel following the structure shown in Listing 1. The template provides the oscillation control framework while the generated instructions form the high-power computational payload. For the low-power phase, minimal operations such as register moves (`mov`) or sleep instructions are selected to maintain timing control while minimizing power consumption.

To create oscillatory behavior, the framework uses a timer mechanism to synchronize threads across all streaming multiprocessors (SMs). The implementation can utilize timing sources such as `clock()` or the `globaltimer` register, where each bit position toggles at a distinct frequency to control the speed of the workload oscillation.

Listing 1: Simplified pseudocode for the frequency-controlled GPU stressor.

```
__global__ void stressKernel(args) {
  tid = blockIdx.x * blockDim.x + threadIdx.x;
  declare PTX registers;
  for iters {
    prev_bit = timerCheck();
    // High-power phase
    do {
      asm volatile {
        "mma.sync.aligned.m16n8k8...";
        "ld.global.f32...;";
        // etc... Generated based on knobs
      }
      curr_bit = timerCheck();
    } while (curr_bit == prev_bit);
    prev = curr;
    // Low-power phase
    do {
      asm volatile { "mov.b32..." }
      curr_bit = timerCheck();
    } while (curr_bit == prev_bit);
  }
}
```

The stressor monitors specified bits of the timer to control phase transitions between high-power and low-power phases. Multiple adjacent bits can be combined to achieve finer-granularity control over the phase transition intervals. This allows for more precise adjustments of the workload frequency to match the resonant frequency.

C. Search Algorithm Design

The framework employs a genetic algorithm (GA) using the DEAP library [9] to explore the 20-dimensional parameter space and identify configurations maximizing power consumption while maintaining oscillatory behavior. GA was chosen because prior works [28], [29] successfully used it for oscillation search, which differs from static power search.

Genetic operators are customized for mixed parameter types. Crossover applies two-point methods for scalar parameters and partially matched methods for pipeline ordering arrays. Mutation applies parameter-specific perturbations, ensuring instruction counts, pipeline depths, and memory proportions remain within valid ranges while maintaining normalized constraints for memory operation distributions.

The optimization targets maximum power consumption creating controlled oscillations. Rather than pursuing only peak power, the algorithm maintains a diverse stressor pool spanning the entire power spectrum from minimal to maximal swings, enabling systematic analysis of how power magnitude affects di/dt stress and voltage droop characteristics.

The algorithm terminates when either the generation limit is reached or best fitness remains unchanged for consecutive generations. All evaluated configurations and power measurements are cached to avoid redundant evaluations and preserved for subsequent analysis.

D. Power Measurement

Power consumption measurement relies on NVML to capture GPU power consumption during stressor execution. To analyze oscillatory behavior within the hardware's measurement capabilities, workloads are executed at frequencies that satisfy the Nyquist sampling criterion—operating at twice the sampling frequency to ensure accurate trend detection and magnitude characterization on physical hardware. The workload frequency is determined by a timer-based mechanism, as described in Section V-B.

E. Power-to-Current Conversion

Voltage droop analysis requires converting power traces into load current profiles to model GPU dynamics. This conversion relies on KVL, KCL, and the power equation $P = VI$, which relates power consumption to current and voltage. Together, these principles form a system of ODEs that links time-varying power consumption to load current.

The ODE system incorporates reactive PDN components, such as inductances and capacitances, which introduce temporal dependencies and transient current spikes. Solving these equations requires numerical stability due to the stiff nature of PDN dynamics. For this, we use the SciPy library's backward differentiation formula (BDF) method, which efficiently handles rapid time-scale variations while maintaining computational efficiency for extended simulations.

The resulting current profiles serve as essential inputs for SPICE-based voltage droop analysis, enabling accurate modeling of the PDN's response to dynamic power demands.

F. Current-to-Voltage Conversion and Resonance Detection

Current profiles are input into SPICE-based circuit simulations to analyze the GPU PDN's voltage response, translating time-varying current demands into voltage droop patterns that highlight resonance-induced instabilities.

The SPICE simulation uses the full PDN circuit topology from Figure 1 within Cadence Virtuoso, including voltage regulators, bulk capacitors, package inductances, on-die decoupling elements, and their parasitics. Current injection uses the `ipwlf` (independent piecewise linear current source) component, which accepts CSV-formatted current traces from the power-to-current conversion stage and applies the time-varying profile to the PDN load node.

Resonance detection employs an iterative frequency search to identify operating points maximizing voltage droop amplitude. The algorithm initializes oscillation frequencies near the PDN resonant frequency and varies frequency across a defined range. For each point, the corresponding stressor generates a current profile driving the SPICE simulation to produce voltage response characteristics.

While SPICE frequency sweeps provide the PDN's resonant frequency, iterative search remains necessary because the timer

mechanism controlling workload frequency does not specify exact execution frequency. Slight adjustments around resonance ensure alignment with the PDN's actual resonant mode and maximize constructive interference, refining frequency configuration to produce maximum voltage droop.

This framework enables systematic PDN vulnerability characterization across the frequency spectrum, providing foundation for analyzing how stressor configurations interact with the PDN.

VI. Results and Analysis

A. Comparative Voltage Droop Analysis

This work evaluates both real LLM voltage droop characteristics and stressmark-generated voltage instabilities to provide a comprehensive understanding of GPU PDN vulnerability. The analysis applies the power-to-voltage conversion methodology from Section V to characterize voltage droop behavior across different workload types and synthetic stress conditions.

Figure 12 presents comparative voltage droop analysis across five stressor configurations and LLM workloads from DeepSeekR1 and Gemma3 models. All stressors operate at the PDN's resonant frequency of 35 MHz with varying power swing magnitudes: $\mathbf{S}\alpha$ (116 W), $\mathbf{S}\beta$ (86 W), $\mathbf{S}\gamma$ (77 W), $\mathbf{S}\delta$ (67 W), and $\mathbf{S}\varepsilon$ (49 W). These differences result from varying instruction patterns, detailed in Section VI-C. The stressors demonstrate that resonant frequency operation is extremely detrimental to PDN stability.

LLM workloads exhibit distinct voltage droop patterns across kernel types. GEMM operations generate the largest voltage droops due to high computational intensity and power oscillations near the resonant frequency, with DeepSeekR1 ranging from 45 mV to 103 mV and Gemma3 from 55 mV to 120 mV. Reduction kernels show minimal droops due to low static power and few oscillations near resonant frequencies. Elementwise operations vary substantially (20-105 mV range) as they encompass diverse operations including data copying, quantization, arithmetic, and memory transfers.

The comparative analysis demonstrates that stressors operating at resonant frequencies achieve voltage droop magnitudes approximately **2×** larger than those observed in typical LLM inference kernels, illustrating the severe impact of resonant frequency alignment on PDN stability. However, LLM workloads operating near resonant frequencies can still generate voltage droops exceeding 100 mV, representing a dangerous 10% deviation from the 1 V nominal supply voltage. Prior GPU voltage noise studies [31] establish that 5–12% droops from nominal voltage induce timing violations and computational errors, with larger droops increasing error probability. Our analysis reveals that kernels exceeding the 100 mV critical threshold occur twice per inference run, demonstrating that resonance-induced voltage instabilities are recurring reliability threats in production LLM deployments, not rare edge cases.

Observation 4

The stressor framework demonstrates the critical severity of oscillations at the PDN resonant frequency, which can generate voltage droops from 100,mV to 250,mV. This is critical as analysis of real LLM workloads shows the possibility of droops over 100,mV during execution of many kernels near (even if not exactly at) the resonant frequency.

B. Search Mechanism Analysis

The di/dt stressor generation framework employs a two-stage search methodology to systematically identify optimal stressor configurations and resonant frequency alignment.

Figure 13(a) demonstrates the genetic algorithm convergence behavior for the power swing search from Section V during stressor generation. Starting at approximately 50W power differences, the algorithm reaches maximum capability around generation 15 and maintains stable performance thereafter. Each generation requires 15-20 minutes due to comprehensive power measurement across candidate populations.

Figure 13(b) illustrates the iterative resonant frequency search from Section V across five stressor configurations. Each stressor begins at non-resonant frequencies with minimal voltage droops, then approaches the 35MHz resonant frequency with dramatically increased voltage perturbations. $\mathbf{S}\alpha$, the rightmost column, achieves the highest voltage droops, while $\mathbf{S}\varepsilon$, the leftmost column, produces proportionally smaller perturbations. The transition demonstrates voltage droop amplification factors of 2-3× across all configurations, confirming the critical importance of frequency alignment in PDN vulnerability assessment.

C. Instruction Pattern Impact on Power Oscillation

Analysis of the PTX instruction sequences reveals how specific instruction patterns influence power oscillation magnitude. Stressor $\mathbf{S}\alpha$ contains the highest number of MMA instructions, directly correlating with its larger power variations, while $\mathbf{S}\varepsilon$ contains the fewest.

Instruction density between MMA operations significantly affects tensor core utilization and power patterns. In stressor $\mathbf{S}\varepsilon$, multiple arithmetic and memory operations separate each MMA instruction, creating idle periods where tensor cores await operands. Conversely, stressor $\mathbf{S}\alpha$ maintains minimal instruction gaps between MMA operations, enabling continuous pipeline utilization. This dense MMA packing maximizes computational intensity and amplifies power oscillations.

The evolutionary optimization process eliminated global memory access instructions from the final stressor patterns. Global memory operations introduce access latencies of approximately 400-600 cycles [38], exceeding the target oscillation period and introducing timing irregularities that disrupt the precise high-low power switching required for resonant frequency matching. The removal of off-chip memory operations ensures that power oscillations remain synchronized with the PDN's resonant characteristics, maintaining the temporal

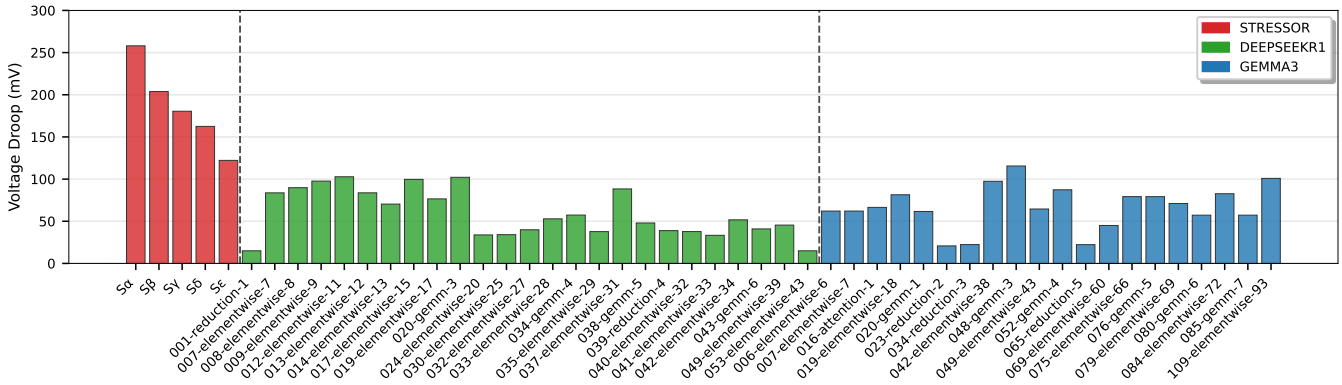
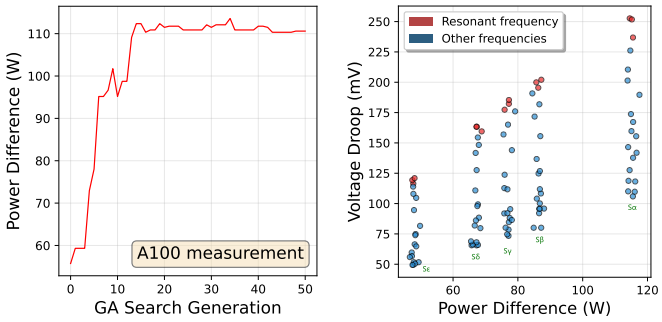


Fig. 12: Comparative voltage droop analysis across LLM kernels during the DeepSeekR1 and Gemma inference, and generated di/dt stressors, showing voltage droop levels. In red are Stressors **S**α (leftmost) through **S**ε (rightmost).



(a) (b)

Fig. 13: Power swing on GPU hardware and simulator measured with kernels in each generation of genetic algorithm (a) Power swing **measured on real hardware**, and (b) Voltage droops for stressor kernels versus respective power swings measured as the stressor moves across different operating frequencies **obtained through simulation**. Stressor **S**α (leftmost) through **S**ε (rightmost). The peak power swing on hardware (left) is approx 113W and the peak on the simulator (observing the rightmost red dot in the right figure) is also approx 115W, validating the relative accuracy of the simulator.

precision necessary for maximum voltage droop generation. However, this constraint prevents the maximum power consumption from reaching the GPU's TDP limits.

D. Stressor Validation on Real Hardware

To validate stressor behavior on real hardware, an indirect observation method is employed: thread synchronization patterns are monitored across oscillation frequencies ranging from 1 MHz to 70 MHz. This approach reveals GPU protection mechanisms triggered by voltage instabilities.

To quantify synchronization, we trace the oscillation start time for each thread and compute their average as the reference time. We then compare each thread's individual cycle time against this reference. A thread is considered "in sync" if its cycle time falls within $\pm 10\%$ of the average period.

Figure 14 demonstrates the thread synchronization patterns under varying oscillation frequencies. Near resonant conditions of 47 MHz, thread synchronization rapidly deteriorates to below 20%, indicating activation of GPU protection mechanisms that stall and release threads to mitigate voltage droop effects. In contrast, operation at non-resonant frequencies exhibits different behaviors: higher frequencies (70.5 MHz) show gradual desynchronization, while lower frequencies (7.05 MHz) demonstrate periodic resynchronization capability. Frequencies at and below 1 MHz maintain consistent thread synchronization throughout execution.

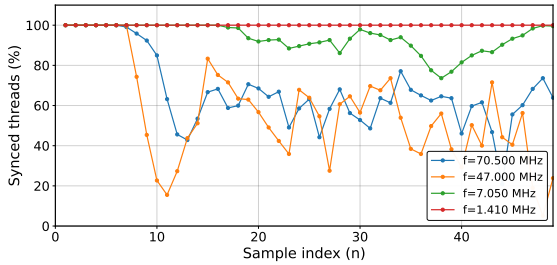


Fig. 14: Thread synchronization percentage across different oscillation frequencies. The x-axis represents samples at each oscillation period, and the y-axis shows the percentage of synchronized threads.

Critically, even with protection mechanisms active, the initial 3-5 oscillations at resonant frequency generate instantaneous voltage droops exceeding 100 mV before protective measures take effect, which has the potential of causing the voltage droop error.

VII. Reducing LLM Voltage Droops

A. Warp-Level Staggering Execution for Power Smoothing

Warp-level staggering mitigates voltage droop vulnerability by introducing controlled delays to thread or warp start times. In this section, we modify the warp scheduling in software to apply a calibrated delay between consecutive warps. This approach reduces synchronized power transitions, smoothing the aggregate power profile and reducing the risk of exciting PDN resonant frequencies.

The delay magnitude is calibrated relative to the oscillation period $T = 1/f$, with optimal delays spanning $T/4$ to $T/2$. This ensures effective temporal distribution of power transitions while avoiding excessive performance degradation. By delaying warp execution, the technique retains block-level simplicity while achieving power-smoothing benefits.

Figure 15 illustrates the impact of warp-level staggering. Without staggering, synchronized execution produces sharp power oscillations that align with resonant frequencies, leading to significant voltage droops. In contrast, warp-level staggering temporally distributes power transitions, reducing oscillatory amplitude and mitigating droop effects.

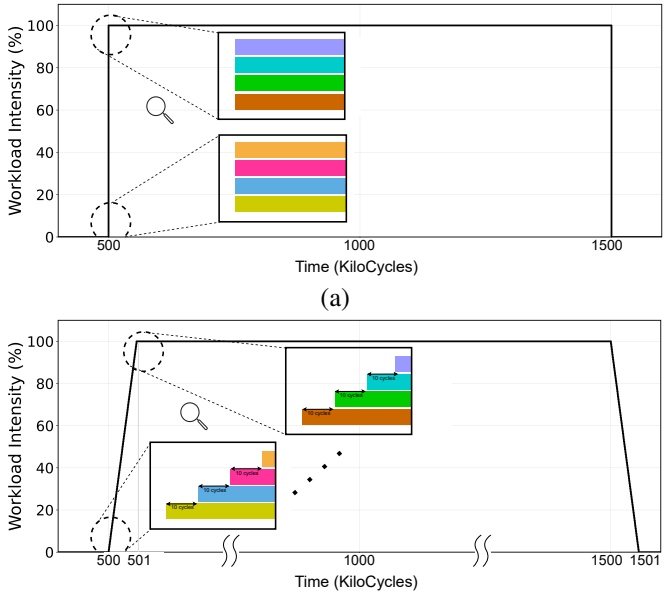


Fig. 15: Power consumption profile comparison between synchronized and staggered warp execution: (a) Synchronized execution with abrupt power changes, where all warps start within a very small time range; (b) Staggered execution showing a gradual power ramp with reduced resonance, using an N-cycle delay between warps to place them out of phase.

The runtime overhead introduced by warp-level staggering can be analytically estimated by modeling the cumulative per-warp delay. Assuming a fixed stagger interval, the total additional latency can be approximated as:

$$T_{\text{delay}} = N_{\text{warp}} \times t_{\text{stagger}}, \tag{3}$$

where N_{warp} denotes the total number of warps launched by the kernel (reported by `nsys`), and t_{stagger} is the per-warp delay in seconds. In our configuration, the stagger delay is 10 cycles at 1410 MHz GPU frequency, corresponding to $t_{\text{stagger}} \approx 7.09$ ns. For the 6000-size GEMM kernel with $N_{\text{warp}} = 8{,}836$ warps, this yields a theoretical upper bound of $T_{\text{delay}} \approx 62.6$ μs.

Empirical results in Figure 16 show that the runtime increase remains well below the analytical bound while effectively suppressing voltage droops. The measured overhead is within tens of microseconds—about 0.7% of total kernel runtime—demonstrating that warp-level staggering achieves power smoothing with negligible performance impact.

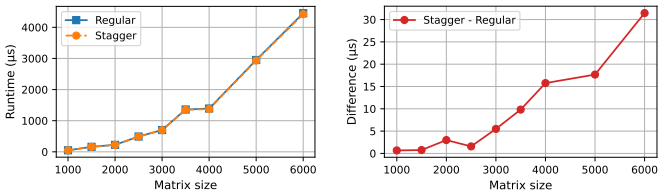


Fig. 16: Runtime analysis of warp-level staggering: (a) comparison of total kernel runtime between synchronized and staggered execution; (b) measured runtime difference across varying matrix sizes.

B. Experimental Validation of Staggered Execution

We applied the staggered execution technique to low-level **CUTLASS** [42] GEMM kernels and evaluated its effectiveness on voltage droop mitigation. The evaluation focuses on oscillations above 20 MHz, which correspond to the first droop resonant frequency, while excluding lower frequency oscillations that exhibit more static behavior.

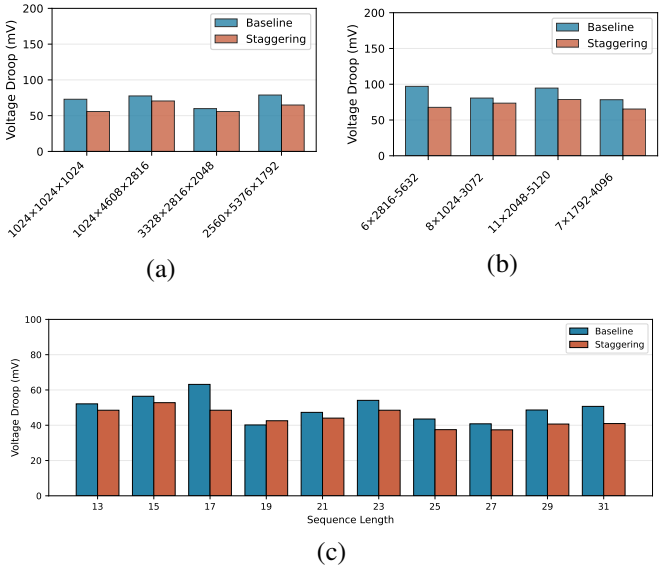


Fig. 17: Voltage droop reduction with staggered execution: (a) TF32 GEMM kernels, (b) Grouped GEMM kernels, and (c) End-to-end validation on Gemma3 (batch=4) within vLLM using a custom-linked CUTLASS build.

GEMM kernels were chosen for this evaluation because they are a dominant component in LLM workloads and exhibit significant oscillatory behavior, as shown in Section IV. Two GEMM kernel types were evaluated: **TF32 GEMM** kernels that perform standard dense matrix multiplication operations optimized for tensor cores, and **Grouped GEMM** kernels that execute multiple independent GEMM operations in parallel with potentially different matrix dimensions, enabling higher computational throughput through increased parallelism. For

example, a 6×2816-5632 Grouped GEMM means six independent GEMMs with matrix sizes ranging from 2816 to 5632. We also observe that even with the staggering technique, the effect on latency is negligible.

TF32 GEMM Results: Figure 17(a) shows consistent voltage droop reduction across varying matrix dimensions, with the maximum reduction of approximately 20%. For example, the $1024 \times 1024 \times 1024$ GEMM operation reduced droop from approximately 70 mV to 50 mV. However, the absolute reduction is limited by low power consumption during high-frequency oscillation periods and oscillation frequencies that do not fully align with PDN resonant bands.

Grouped GEMM Results: Figure 17(b) reveals more substantial droop mitigation, with reductions from approximately 100 mV to 60 mV. The larger improvement is due to higher static power consumption from parallel workloads, which amplifies both the magnitude of voltage droops and the effect of staggering. This highlights the greater benefit of staggering in high-intensity workloads.

Beyond microbenchmarks, we validated that kernel-level modifications generalize to full LLM inference in Figure 17(c). Specifically, vLLM can load a custom CUTLASS build by linking the modified library path, enabling our staggering-aware GEMM kernels in end-to-end runs. Using Gemma3 with batch size 4, we compared GEMM kernels at multiple sequence lengths and observed that, in most cases, staggering reduces voltage droop across sequences, consistent with the results Figure 17(a) and (b). While effective, this approach depends on the ability to integrate custom kernels; closed-source or tightly coupled vendor libraries may limit deployability in some environments.

Observation 5

Warp-level staggering is an effective strategy for mitigating voltage droops in LLM workloads, especially in computationally intensive scenarios. Higher baseline power in such scenarios increases baseline voltage droops, but staggering reduces the noise amplitude.

VIII. Related Work

Power delivery network modeling and automated stressor generation have been extensively developed for CPU systems. PDN fundamentals were established by Joseph et al. [21] and Powell & Vijaykumar [47], who demonstrated RLC circuit modeling of PDNs showed how alternating high-power/low-power workload sequences excite these resonances. Automated frameworks advanced from Joshi et al.'s [22] pioneering genetic algorithm approach through SYMPO [10] and MAMPO [11] introducing configurable knobs for multicore stress testing for max power, to Kim & John's [28], [29] di/dt-specific stressmark generation using SimpleScalar/Wattch/H-SPICE simulation chains for resonant effects. Gupta et al. [12] and Brooks et al. [1] provided sophisticated PDN modeling and power-performance analysis methodologies. Runtime mitigation approaches include voltage emergency elimination [13], [15], [37], [49], droop prediction [14], [50], [51], and thread scheduling techniques [46], [52], while [25] analyzing compiler optimization impacts on voltage noise.

Silent Data Corruption concerns have been extensively documented in large-scale production systems through industry studies from Meta [6], Facebook [7], Google [16], [53], Alibaba [66], and Intel [33], all demonstrating real-world SDC detection challenges and emphasizing that critical timing paths with small margins are particularly susceptible to voltage droop-induced SDC. Comprehensive surveys and analysis frameworks established the connection between voltage stability and SDC vulnerability, providing strong motivation for voltage droop analysis in reliability-critical systems.

Prior works in stressmark generation include SYMPO [10], MAMPO [11], Guser [54], and AUDIT [29]. While CPU stressor frameworks like SYMPO/MAMPO/AUDIT effectively used genetic algorithms for parameter space exploration, these cannot transfer to GPUs due to fundamental architectural differences (specialized units, CUDA programming models, different pipelines). Some prior GPU simulation works focused on performance [34], [57], rather than power analysis. Shan et al.'s Guser [54] provided the first systematic GPGPU power stressmark generation for the max power, but it does not study di/dt effects or voltage droops. GPU voltage noise modeling and mitigation [30], [31], [32], [48], [60], [63], [64], [67], [71] provides comprehensive voltage noise characterization with microarchitectural component analysis.

IX. Conclusion

This work presents a comprehensive framework for analyzing and mitigating GPU Power Delivery Network vulnerability considering LLM workloads. The research makes three key contributions to understanding and addressing oscillation-induced voltage droop phenomena.

First, systematic analysis of LLM power consumption reveals oscillatory behavior across multiple granularities, with intra-kernel oscillations approaching PDN resonant frequency ranges. This characterization demonstrates that LLM workloads can inadvertently trigger voltage instabilities through resonance effects during kernel executions. We also note that the swings caused by repetition of kernels in LLMs are at a lower frequency outside the resonant frequency range. Additionally, experimental validation demonstrates that minimal power oscillations of even 10 W at resonant frequencies generate large voltage perturbations.

Second, we present a controllable di/dt stressor generation framework that enables systematic GPU reliability assessment under LLM oscillatory conditions. The generated stressors can produce 2× larger voltage droops compared to GPU workloads while providing precise control over any frequency and voltage magnitude combination. This capability enables comprehensive PDN vulnerability characterization across the complete spectrum of potential operating scenarios.

Third, the proposed staggered warp level execution methodology provides practical mitigation for reducing LLM vulner-

ability to voltage droops by smoothing power consumption profiles and reducing resonance excitation likelihood.

Future work should extend analysis through hardware-based power measurement using direct probing techniques and comprehensive profiling of diverse LLM architectures to establish a broader understanding of oscillatory power characteristics across different model types and deployment scenarios.

X. Acknowledgement

This research was supported in part by the Semiconductor Research Corporation (SRC) Task 3281.001. Any opinions, findings, conclusions, or recommendations are those of the authors and not of the funding agencies.

References

[1] D. Brooks, P. Bose, V. Srinivasan, M. K. Gschwind, P. G. Emma, and M. G. Rosenfield, "New methodology for early-stage, microarchitecture-level power-performance analysis of microprocessors," *IBM Journal of Research and Development*, vol. 47, no. 5.6, pp. 653–670, 2003.

[2] E. Choukse, B. Warrier, S. Heath, L. Belmont, A. Zhao, H. A. Khan, B. Harry, M. Kappel, R. J. Hewett, K. Datta, Y. Pei, C. Lichtenberger, J. Siegler, D. Lukofsky, Z. Kahn, G. Sahota, A. Sullivan, C. Frederick, H. Thai, R. Naughton, D. Jurnove, J. Harp, R. Carper, N. Mahalingam, S. Varkala, A. G. Kumbhare, S. Desai, V. Ramamurthy, P. Gottumukkala, G. Bhatia, K. Wildstone, L. Olariu, I. Incorvaia, A. Wetmore, P. Ram, M. Raghuraman, M. Ayna, M. Kendrick, R. Bianchini, A. Hurst, R. Zamani, X. Li, M. Petrov, G. Oden, R. Carmichael, T. Li, A. Gupta, P. Patel, N. Dattani, L. Marwong, R. Nertney, H. Kobayashi, J. Liott, M. Enev, D. Ramakrishnan, I. Buck, and J. Alben, "Power stabilization for ai training datacenters," 2025. [Online]. Available: https://arxiv.org/abs/2508.14318

[3] N. Corporation, *NVIDIA A100 Tensor Core GPU Architecture*, 2020, architecture analysis; includes MIG, tensor cores, HBM2, NVLink. [Online]. Available: https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf

[4] DeepSeek-AI, D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, X. Zhang, X. Yu, Y. Wu, Z. F. Wu, Z. Gou, Z. Shao, Z. Li, Z. Gao, A. Liu, B. Xue, B. Wang, B. Wu, B. Feng, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, D. Dai, D. Chen, D. Ji, E. Li, F. Lin, F. Dai, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Bao, H. Xu, H. Wang, H. Ding, H. Xin, H. Gao, H. Qu, H. Li, J. Guo, J. Li, J. Wang, J. Chen, J. Yuan, J. Qiu, J. Li, J. L. Cai, J. Ni, J. Liang, J. Chen, K. Dong, K. Hu, K. Gao, K. Guan, K. Huang, K. Yu, L. Wang, L. Zhang, L. Zhao, L. Wang, L. Zhang, L. Xu, L. Xia, M. Zhang, M. Zhang, M. Tang, M. Li, M. Wang, M. Li, N. Tian, P. Huang, P. Zhang, Q. Wang, Q. Chen, Q. Du, R. Ge, R. Zhang, R. Pan, R. Wang, R. J. Chen, R. L. Jin, R. Chen, S. Lu, S. Zhou, S. Chen, S. Ye, S. Wang, S. Yu, S. Zhou, S. Pan, S. S. Li, S. Zhou, S. Wu, S. Ye, T. Yun, T. Pei, T. Sun, T. Wang, W. Zeng, W. Zhao, W. Liu, W. Liang, W. Gao, W. Yu, W. Zhang, W. L. Xiao, W. An, X. Liu, X. Wang, X. Chen, X. Nie, X. Cheng, X. Liu, X. Xie, X. Liu, X. Yang, X. Li, X. Su, X. Lin, X. Q. Li, X. Jin, X. Shen, X. Chen, X. Sun, X. Wang, X. Song, X. Zhou, X. Wang, X. Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. Zhang, Y. Xu, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Yu, Y. Zhang, Y. Shi, Y. Xiong, Y. He, Y. Piao, Y. Wang, Y. Tan, Y. Ma, Y. Liu, Y. Guo, Y. Ou, Y. Wang, Y. Gong, Y. Zou, Y. He, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Y. X. Zhu, Y. Xu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Y. Tang, Y. Zha, Y. Yan, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Ma, Z. Yan, Z. Wu, Z. Gu, Z. Zhu, Z. Liu, Z. Li, Z. Xie, Z. Song, Z. Pan, Z. Huang, Z. Xu, Z. Zhang, and Z. Zhang, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," 2025. [Online]. Available: https://arxiv.org/abs/2501.12948

[5] Y. Deng, X. Shi, Z. Jiang, X. Zhang, L. Zhang, Z. Zhang, B. Li, Z. Song, H. Zhu, G. Liu, F. Li, S. Wang, H. Lin, J. Ye, and M. Yu, "Minder: Faulty machine detection for large-scale distributed model training," in *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*. Philadelphia, PA: USENIX Association, Apr. 2025, pp. 505–521. [Online]. Available: https://www.usenix.org/conference/nsdi25/presentation/deng

[6] H. D. Dixit, L. Boyle, G. Vunnam, S. Pendharkar, M. Beadon, and S. Sankar, "Detecting silent data corruptions in the wild," 2022. [Online]. Available: https://arxiv.org/abs/2203.08989

[7] H. D. Dixit, S. Pendharkar, M. Beadon, C. Mason, T. Chakravarthy, B. Muthiah, and S. Sankar, "Silent data corruptions at scale," 2021. [Online]. Available: https://arxiv.org/abs/2102.11245

[8] W. El-Essawy and D. Albonesi, "Mitigating inductive noise in smt processors," in *Proceedings of the 2004 International Symposium on Low Power Electronics and Design (IEEE Cat. No.04TH8758)*, 2004, pp. 332–337.

[9] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, jul 2012.

[10] K. Ganesan, J. Jo, W. L. Bircher, D. Kaseridis, Z. Yu, and L. K. John, "System-level max power (sympo) - a systematic approach for escalating system-level power consumption using synthetic benchmarks," in *2010 19th International Conference on Parallel Architectures and Compilation Techniques (PACT)*, 2010, pp. 19–28.

[11] K. Ganesan and L. K. John, "Maximum multicore power (mampo) — an automatic multithreaded synthetic power virus generation framework for multicore systems," in *SC '11: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, 2011, pp. 1–12.

[12] M. S. Gupta, J. L. Oatley, R. Joseph, G.-Y. Wei, and D. M. Brooks, "Understanding voltage variations in chip multiprocessors using a distributed power-delivery network," in *2007 Design, Automation & Test in Europe Conference & Exhibition*, 2007, pp. 1–6.

[13] M. S. Gupta, K. K. Rangan, M. D. Smith, G.-Y. Wei, and D. Brooks, "Decor: A delayed commit and rollback mechanism for handling inductive noise in processors," in *2008 IEEE 14th International Symposium on High Performance Computer Architecture*, 2008, pp. 381–392.

[14] M. S. Gupta, V. J. Reddi, G. Holloway, G.-Y. Wei, and D. M. Brooks, "An event-guided approach to reducing voltage noise in processors," in *2009 Design, Automation & Test in Europe Conference & Exhibition*, 2009, pp. 160–165.

[15] M. S. Gupta, K. K. Rangan, M. D. Smith, G.-Y. Wei, and D. Brooks, "Towards a software approach to mitigate voltage emergencies," in *Proceedings of the 2007 international symposium on Low power electronics and design (ISLPED '07)*, 2007, pp. 123–128.

[16] P. H. Hochschild, P. Turner, J. C. Mogul, R. Govindaraju, P. Ranganathan, D. E. Culler, and A. Vahdat, "Cores that don't count," New York, NY, USA, p. 9–16, 2021. [Online]. Available: https://doi.org/10.1145/3458336.3465297

[17] Q. Hu, Z. Ye, Z. Wang, G. Wang, M. Zhang, Q. Chen, P. Sun, D. Lin, X. Wang, Y. Luo, Y. Wen, and T. Zhang, "Characterization of large language model development in the datacenter," in *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI'24. USA: USENIX Association, 2024.

[18] M. Jeon, S. Venkataraman, A. Phanishayee, J. Qian, W. Xiao, and F. Yang, "Analysis of Large-Scale Multi-Tenant GPU clusters for DNN training workloads," in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*. Renton, WA: USENIX Association, Jul. 2019, pp. 947–960. [Online]. Available: https://www.usenix.org/conference/atc19/presentation/jeon

[19] P. Jiang, C. Sonne, W. Li, F. You, and S. You, "Preventing the immense increase in the life-cycle energy and carbon footprints of LLM-powered intelligent chatbots," *PII*, vol. 40, pp. 202–210, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2095809924002315

[20] Z. Jiang, H. Lin, Y. Zhong, Q. Huang, Y. Chen, Z. Zhang, Y. Peng, X. Li, C. Xie, S. Nong, Y. Jia, S. He, H. Chen, Z. Bai, Q. Hou, S. Yan, D. Zhou, Y. Sheng, Z. Jiang, H. Xu, H. Wei, Z. Zhang, P. Nie, L. Zou, S. Zhao, L. Xiang, Z. Liu, Z. Li, X. Jia, J. Ye, X. Jin, and X. Liu, "Megascale: scaling large language model training to more than 10,000 gpus," in *Proceedings of the 21st USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI'24. USA: USENIX Association, 2024.

[21] R. Joseph, D. Brooks, and M. Martonosi, "Control techniques to eliminate voltage emergencies in high performance processors," in *The Ninth International Symposium on High-Performance Computer Architecture, 2003. HPCA-9 2003. Proceedings.*, 2003, pp. 79–90.

[22] A. M. Joshi, L. Eeckhout, L. K. John, and C. Isen, "Automated microprocessor stressmark generation," in *2008 IEEE 14th International*

Symposium on High Performance Computer Architecture, 2008, pp. 229–239.

[23] D. A. Kamakshi, M. Fojtik, B. Khailany, S. Kudva, Y. Zhou, and B. H. Calhoun, "Modeling and analysis of power supply noise tolerance with fine-grained gals adaptive clocks," in *2016 22nd IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, 2016, pp. 75–82.

[24] 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*, ser. MICRO '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 738–753. [Online]. Available: https://doi.org/10.1145/3466752.3480063

[25] S. Kanev, T. M. Jones, G. Wei, D. Brooks, and V. J. Reddi, "Measuring code optimization impact on voltage noise," in *Proceedings of the 9th Silicon Errors in Logic – System Effects Workshop (SELSE 9)*, Mar. 2013. [Online]. Available: http://www.eecs.harvard.edu/~skanev/papers/selse13vopt.pdf

[26] M. Khairy, Z. Shen, T. M. Aamodt, and T. G. Rogers, "Accel-sim: An extensible simulation framework for validated gpu modeling," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, 2020, pp. 473–486.

[27] S. Kim, C. Hooper, T. Wattanawong, M. Kang, R. Yan, H. Genc, G. Dinh, Q. Huang, K. Keutzer, M. W. Mahoney, Y. S. Shao, and A. Gholami, "Full stack optimization of transformer inference: a survey," 2023. [Online]. Available: https://arxiv.org/abs/2302.14017

[28] Y. Kim and L. K. John, "Automated di/dt stressmark generation for microprocessor power delivery networks," in *IEEE/ACM International Symposium on Low Power Electronics and Design*, 2011, pp. 253–258.

[29] Y. Kim, L. K. John, S. Pant, S. Manne, M. Schulte, W. L. Bircher, and M. S. S. Govindan, "Audit: Stress testing the automatic way," in *2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, 2012, pp. 212–223.

[30] J. Leng, A. Buyuktosunoglu, R. Bertran, P. Bose, and V. J. Reddi, "Safe limits on voltage reduction efficiency in gpus: A direct measurement approach," in *2015 48th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2015, pp. 294–307.

[31] J. Leng, Y. Zu, and V. J. Reddi, "Gpu voltage noise: Characterization and hierarchical smoothing of spatial and temporal voltage noise interference in gpu architectures," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 161–173.

[32] J. Leng, Y. Zu, M. Rhu, M. S. Gupta, and V. J. Reddi, "Gpuvolt: Modeling and characterizing voltage noise in gpu architectures," in *2014 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 2014, pp. 141–146.

[33] D. P. Lerner, B. Inkley, S. H. Sahasrabudhe, E. Hansen, L. D. R. Munoz, and A. v. de Ven, "Optimization of tests for managing silicon defects in data centers," in *2022 IEEE International Test Conference (ITC)*, 2022, pp. 578–582.

[34] C. Liu, Y. Sun, and T. E. Carlson, "Photon: A fine-grained sampled simulation methodology for gpu workloads," in *MICRO '23*. New York, NY, USA: Association for Computing Machinery, 2023, p. 1227–1241. [Online]. Available: https://doi.org/10.1145/3613424.3623773

[35] K. Liu, Z. Jiang, J. Zhang, H. Wei, X. Zhong, L. Tan, T. Pan, and T. Huang, "Hostping: Diagnosing intra-host network bottlenecks in RDMA servers," in *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*. Boston, MA: USENIX Association, Apr. 2023, pp. 15–29. [Online]. Available: https://www.usenix.org/conference/nsdi23/presentation/liu-kefei

[36] K. Maeng, S. Bharuka, I. Gao, M. Jeffrey, V. Saraph, B.-Y. Su, C. Trippel, J. Yang, M. Rabbat, B. Lucia, and C.-J. Wu, "Understanding and improving failure tolerant training for deep learning recommendation with partial recovery," in *Proceedings of Machine Learning and Systems*, A. Smola, A. Dimakis, and I. Stoica, Eds., vol. 3, 2021, pp. 637–651. [Online]. Available: https://proceedings.mlsys.org/paper_files/paper/2021/file/f0f9e98bc2e2f0abc3e315eaa0d808fc-Paper.pdf

[37] T. N. Miller, R. Thomas, X. Pan, and R. Teodorescu, "Vrsync: Characterizing and eliminating synchronization-induced voltage emergencies in many-core processors," in *2012 39th Annual International Symposium on Computer Architecture (ISCA)*, 2012, pp. 249–260.

[38] NVIDIA Corporation, *CUDA C Best Practices Guide*, 2016, version 8.0. [Online]. Available: https://docs.nvidia.com/cuda/archive/8.0/cuda-c-best-practices-guide/index.html

[39] NVIDIA Corporation, "Nvidia volta: The world's most advanced data center gpu," https://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf, 2017.

[40] NVIDIA Corporation, "Nvidia ampere architecture: Whitepaper," https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf, 2020.

[41] NVIDIA Corporation, "Nvidia h100 tensor core gpu: Hopper architecture whitepaper," https://www.advancedclustering.com/wp-content/uploads/2022/03/gtc22-whitepaper-hopper.pdf, 2022.

[42] NVIDIA Corporation, "CUTLASS: Cuda templates for linear algebra subroutines," https://github.com/NVIDIA/cutlass, 2024, version 3.2.0, accessed August 2025.

[43] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph, "Gpt-4 technical report," 2024. [Online]. Available: https://arxiv.org/abs/2303.08774

[44] G. Ostrouchov, D. Maxwell, R. A. Ashraf, C. Engelmann, M. Shankar, and J. H. Rogers, "Gpu lifetimes on titan supercomputer: Survival analysis and reliability," in *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020, pp. 1–14.

[45] N. Pinckney, K. Sewell, R. G. Dreslinski, D. Fick, T. Mudge, D. Sylvester, and D. Blaauw, "Assessing the performance limits of parallelized near-threshold computing," in *Proceedings of the 49th Annual Design Automation Conference*, ser. DAC '12. New York, NY, USA: Association for Computing Machinery, 2012, p. 1147–1152. [Online]. Available: https://doi.org/10.1145/2228360.2228571

[46] M. Powell and T. Vijaykumar, "Pipeline muffling and a priori current ramping: architectural techniques to reduce high-frequency inductive

noise," in *Proceedings of the 2003 International Symposium on Low Power Electronics and Design, 2003. ISLPED '03.*, 2003, pp. 223–228.

[47] M. Powell and T. Vijaykumar, "Exploiting resonant behavior to reduce inductive noise," in *Proceedings. 31st Annual International Symposium on Computer Architecture, 2004.*, 2004, pp. 288–299.

[48] T. Rahal-Arabi, P. Van der Arend, A. Jain, M. Saidi, R. Oreifej, S. Sundaram, S. Manne, I. Paul, R. Seahra, F. Helms, E. Choukse, N. Mahalingam, B. Warrier, and R. Bianchini, "Optimizing gpu data center power," in *2024 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2024, pp. 358–362.

[49] V. J. Reddi, S. Campanoni, M. S. Gupta, M. D. Smith, G.-Y. Wei, D. Brooks, and K. Hazelwood, "Eliminating voltage emergencies via software-guided code transformations," *ACM Trans. Archit. Code Optim.*, vol. 7, no. 2, Oct. 2010. [Online]. Available: https://doi.org/10.1145/1839667.1839674

[50] V. J. Reddi, M. Gupta, G. Holloway, M. D. Smith, G.-Y. Wei, and D. Brooks, "Predicting voltage droops using recurring program and microarchitectural event activity," *IEEE Micro*, vol. 30, no. 1, pp. 110–110, 2010.

[51] V. J. Reddi, M. S. Gupta, G. Holloway, G.-Y. Wei, M. D. Smith, and D. Brooks, "Voltage emergency prediction: Using signatures to reduce operating margins," in *2009 IEEE 15th International Symposium on High Performance Computer Architecture*, 2009, pp. 18–29.

[52] V. J. Reddi, S. Kanev, W. Kim, S. Campanoni, M. D. Smith, G.-Y. Wei, and D. Brooks, "Voltage smoothing: Characterizing and mitigating voltage noise in production processors via software-guided thread scheduling," in *2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture*, 2010, pp. 77–88.

[53] K. Serebryany, M. Lifantsev, K. Shtoyk, D. Kwan, and P. Hochschild, "Silifuzz: Fuzzing cpus by proxy," 2021. [Online]. Available: https://arxiv.org/abs/2110.11519

[54] Y. Shan, Y. Yang, X. Qian, and Z. Yu, "Guser: A gpgpu power stressmark generator," in *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 2024, pp. 1111–1124.

[55] A. Stillmaker and B. Baas, "Scaling equations for the accurate prediction of CMOS device performance from 180nm to 7nm," *Integration*, vol. 58, pp. 74–81, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926017300755

[56] Y. Sun, N. B. Agostini, S. Dong, and D. Kaeli, "Summarizing cpu and gpu design trends with product data," 2020. [Online]. Available: https://arxiv.org/abs/1911.11313

[57] Y. Sun, T. Baruah, S. A. Mojumder, S. Dong, X. Gong, S. Treadway, Y. Bao, S. Hance, C. McCardwell, V. Zhao, H. Barclay, A. K. Ziabari, Z. Chen, R. Ubal, J. L. Abellán, J. Kim, A. Joshi, and D. Kaeli, "Mgpusim: Enabling multi-gpu performance modeling and optimization," in *2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA)*, 2019, pp. 197–209.

[58] A. Taherin, T. Patel, G. Georgakoudis, I. Laguna, and D. Tiwari, "Examining failures and repairs on supercomputers with multi-gpu compute nodes," in *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2021, pp. 305–313.

[59] J. Tan, K. Chen, W. Wang, K. Yan, and X. Wei, "Mcm-gpu voltage noise characterization and architecture-level mitigation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 12, pp. 5084–5097, 2023.

[60] J. Tan, S. L. Song, K. Yan, X. Fu, A. Marquez, and D. Kerbyson, "Combating the reliability challenge of gpu register file at low supply voltage," in *2016 International Conference on Parallel Architecture and Compilation Techniques (PACT)*, 2016, pp. 3–15.

[61] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, P. Tafti, L. Hussenot, P. G. Sessa, A. Chowdhery, A. Roberts, A. Barua, A. Botev, A. Castro-Ros, A. Slone, A. Héliou, A. Tacchetti, A. Bulanova, A. Paterson, B. Tsai, B. Shahriari, C. L. Lan, C. A. Choquette-Choo, C. Crepy, D. Cer, D. Ippolito, D. Reid, E. Buchatskaya, E. Ni, E. Noland, G. Yan, G. Tucker, G.-C. Muraru, G. Rozhdestvenskiy, H. Michalewski, I. Tenney, I. Grishchenko, J. Austin, J. Keeling, J. Labanowski, J.-B. Lespiau, J. Stanway, J. Brennan, J. Chen, J. Ferret, J. Chiu, J. Mao-Jones, K. Lee, K. Yu, K. Millican, L. L. Sjoesund, L. Lee, L. Dixon, M. Reid, M. Mikuła, M. Wirth, M. Sharman, N. Chinaev, N. Thain, O. Bachem, O. Chang, O. Wahltinez, P. Bailey, P. Michel, P. Yotov, R. Chaabouni, R. Comanescu, R. Jana, R. Anil, R. McIlroy, R. Liu, R. Mullins, S. L. Smith, S. Borgeaud, S. Girgin, S. Douglas, S. Pandya, S. Shakeri, S. De, T. Klimenko, T. Hennigan, V. Feinberg, W. Stokowiec, Y. hui Chen, Z. Ahmed, Z. Gong, T. Warkentin, L. Peran, M. Giang, C. Farabet, O. Vinyals, J. Dean, K. Kavukcuoglu, D. Hassabis, Z. Ghahramani, D. Eck, J. Barral, F. Pereira, E. Collins, A. Joulin, N. Fiedel, E. Senter, A. Andreev, and K. Kenealy, "Gemma: Open models based on gemini research and technology," 2024. [Online]. Available: https://arxiv.org/abs/2403.08295

[62] I. Team, "Intel® pentium® 4 processor in the 423-pin package / intel® 850 chipset platform design guide," Intel Corporation, Tech. Rep. 298245-005, 2002, design Guide.

[63] R. Thomas, K. Barber, N. Sedaghati, L. Zhou, and R. Teodorescu, "Core tunneling: Variation-aware voltage noise mitigation in gpus," in *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2016, pp. 151–162.

[64] R. Thomas, N. Sedaghati, and R. Teodorescu, "Emergpu: Understanding and mitigating resonance-induced voltage noise in gpu architectures," in *2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2016, pp. 79–89.

[65] B. Wan, M. Han, Y. Sheng, Y. Peng, H. Lin, M. Zhang, Z. Lai, M. Yu, J. Zhang, Z. Song, X. Liu, and C. Wu, "Bytecheckpoint: A unified checkpointing system for large foundation model development," 2025. [Online]. Available: https://arxiv.org/abs/2407.20143

[66] S. Wang, G. Zhang, J. Wei, Y. Wang, J. Wu, and Q. Luo, "Understanding silent data corruptions in a large production cpu population," in *Proceedings of the 29th Symposium on Operating Systems Principles*, ser. SOSP '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 216–230. [Online]. Available: https://doi.org/10.1145/3600006.3613149

[67] K. Wong, T. Rahal-Arabi, M. Ma, and G. Taylor, "Enhancing microprocessor immunity to power supply noise with clock-data compensation," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 4, pp. 749–758, 2006.

[68] Q. Xie, X. Lin, Y. Wang, S. Chen, M. J. Dousti, and M. Pedram, "Performance comparisons between 7-nm finfet and conventional bulk cmos standard cell libraries," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 8, pp. 761–765, 2015.

[69] Z. Yao, P. Hu, C. Miao, X. Jia, Z. Liang, Y. Xu, C. He, H. Lu, M. Chen, X. Li, Z. He, Y. Wang, X. Zou, and J. Jiang, "Holmes: Localizing irregularities in LLM training with mega-scale GPU clusters," in *22nd USENIX Symposium on Networked Systems Design and Implementation (NSDI 25)*. Philadelphia, PA: USENIX Association, Apr. 2025, pp. 523–540. [Online]. Available: https://www.usenix.org/conference/nsdi25/presentation/yao

[70] R. Zhang, W. Xiao, H. Zhang, Y. Liu, H. Lin, and M. Yang, "An empirical study on program failures of deep learning jobs," in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, 2020, pp. 1159–1170.

[71] A. Zou, J. Leng, X. He, Y. Zu, C. D. Gill, V. Janapa Reddi, and X. Zhang, "Voltage-stacked gpus: A control theory driven cross-layer solution for practical voltage stacking in gpus," in *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2018, pp. 390–402.