Friedrich-Alexander-Universität
Technische Fakultät

NHR FAU

Power Consumption Properties of Modern GPUs

Bachelor Thesis

at Friedrich-Alexander-Universität Erlangen-Nürnberg
at the Department of Computer Science
Professorship of High Performance Computing

in collaboration with the Research Division
(Erlangen National High Performance Computing Center)

Principal Supervisor:	Prof. Dr. Gerhard Wellein
Associate Supervisor:	MSc. Dominik Ernst
Presented by:	Jonah Holtmann 90480 Nürnberg jonah.holtmann@fau.de B.Sc. Computational Engineering 22828933
Submission:	5th December 2024

Abstract

In this thesis the power consumption properties of Floating-Point Operation (Flop) and data transfers to main memory are evaluated on modern Graphic Processing Units (GPUs). The applicability of the existing archline model of energy is verified and tested. For this purpose a set of micro benchmarks is proposed both for evaluation as well as verification of the model. To measure the device's energy consumption a driver based approach is chosen. The resulting data is used to fit the machine parameters by using linear regression. Overall, significant reductions in energy usage per operation and per memory transfer can be shown compared to the original work. On the newest GPUs the high transistor counts and the increased level of integration allow for high Flop energy efficiencies. Overall, the GPUs become compute bound in energy before becoming compute bound in time.
It is shown that the NVIDIA GPUs are not limited by their Thermal Design Power (TDP) while the MI210 is severely limited in FP64 performance by its TDP.

Contents

Figures

Tables

Abbreviations

R^2	Coefficient of Determination
Flop	Floating-Point Operation
CPU	Central Processing Unit
DFG	German Research Foundation
DRAM	Dynamic Random Access Memory
DVFS	Dynamic Voltage Frequency Scaling
FAU	Friedrich-Alexander-Universität Erlangen-Nürnberg
FP32	Single Precision Floating Point
FP64	Double Precision Floating Point
GPU	Graphic Processing Unit
HBM	High Bandwidth Memory
HPC	High-Performance Computing
Mop	Memory Operation
NHR@FAU	Erlangen National High-Performance Computing Center
NVML	NVIDIA Management Library
OAM	OCP Accelerator Module
PCIe	Peripheral Component Interconnect Express
ROCm	ROCm System Management
SXM	Server PCI Express Module
TDP	Thermal Design Power

Symbols

B_ϵ	Balance Point in Energy
$\hat{B}_\epsilon(I)$	Effective Energy Balance Point
b_{peak}	Peak Memory Bandwidth
B_τ	Balance Point in Time
E	Total Energy
$\epsilon_{0,\text{flop}}$	Constant Energy per Flop
E_{active}	Total Energy of Operations
$\hat{\epsilon}_{\text{flop}}$	Actual Energy per Flop
E_{base}	Total Baseline Energy
E_{flop}	Total Energy of Flops
ϵ_{flop}	Energy per Flop
E_{mem}	Total Energy of Bytes
ϵ_{mem}	Energy per Byte
ϵ_{mop}	Energy per Mop
I	Intensity
η_{flop}	Flop Energy Efficiency
P	Total Power
π_0	Baseline Power Consumption
π_{flop}	Power for Flop
π_{mem}	Power for Data Transfer
p_{peak}	Peak Floating Point Performance
$p_{\text{peak},32}$	Peak Single Precision Floating Point Performance (FP32)
$p_{\text{peak},64}$	Peak Double Precision Floating Point Performance (FP64)
Q	Number of Transferred Memory Bytes
T	Total Time
τ_{flop}	Time per Flop
W	Number of Floating Point Operations (Flops)

1 Introduction

With the rise in climate crisis awareness comes the need to conserve and efficiently make use of electric energy. Data centers' power consumption is estimated to make up around 3% of the global power consumption [22]. With new developments in Artificial Intelligence and its inherent need for computing power, this will probably increase even further. Since these applications benefit greatly from GPU acceleration, GPUs have become widely adopted. This is beneficial as GPUs provide larger amounts of computing performance at lower power consumption than Central Processing Units (CPUs). However, with modern GPUs not only the performance has increased but also the total power consumption per card. Some bleeding edge cards are allowed to draw up to 700W and higher [8][15]. For data center operators and programmers this leads to an increasing demand in efficient programs. This includes hardware efficiency, i.e., the use of the given hardware to its designed limits, as well as establishing and defining such limits for energy efficiency.

Carbon taxes and efficiency regulations add to the necessity for data centers to become more energy efficient. Cooling infrastructure is usually the largest single contribution to power consumption. Therefore, data centers that choose energy efficient architectures may be able to decrease the size of their cooling infrastructure.

The goal of this paper is to assess modern GPUs and to provide insight into their power consumption properties, i.e., energy per floating point arithmetic operation and per memory operation. While new generations of GPUs have higher peak performance numbers than their older counterparts, they usually have increased powerlimits as well. Insight into the power consumption properties would allow an easier comparison between hardware generations.
Furthermore, the applicability of the "Roofline Model of Energy" by [5] on modern GPUs will be evaluated. For this purpose the performance counters and the power measurement interface present on todays accelerators are used. Most often hardware vendors such as NVIDIA and AMD do not publish the energy efficiency of their GPUs.

The overarching purpose of this paper is to provide programmers and data center operators with the necessary tools to evaluate and compare modern GPUs in order to model and optimize the power consumption of a given code.

2 Background

The following chapter provides an overview of the relevant background. This thesis is closely related to the “Roofline Model of Energy” proposed by [5], as it focuses on evaluating the proposed model on modern GPUs.

Roofline Model

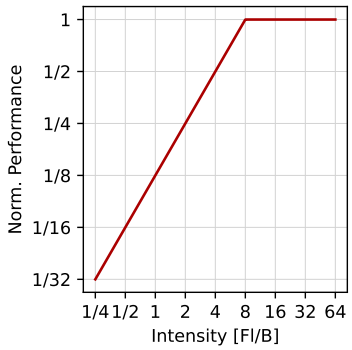


Figure 1 Example Roofline Plot

The roofline model [24] is a well established model to evaluate the time efficiency of any given program. The model uses the intensity I, peak bandwidth b_{peak} and peak performance p_{peak} to determine whether a program is either compute or memory bound in time.

$$\text{Performance} = \min(p_{\text{peak}}, I \cdot b_{\text{peak}}) \tag{1}$$

It shows the intensity on the the x-axis and the achievable peak performance on the y-axis. It provides an easy to understand visual guide to identify a potential bottleneck and improve the codes’ performance. Over the years, the roofline model has been extended to better reflect the constraints imposed by the microarchitecture [4]. Multiple papers on extensions of the roofline model towards energy [5] and on explorations towards energy efficiency [18] have been published.

Archline Model

The archline model [5] has been proposed to provide a model for energy efficiency. Similar to the roofline model [24] the intensity I is used to determine whether a code is compute bound or memory bound in energy. On the y-axis the achieved arithmetic performance per Joule is plotted. Contrary to the roofline model, the archline model does not have an inflection point as the operations’ energy costs cannot be overlapped [5, p. 664]. The upper limit of performance per watt is given by the inverse of the energy

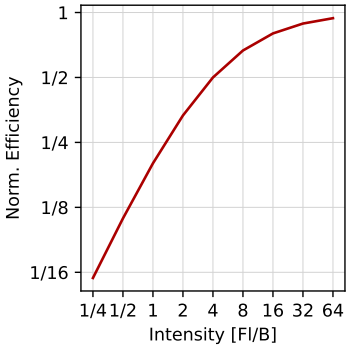


Figure 2 Example Archline Plot

per floating point operation ϵ_{flop}. The archline is missing the sharp inflection point of the roofline, as the energy usages cannot be hidden by overlapping operations. Evaluating the applicability of [5] on modern hardware is part of this thesis.

Existing Measurement Methods

Currently multiple different physical power measurement suites exist, notably PowerMon [2] and PowerPack [12]. These suites offer accurate and precise power measurements by measuring the exact voltage and the current supplied to the hardware. With increasing levels of integration and new socket designs these tools require expensive and elaborate infrastructure for use. Most modern hardware offer access to the on-board power measurement infrastructure. NVIDIA does so by means of NVIDIA Management Library (NVML) [9] and AMD by means of ROCm System Management (RSMI) [16]. Other research suggests that these measurements have gotten more accurate and can be used to measure the actual power consumption accurately [17][3].

This thesis focuses on using the hardware's power and performance measuring capabilities. The proposed measuring program is designed to be portable between different hardware vendors and easily adaptable to new benchmarks.

3 Methodology

This chapter covers the basic theory of the model behind the fitting of the power consumption properties and the proposed measurement software. The algorithm behind the linear regression fit is explained in section 3.1. The underlying formulas and graphs proposed by [5] are introduced in section 3.2. In section 3.3 a set of benchmarks is introduced to evaluate the hardware. The last section 3.4 outlines the hardware that will be evaluated.

3.1 Model

Both NVIDIA and AMD report the power draw of their GPUs as total board power value. This singular value is not attributable to any single hardware mechanism like instruction execution or data movement. Ever since the proposition of the roofline model for time [24] programmers use the time balance B_τ to evaluate the performance of a code segment. This metric focusses on the amount of Flops performed per Memory Operation (Mop). To derive a model of energy, a number of variables have to be introduced. Similar to [5], the total expended energy E is split into two partitions; (a) the active energy E_{active} that is expended for completing some work and (b) the baseline energy E_{base} expended for powering up the device:

$$E = E_{\text{active}} + E_{\text{base}} \tag{2}$$

This, of course, is a very high abstraction level and consequently needs to be refined further to evaluate the actual power consumption properties. In any case, the model does not distinguish between the actual energy expended for a single operation and the overheads that are incurred by the operation. This means that all constant overheads, such as the impact of memory and GPU clock frequencies, will be attributed to the baseline energy E_{base}. Therefore, E_{base} defines the energy needed to run the GPU in its highest performance state, i.e., the GPU clock frequency is at its “boost” clock. Power reduction measures such as Dynamic Voltage Frequency Scaling (DVFS) are not taken into account by this model. The stated parameters can be fixed before the beginning of computation and are therefore assumed to be constant during the benchmark execution. The model takes only the energy consumption of the GPU into account, disregarding the energy consumption of the rest of the system. E_{active} captures the energy not only of the completed work but also of all overheads that are incurred by completing this work. This includes among others the cost of accessing registers, look-up tables and levels of cache access. This allows the expansion of E_{active} to the sum of the total energy expended on arithmetic operations E_{flop} and transferred bytes E_{mem}. These can be seen as the sum of arithmetic operations W and transferred bytes

Q multiplied by the respective energy required per arithmetic operation ϵ_{flop} and per transferred byte ϵ_{mem}:

$$E_{\text{active}} = E_{\text{flop}} + E_{\text{mem}} = W \cdot \epsilon_{\text{flop}} + Q \cdot \epsilon_{\text{mem}} \tag{3}$$

This separation of parameters is necessary for the linear regression model otherwise no useful insight into the power consumption properties can be gained.
First, ϵ_{flop} denotes at least the energy needed to complete a single floating point instruction. This is only an estimate of the true cost of an arithmetic instruction as the model does not distinguish between instructions and the overhead that is incurred for each instruction. In contrast to [5], the presented model does not include a parameter for additional energy cost due to double precision instructions. This additional parameter allows to fit both Single Precision Floating Point (FP32) and Double Precision Floating Point (FP64) instructions as ϵ_{flop} at the same time. However, each data point is assumed to be either FP32 or FP64, i.e., the data set is comprised of either FP32 or FP64 instructions. Therefore, this additional parameter is discarded for the proposed model.
Second, ϵ_{mem} symbolizes the energy needed to transfer one byte of data from Dynamic Random Access Memory (DRAM) to the registers. A single word is either 4 Bytes for FP32 or 8 Bytes for FP64. This is supplemented by ϵ_{mop} which includes the cost to transfer the respective amount of Bytes between the thread and the GPU's main memory. Similar to ϵ_{flop}, overheads incurred for the completion of these data transfers are included.
As stated, the baseline energy E_{base} is assumed to be fixed during the kernel benchmark. Therefore, it can be expressed as a function of the total execution time T and the baseline power π_0:

$$E_{\text{base}} = T \cdot \pi_0 \tag{4}$$

Using Eq. 3 and Eq. 4 the linear regression model can be instantiated as:

$$E = W \cdot \epsilon_{\text{flop}} + Q \cdot \epsilon_{\text{mem}} + T \cdot \pi_0 \tag{5}$$

The linear regression will be used to estimate the parameters ϵ_{flop}, ϵ_{mem} and π_0 for each GPU. The values for E and T are the measured energy consumed during the execution of the benchmark and the measured total execution time of the total amount of work. The values for W and Q are the instructions and transferred bytes during the execution. Therefore, the model shoudl be able to predict the upper bound of power consumption of a given GPU kernel.

3.1.1 Linear Regression

To evaluate the measured data the linear regression routine from scikit_learn's linear model library [21] is used. It uses the ordinary least squares algorithm to fit the coefficients of the model. The routine expects a vector of dependant variables and a matrix of independant variables. Further, it either centers the data set by fitting an intercept or expects the data set to be centered. In order to score the accuracy of the model, the Coefficient of Determination (R^2) is used. This coefficient gives insight into how well the model's coefficients explain the dependent variable. It ranges from 0 to 1.0 and indicates the percentage of data that is explained by the model. Therefore, the closer this value is to 1.0, the more accurate the parameters can explain variances in E.

3.1.2 Important Considerations

For the proposed model the GPU is assumed to maintain its clock frequencies during the full computation, i.e., the hardware is not throttled. Since hardware vendors have implemented advanced features like DVFS, it is necessary to sanitize the dataset before estimating the parameters. The peak operating frequencies are chosen and only data points taken at these clock frequencies are admitted to the model. Otherwise the parameters can not be accurately approximated as they are dependant on the GPU's clock frequency. Furthermore, the model does not take power limits into account. This means that the parameters might not reflect the real world performance of a GPU as soon as it hits the powerlimit. Additionally, the model assumes that data transfers and floating point operations are performed individually. However, in practice a number of floating point operations are performed in combination with a number of data transfers. Since linear regression expects the independant variables to be fully independant, a normalization has to take place. For this purpose the codependant variable is used to normalize Eq. 5. In the case of Flops depending on Mops, the model is normalized to the amount of Flops [5, p. 666]:

$$\frac{E}{W} = \epsilon_{\text{flop}} + \frac{Q}{W} \cdot \epsilon_{\text{mem}} + \frac{T}{W} \cdot \pi_0 \tag{6}$$

The left hand side of Eq. 6, E/W is taken as a direct estimation of the total energy spent during the computation of a single floating point operation. On the right hand side, the data set is centered to the energy per Flop ϵ_{flop}. Q/W is the inverse of I and denotes the overhead induced from ϵ_{mem}. Lastly, T/W is the actual time needed to complete a single Flop, i.e., the inverse of the achieved performance, and shows the overhead of π_0. Similarly, the normalization can be made for Q:

$$\frac{E}{Q} = \frac{W}{Q} \cdot \epsilon_{\text{flop}} + \epsilon_{\text{mem}} + \frac{T}{Q} \cdot \pi_0 \tag{7}$$

In this case the overheads would be calculated based on the achieved bandwidth and intensity of the code. The data set is then centered to the energy per transferred Byte ϵ_{mem}.

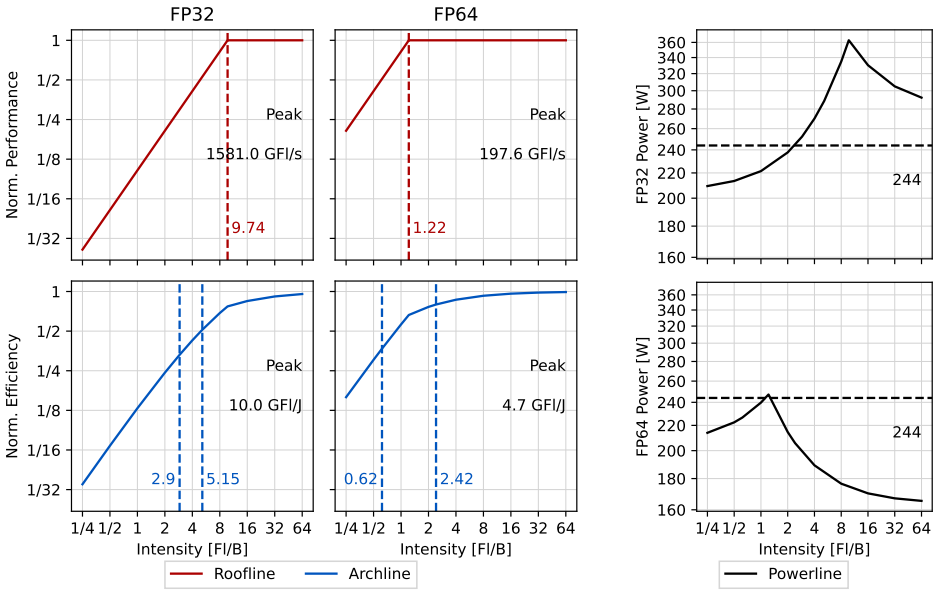
3.2 Evaluation

[5] introduces the archline model of energy as an analogue to the roofline model of performance [24]. Eq. 9 shows the relationship between the intensity I and the performance in energy. The effective energy per Flop is defined as $\hat{\epsilon}_{\text{flop}} = \epsilon_{\text{flop}} + \tau_{\text{flop}} \cdot \pi_0$. Similarly, the effective energy balance point $\hat{B}_\epsilon(I)$ is derived from a Flop's energy efficiency $\eta_{\text{flop}} = \epsilon_{\text{flop}}/\hat{\epsilon}_{\text{flop}}$ as well as both the machine's energy balance point $B_\epsilon = \epsilon_{\text{mem}}/\epsilon_{\text{flop}}$ and time balance point $B_\tau = I$.

$$\hat{B}_\epsilon(I) \equiv \eta_{\text{flop}} \cdot B_\epsilon + (1 - \eta_{\text{flop}}) \cdot \max(0, B_\tau - I) \tag{8}$$

$$E = W \cdot \hat{\epsilon}_{\text{flop}} \cdot \left(1 + \frac{\hat{B}_\epsilon(I)}{I}\right) \tag{9}$$

Fig. 3 shows the roofline and the archline for the evaluated NVIDIA GTX 580. The



(a) GTX 580 Rooflines and Archlines **(b)** GTX 580 Powerlines

Figure 3 Example Diagrams for the different plots from [5]

roofline model and its associated time balance point B_τ are shown in red. The archline model, the effective energy balance point $\hat{B}_\epsilon(I)$ and the theoretical energy balance point B_ϵ are shown in blue. In all cases, the left dotted line and value indicates $\hat{B}_\epsilon(I)$ and the right dotted line indicates B_ϵ. The balance gap between B_τ and B_ϵ is quite large

and easily visible. On the other hand $\hat{B}_\epsilon(I)$ is smaller, therefore the GPU becomes energy bound in computation while being memory bound in time. This is stated as one reason for the effectivity for race-to-halt strategies to conserve energy [5, p. 665][1]. [5] concludes that for lower π_0 other strategies could therefore be better suited to conserve energy. Looking at the powerline in Fig. 3 a theoretical sharp spike in power consumption at B_τ can be seen. Eq. 10 provides the idea behind the powerline.

$$P = \frac{\pi_{\text{flop}}}{\eta_{\text{flop}}} \cdot \left[\frac{\min(I, B_\tau)}{B_\tau} + \frac{\hat{B}_\epsilon(I)}{\max(I, B_\tau)} \right] \tag{10}$$

This spike is due to an overlap of computation and data transfer in time. Since the energy consumption can not be hidden [5, p. 664], the maximum power consumption has to be the combination of the power spent on Flops and Mops. The findings of [5] include that the GTX 580 can not track the roofline closely around B_τ which could be due to the fact that the powerline predicts a much higher power consumption at this point than the GTX 580's power limit.

3.3 Benchmarks

The benchmark suite of this paper mainly consists of four micro benchmarks [11]. In the subsections below are pseudo code overviews and explanations for each. Together the benchmarks are designed to allow sanity checking of the model. To limit interference between measurement series, a random workload is submitted between benchmark runs. All benchmarks are validated to perform exactly as they are designed by inspection of the intermediate assembly code. The established benchmarks are designed to control W and Q. Tensor cores are not modeled and not part of this evalualtion. Similar, the benchmarks do not use the GPU's cache. The energies may therefore differ for codes relying heavily on cache and tensor cores. In the following subsections the four benchmarks are introduced. To provide a full picture of the GPU's performance the thread count is varied. This allows gathering data for occupancy ranging from around 3% up to 100%.

3.3.1 Baseline Energy

The baseline energy benchmark kernel allows to establish the baseline energy E_{base} required by the GPU. It serves the purpose to provide a general idea of the energy required to maintain the GPU's operating state. In essence, this benchmark is designed to capture all overheads incurred by the other benchmarks. This includes the energy cost incurred by branching operations, index computation and thread specific overheads such as warp and block scheduling. By using variable s defined outside of

Algorithm 1 Baseline Kernel

Require: nonnegative integer N, secret s
  function KERNEL(A, B)
    $threadIdx \leftarrow$ Calculate Index
    if s **then**
      shared memory$[threadIdx] \leftarrow threadIdx$

the compilation unit and a shared memory space, the compiler does not remove this kernel, as it can not know whether this branch is taken or not.

3.3.2 Memory Operation's Energy

Algorithm 2 Copy Kernel

Require: nonnegative integer N
  function KERNEL(A, B)
    $threadIdx \leftarrow$ Calculate Index
    $\mathrm{A}[threadIdx] \leftarrow \mathrm{B}[threadIdx]$

The memory benchmark is designed to achieve the highest possible bandwidth. It is an extension of the baseline kernel to include memory operations. Specifically, each thread loads one word from the main memory and immediately stores it back.

$$\text{per thread: } Q = 2 * \#\mathrm{Bytes} \tag{11}$$

This means Q is directly dependant on the number of launched threads and therefore easily tunable during the compilation, i.e., scanning the occupancy implicitly varies Q.

3.3.3 Arithmetic Operation's Energy

Algorithm 3 Flop Kernel

Require: nonnegative integer N
  function KERNEL(A, B)
    $threadIdx \leftarrow$ Calculate Index
    for $n \leq N$ **do**
      $x \leftarrow x + x \cdot t$

The arithmetic operation's benchmark extends the baseline kernel to perform arithmetic instructions. W is tunable by the parameter N during compilation which sets the number of concurrent instructions issued by a single thread.

$$\text{per thread: } W = 2 * N \tag{12}$$

Instead of loading and storing any data, each thread is supplied with a fixed input value and the result is discarded. Exactly as for the baseline kernel an external variable is used to prevent the writing of the result to memory without the compiler removing the kernel.

3.3.4 Roofline

Algorithm 4 Roofline Kernel

Require: nonnegative integer N
  function KERNEL(A, B)
    $threadIdx \leftarrow$ Calculate Index
    $t \leftarrow \mathrm{B}[threadIdx]$
    for $n \leq N$ **do**
      $x \leftarrow x + x \cdot t$
    $\mathrm{A}[threadIdx] \leftarrow x$

The roofline benchmark is designed to have a tunable intensity I, such that it can achieve both peak bandwidth in time for $I < B_\tau$ as well as peak performance for $I > B_\tau$. W is tunable by a parameter N during the compilation. This allows the compiler to completely unroll the kernel. Similarly, Q can be set during the compilation. W and Q are the same per thread as in arithmetic and copy kernels. In essence, it limits the amount of memory accesses by reducing the total count of launched threads. As each thread has a fixed number of memory accesses, the amount of memory operations is reduced.

3.4 Evaluated Hardware

	A100 SXM4 40GB	A100 SXM4 80GB	Grace Hopper GH200
$p_{\text{peak},64}$	$9.7\,\text{TFlop}/s$	$9.7\,\text{TFlop}/s$	$34\,\text{TFlop}/s$
$p_{\text{peak},32}$	$19.5\,\text{TFlop}/s$	$19.5\,\text{TFlop}/s$	$67\,\text{TFlop}/s$
b_{peak}	$1.6\,\text{TB}/s$	$2.0\,\text{TB}/s$	$4.0\,\text{TB}/s$
TDP	$400\,\text{W}$	$500\,\text{W}$	$900\,\text{W}$

Table 1 Key performance numbers of the evaluated NVIDIA GPUs [7][8]

	MI210
$p_{\text{peak},64}$	$22.6\,\text{TFlop}/s$
$p_{\text{peak},32}$	$22.6\,\text{Tflop}/s$
b_{peak}	$1,600\,\text{GB}/s$
TDP	$300\,\text{W}$

Table 2 Key Performance numbers of the evaluated AMD GPU [14]

For this thesis NVIDIA's A100 and GH200 GPUs as well as AMD's MI210 data center GPU are evaluated. Tab. 1 and Tab. 2 list the key performance parameters of these

GPUs. The values have been taken from their data sheets ([7], [8], [14]). $p_{\mathrm{peak},64}$ and $p_{\mathrm{peak},32}$ denote the peak double and single precision floating point performance in billion floating point operations per second (TFlop/s). The peak bandwidth b_{peak} is given in gigabyte per second (GB/s). These GPUs or members of their architecture family provide a good overview of well adopted modern GPUs [13][23]. This selection was also affected by GPU availability at NHR@FAU. The A100 family, introduced in 2021, is the oldest GPU architecture and the GH200, introduced in 2024, represents the newest architecture. This should allow for a comparison between different GPU generations and provide an overview of general trends for the power consumption properties.
For the GH200 the GPU and CPU form a superchip. The GPU die is connected to the CPU by NVLink, allowing faster communication bandwidths than the traditional Peripheral Component Interconnect Express (PCIe) connection. Additionally, this allows the GPU to access the CPU's memory directly without the need to copy data first. Since we will focus on the properties within a single GPU daughter board, the actual difference of sockets and connection speeds will not play a role in the determination of the power consumption properties. However, the type of socket might make a difference for the power limit of the GPUs. One example is the A100 SXM4 80GB variant which comes with different power limits depending on the configuration. The PCIe variants have a power limit of $400\mathrm{W}$ and the SXM4 variants have a power limit of either $400\mathrm{W}$ or $500\mathrm{W}$. The exact limit depends on whether or not the server into which the card is placed meets the requirements for NVIDIA's DGX certification. The superchip defines a package powerlimit for the combination of GPU and CPU and additionally a hard power limit for the GPU. This powerlimit sits at $900\mathrm{W}$. However, as long as the total power consumption is below the package powerlimit, this powerlimit seems not to be strictly enforced, i.e., measurements show the GH200 drawing close to $940\mathrm{W}$.
As can be seen in Tab. 1 and Tab. 2 newer generations of hardware provide more peak floating point performance and a higher bandwidth. Even within a generation differences to these parameters might exist. In addition to the larger main memory, the A100 SXM4 80GB variant runs the memory interface at a higher clock frequency to achieve a greater bandwidth than both its PCIe 80GB and SXM4 40GB counterparts. However, these numbers alone are not enough to determine a selection based on energy efficiency. Since each of these cards have different power limits, comparison of energy efficiency between them is not trivial. Additionally, the package powerlimit of the GH200 GPU makes such comparisons nearly impossible.

3.5 Peak Performance on the Hardware

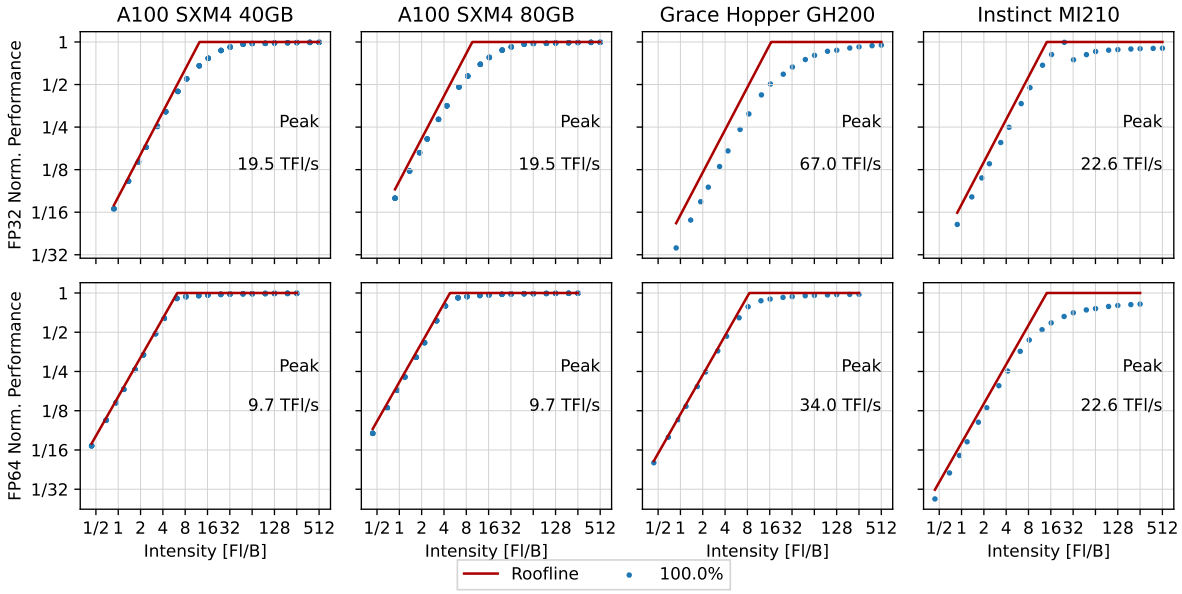


Figure 4 Peak Performance of the Roofline Kernel for $100\,\%$ Occupancy

Fig. 4 shows the performance of the roofline kernel in blue for the different GPUs. The roofline model is shown as a red line. The roofline benchmark serves as source of information for the reproduction and evaluation of the archline model [5]. From the results of this benchmark the values for ϵ_{flop}, ϵ_{mem} and π_0 are estimated. The other three benchmarks are supposed to act as cross examination opportunities. They can be used individually as input to the linear regression model or directly to compute an approximation of these parameters. Each benchmark is run on multiple GPUs and different machines in order to minimize external influences. Only for the GH200 GPU the measurements are repeated multiple times, as only a single GPU was available. The baseline kernel serves as a direct approximation of E_{base}. Going back to Eq. 2 the direct values for either arithmetic operations or memory operations can be computed by subtracting the E_{base} estimation from the measured E, as E_{active} . In Tab. 3, all of the kernels' performances are given as percentages of the peak performance for each GPU. Note that the roofline kernel's performance is given by percentage of peak arithmetic performance and by percentage of peak bandwidth. As can be seen, the kernels are able to achieve significant proportions of the peak architecture capabilities. The baseline benchmark is not included as it is only aimed at capturing the power consumption of the GPU at its highest operating frequency.

GPU	Precision	Arithmetic Op	Memory Op	Roofline	
		$p_{\text{peak}}\,\%$	$b_{\text{peak}}\,\%$	$p_{\text{peak}}\,\%$	$b_{\text{peak}}\,\%$
A100 40GB	FP32	99.52	86.95	99.52	94.87
	FP64	99.70	89.31	99.65	96.25
A100 80GB	FP32	99.52	80.77	99.52	92.06
	FP64	99.68	86.43	99.68	92.71
GH200	FP32	96.71	64.29	96.47	76.91
	FP64	97.9	82.19	97.73	97.28
MI210	FP32	91.02	77.99	90.33	85.44
	FP64	77.76	85.96	82.19	89.39

Table 3 Percentages of the peak bandwidth and performance for each kernel, GPU and precision

4 Data Collection

In this chapter an overview of the different hardware and software designs is provided. Sec. 4.1 describes the differences to socket designs and Sec. 4.2 details the software setup. In this section the exact driver interfaces, their intricacies and the proposed measurement loop are shown.

4.1 Hardware

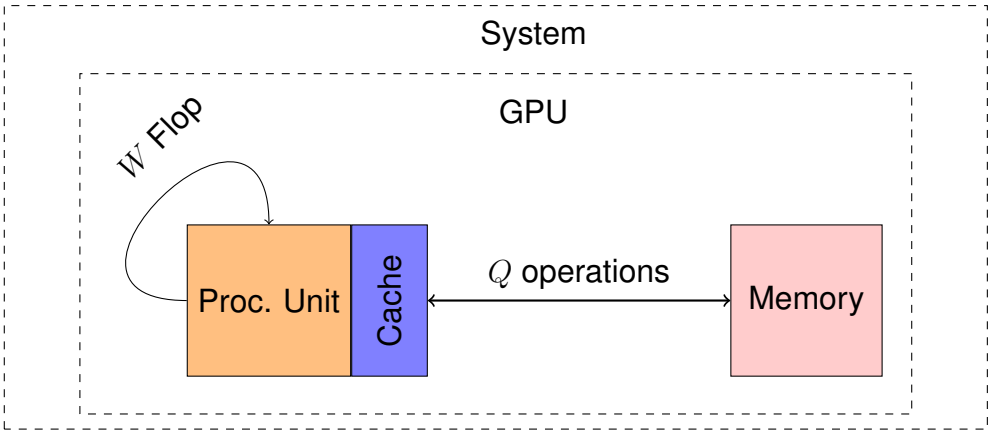


Figure 5 Typical hardware configuration of a GPU accelerated system [5]

In general a GPU accelerated compute node contains one or more GPUs. Fig. 5 shows an abstract view of such a system. Depending on the configuration, the GPUs can be interconnected by additional hardware. This thesis focuses exclusively on the power consumption properties for single GPU workloads within the GPU's main memory, i.e., no communication of data occurs from or to the GPU during the benchmark kernel. The GPU is connected to the rest of the system by PCIe lanes. GPUs are seated on a backplane either in a PCIe–slot or specialized sockets, e.g., Server PCI Express Module (SXM) or OCP Accelerator Module (OAM) sockets. PCIe is currently the wide spread data center form factor. However, specialized sockets offer higher bandwidths and better connections between different accelerators [20][10].
The power delivery systems between these different backplanes vary. As stated in the PCIe standard, a maximum of $75\mathrm{W}$ can be drawn through the slot [19]. Supplemental power can be provided to allow for higher power consumption. For the described specialized standards, the GPUs are supplied with power through the socket. This variety in design means specialized interposers have to be designed to measure the actual power draw. Such a measuring setup would be expensive to build and would require a larger software package to interact. Therefore, the GPU's built-in power measurement capabilities will be used to create a software based power measurement loop.

4.2 Measurement

Measuring of power consumption and consequently measuring of energy can be tricky. On dedicated GPU form factors this can be done by physical sensors as well as through software. On integrated designs such as the GH200, physical measurement of the GPU die's power consumption would require an even more complex approach. Historically, specialized hardware such as Interposers [5], designed to interface between the GPU and the system, have been used. In recent years, the software based approach has become more and more feasible. The accuracy of the built-in sensors has increased and findings show that these sensors correlate strongly with the actual power consumption [3]. In fact, NVIDIA claims a maximum of 5% deviation [9] from on-board measurements to the actual power draw. While this does not sound like much at first, for a $500\mathrm{W}$ GPU class this means a tolerance of $\pm 25\mathrm{W}$. Therefore, the measured dataset has to be able to account for variations. With this in mind, these measurements should allow to make a reasonable prediction of the behaviour of the GPU's power consumption. The results may not be the exact physical properties, but should allow to compute the power consumption properties. This approach has the benefit that no specialized hardware is needed.
Choosing a driver based approach has other benefits as well. Both NVIDIA and AMD follow a very similar driver design and interface approach with their software stacks. For NVIDIA, the NVML [9] and CUDA [6] and for AMD, ROCm System Management (ROCm) [16] and HIP libraries are used. Programmers can run codes on either software stack mostly by replacing the four letters "cuda" to "hip". For this conversion, AMD even provides an automatic tool to translate to and from cuda. The benchmarks and the introduced measurement code are written such that this portability does not affect the performance of the kernels. However, some differences between NVIDIA's CUDA and AMD's ROCm specifications do exist. The measurement program can be tuned to achieve a specific intensity as well as to run with a varying accelerator occupancy. For every variation the code is recompiled to allow the compiler to unroll all loops.

4.2.1 NVIDIA Driver Intricacies

The NVIDIA Management Library [9] is the programming interface to interact with and to manage the device during runtime. It provides information about the GPU's active power limits and consumption as well as clock frequency. One quirk of NVML is that it does not explicitly state the exact time interval at which new values are made available to the programmer. Additionally, not all GPUs measure the power consumption in the same way, some might not even support polling the power consumption. But for most of the data center GPUs such information is made available.

Some older generation models as well as the A100 series GA100 chip, report instantaneous power readings. On other GPUs NVIDIA states that power readings are averaged over a time interval of 1 second. This interval itself is too long to make fine grained assumptions of the kernels energy consumption. Power measurement can be made for shorter intervals by repetition of the kernel over multiple power reading intervals [17]. With this in mind, a first measurement run indicates, that NVML seems to make a new power reading available at a $20ms$ interval. Additionally, the kernel runtime and number of repetitions are selected in a way that an accurate estimation of the mean power consumption during the execution of a single benchmark can be made.

4.2.2 AMD Driver Intricacies

Unlike NVIDIA, AMD's ROCm reports instantaneous power measurements on newer devices. Older GPU generations typically provide averaged power measurements, newer models are set to discard these averaged measurements entirely [15]. By querying "rsmi_dev_power_get", either the instantaneous power or an averaged power are returned [16]. Further, ROCm provides access to an energy counter with a resolution of microJoules together with a resolution modifier for this value. Both can be polled by calling the function "rsmi_dev_energy_count_get". In the ROCm documentation a $1ms$ interval is specified for new values to be made available [16]. The MI210 seems to not entirely support such a short interval, tests revealed new values to be made available between 1–$2ms$. Overall, this allows the creation of very accurate power profiles. Additionally, "rocm–smi" provides access to the devices performance counters.

4.2.3 Measurement Loop

Algorithm 5 Measuring Loop

```
Setup, Memory Initialization
Prepare and Check Device Attributes
Record Start Event
for i ≤ #iterations do
    call kernel
Record End Event
Synchronize to Start Event
Poll Data while Waiting for End Event to Complete
```

The measuring loop begins by initializing all needed memory on the device. Following, the device's active power limit and the clock frequency are checked. With the goal of highest performance, the GPU should be in its highest operating state, i.e., the clock frequency and power limit should be at their maximum. The power limit can be set

through either “nvidia–smi” or “rocm–smi”. To ensure that the clock frequency and the operating state match their highest levels, an initialization kernel is launched as necessary. On both platforms, a kernel is launched to a stream execution model. The more general device synchronization can be used to achieve host and device synchronisation without regard to the stream. This method blocks the host until the device has completed all work that has previously been submitted to the stream. To allow asynchronous polling of the completion, the event management capabilites of ROCm and CUDA are used. These events can be inserted into the stream and can be used to signal the completion of previously submitted work. Contrary to device synchronisation not all work has to be completed within the stream for this synchronization. Both CUDA and ROCm allow for non-blocking and blocking synchronization and querying of an event. Additionally, events can be used to exclude the time overhead of the synchronization from the total execution time. This means that the driver measures the exact timestamp of the completion of the event on the device. Therefore, these timestamps are very accurate. In its most basic form, two events are needed to measure the kernel execution. One event to signal the beginning of the benchmark kernel execution, the other to signal the end. Overall, a single benchmark run launches the kernel multiple times such that the total execution time for all benchmark runs is significantly larger than the power measurement update intervals. This approach can be seen in Alg. 5. The first event is recorded to the stream immediately after the initialization kernel launch. Then, a predetermined number of iterations of the benchmark kernel is launched. After the kernel is scheduled the second event is recorded to the stream. In this approach, the first event is used for two purposes. On the one hand it serves as a marker of the beginning of kernel execution and records the actual starting time. On the other hand it is used to block the host until kernel execution begins. Therefore, the initialization kernel is excluded from the actual power measuring. In combination with the second event, the total execution time of the actual kernel execution can be computed. As the completion of the second event can be queried without blocking, the host can record the reported power measurements and the clock frequencies [11] of the device while periodically checking the status of the second event.
Recording of the reported clock frequencies is important since the model expects the device’s operating state to remain unchanged during the benchmark execution. However, due to hardware resource usage and other factors such as heat and power limits the operating state may change during the execution. The collected information about clock frequencies allows the exclusion of entries with an altered operating state.
As mentioned in Sec. 4.2, the reported power measurements have a tolerance of $\pm 5\%$. The power measurements and execution times are averaged over all the datasets. Information about the operating state of the GPU is kept mostly unchanged, to ensure the correct exclusion.

5 Results

This chapter contains the results obtained from fitting the model to the benchmarks' result data. The results are split into two sections, one for single precision and the other for double precision results. In these sections the obtained results and the resulting graphs are shown and discussed. Similar to Fig.3 the roofline performance of the benchmark is printed in red, the archline performance in blue and the powerline in black, each showing multiple accelerator occupancies as colored dots. Plotted is the average of the multiple combined measurements. The variance is shown with error bars and the occupancies are shown in different colors.

5.1 Accuracy of the Model

	precision	R^2	R^{2*}
A100 40GB	FP32	0.9991	0.9991
	FP64	0.9969	0.9969
A100 80GB	FP32	0.9996	0.9996
	FP64	0.9995	0.9995
GH200	FP32	0.9978	0.9978
	FP64	0.9993	0.9993
MI210	FP32	0.9936	0.9917
	FP64	0.9946	0.9964

Table 4 Accuracy of the model for the GPUs

Tab. 4 shows the accuracy of the model for each of the architectures and precisions. R^{2*} shows the Coefficient of Determination for the full dataset, including the data points with altered clock frequencies. As can be seen for all GPUs and precisions, the model is able to explain the energy usage. R^2 is never below 0.99. Even for the full data set, this accuracy can be obtained. The impact of the operating state alterations can be seen in a small deviation for R^{2*}. Since none of the NVIDIA GPUs made such alterations, $R^2 = R^{2*}$. Therefore, it can be concluded that the model's parameters largely account for the overall energy consumption. The lowered clock frequencies simply lead to altered values for the different parameters.

The baseline, arithmetic operations and memory benchmarks provide a possibility to examine the fitted results. Starting with the baseline benchmark, π_0 can be computed by dividing the average measured energy by the averaged time. To obtain a reference value for ϵ_{flop} and ϵ_{mem} the respective Eq. 6 and Eq. 7 are used.

	Precision	ϵ_{flop} [pJ/Fl]	ϵ_{mem} [pJ/B]	π_0 [W]	R^2
A100 40GB	FP32	7.56	77.62	101.2	0.9948
	FP64	15.93	73.54	96.98	0.9965
A100 80GB	FP32	8.23	90.72	109.75	0.9953
	FP64	16.61	74.23	106.91	0.9995
GH200	FP32	4.63	97.08	161.94	0.9956
	FP64	14.82	96.33	164.82	0.9997
MI210	FP32	8.84	82.98	108.71	0.9968
	FP64	19.32	89.06	107.21	0.9985

Table 5 Power consumption properties of the reference benchmarks

Tab. 5 lists the values for FP32 and FP64 of the reference benchmarks as well as their R^2. The accuracy of the fit for these subsets of the general model is above 0.99 as well. This allows for the conclusion that the independant variables capture the intended properties. The model allows the fitting of the parameters for the reference benchmarks. These values are all within 10% of the combined benchmark's fitted parameters. This supports the assumption of the proposed model classifying the architectural intricacies similarly in these different cases. The fitted value for π_0 varies slightly. This variance is about expected from the reported driver measurement accuracies. The small variance in π_0 allows the conclusion that for both precisions the overheads are similarly attributed to the different parameters.

5.2 Single Precision

Starting with the single precision benchmark results, the graphs in Fig. 6 show the performance of the roofline kernel both for the archline and for the powerline. For all GPUs the time balance point B_τ, the effective energy balance point $\hat{B}_\epsilon(I)$ and the theoretical energy balance point B_ϵ if $\pi_0 = 0$ are shown. As expected, the kernel deviates from the roofline slightly around B_τ for all GPUs. The exact reason for this deviation is unclear. The GPUs seem to be unable to fully utilize the resources due to some architectural overheads. In the case of the MI210 GPU the low roofline performance is caused by the power limit of 300W. In Fig. 7 the powerline, the actual power consumption and the machine power limits are shown. Overall, for NVIDIA GPUs $B_\epsilon > B_\tau$ is true but not by a large margin. On the MI210, B_ϵ is lower than B_τ. Additionally, it has the lowest B_ϵ of all evaluated GPUs at 9.6Fl/B. The A100 variants have a B_ϵ of 15 to 15.3Fl/B, followed by the GH200 with 19.7Fl/B. For all evaluated GPUs $\hat{B}_\epsilon(I)$

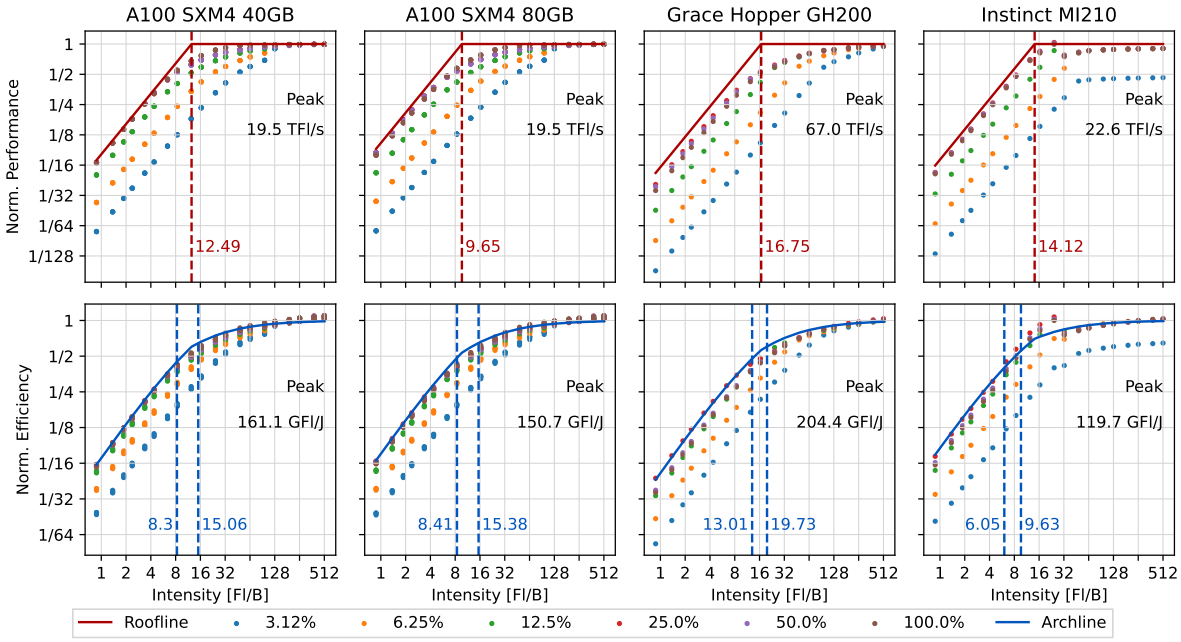


Figure 6 FP32 Rooflines and Archlines for the evaluated GPUs

is lower than B_τ. Interestingly, all GPUs except the MI210 are able to reach the peak performance in energy for the different occupancies. The MI210 is not able to reach the peak performance with only vector FP32 instructions on the lowest occupancy. Additionally, the data set contains packed FP32 instructions, i.e., instructions performing double the floating point operations compared to vector instructions. Since the compiler generated these for I lower than $16\mathrm{Fl/B}$, the graphs contain some artifacts. The performance in time can be seen to dip above the expected $22.6\mathrm{TFl}/s$. For packed instructions the MI210 can achieve a theoretical peak of $44.6\mathrm{TFl}/s$. Therefore, on these lower intensities, the GPU achieves a higher than expected peak performance in energy. This means the findings of [5] are valid on modern hardware. However, the balance gap sizings have decreased significantly. Additionally, the MI210's low B_ϵ implies that race-to-halt will be the ideal strategy to conserve power. On the A100 40GB and GH200 B_τ is centered in the balance gap. However on the A100 80GB $\hat{B}_\epsilon(I)$ is close to B_τ. Effectively, time efficiency barely implies energy efficiency. This allows the assumption that other strategies than race-to-halt could be used for power usage reduction. The NVIDIA GPUs all have powerlimits higher than the predicted powerline. This means, that in theory all cards should be able to achieve their peak performance even at B_τ. The measurements show the powerline accurately tracking the actual power usage closer to the sides. Near B_τ, all cards consume less power than expected. Since the roofline kernel carries some overhead it can not achieve peak performance close to B_τ. Therefore, due to the lower achieved bandwidth and computing performance, the power consumption has to be lower than the powerline predicts.

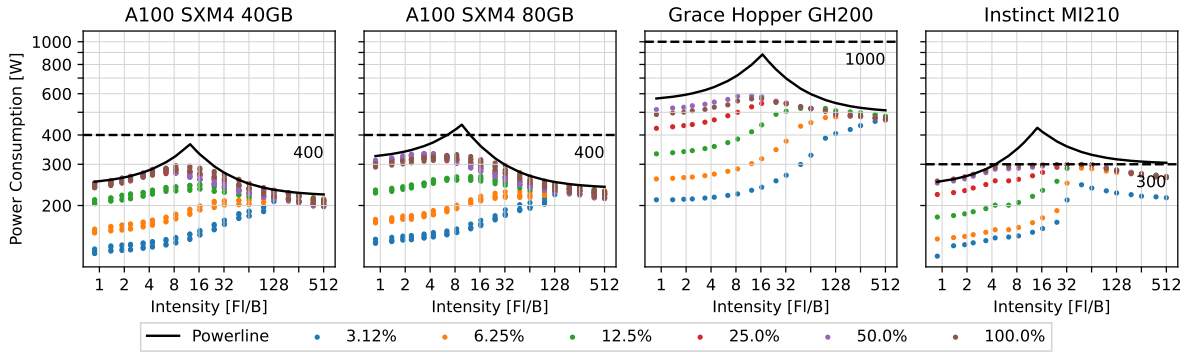


Figure 7 Powerlines for the evaluated GPUs FP32

Well visible is the impact of occupancy on accelerator power consumption. When the maximum achieved bandwidth of the GPU is limited, B_τ will shift towards the right. This explains the increase in power consumption for higher $\mathrm{Fl/B}$ than the machine balance point would suggest. After reaching these points the power consumption then drops again, except on the NVIDIA GPUs, where the lowest occupancy does not hit the GPU's peak performance within the measured set of intensities. The transition from memory bound to compute bound code execution is largely seen on these GPUs as an increase in power consumption. The fitted values for π_0, ϵ_{flop} and ϵ_{mem} can be seen

	ϵ_{flop} $[p\mathrm{J/Fl}]$	ϵ_{mem} $[p\mathrm{J/B}]$	ϵ_{mop} $[p\mathrm{J/Op}]$	π_0 [W]
A100 40GB	6.21	93.48	373.9	98.42
A100 80GB	6.63	102.06	408.2	107.21
GH200	4.89	96.53	386.1	169.27
MI210	8.36	80.48	321.9	112.00

Table 6 FP32 Fitted Parameters for the combined benchmark

in Tab. 6. Note that ϵ_{flop} is given in pico-Joule per Flop, ϵ_{mem} in pico-Joule per Byte, ϵ_{mop} in pico-Joule per Mop and π_0 in Watt. Overall, the evaluated GPUs all show a noticable power consumption due to their baseline energy. The reason behind the higher baseline power of the A100 80GB in comparison to the A100 40GB lies primarily in the difference of the memory clock frequency. This frequency increase is the reason behind the 80GB variant's higher memory bandwidth. Additionally, ϵ_{mem} at $102.06p\mathrm{J/B}$ is also higher than the 40GB's $93.48p\mathrm{J/B}$. Even more interesting is the increased energy consumption per Flop. Across all tests, the A100 80GB's ϵ_{flop} is fitted around $0.5p\mathrm{J/Fl}$ higher than its 40GB counterpart. Since the model makes significant simplifications, this could be explained by artifacts from the interaction with the memory subsystem. Between the older and newer generations, a reduction in ϵ_{flop} can be seen.

However, on the MI210 GPU ϵ_{flop} is higher than on all the measured NVIDIA cards. Since the GPUs have different baseline power consumptions, the Flop energy efficiency η_{flop} is better suited to compare between these GPUs.

	π_{flop} [W]	π_{mem} [W]	η_{flop}
A100 40GB	121.0	145.8	55.14%
A100 80GB	129.3	206.2	54.67%
GH200	327.8	386.1	65.94%
MI210	188.8	128.8	62.77%

Table 7 Flop Energy Efficiency of the measured GPUs

Tab. 7 shows the Flop energy efficiency as energy per Flop divided by the sum of energy per Flop and baseline power per Flop for the measured GPUs:

$$\eta_{\text{flop}} = \epsilon_{\text{flop}}/(\epsilon_{\text{flop}} + \epsilon_{0,\text{flop}}) \tag{13}$$

$\epsilon_{0,\text{flop}}$ denotes the baseline power expended during the completion of one Flop, i.e., $\epsilon_{0,\text{flop}} = \tau_{\text{flop}} \cdot \pi_0$. It highlights the importance of using energy efficiency to compare between the different GPUs. Even though the MI210's has a larger ϵ_{flop}, its comparatively low $\epsilon_{0,\text{flop}}$ allows for a high η_{flop}. Therefore, the MI210 can be seen as the more energy efficient GPU. Between the two A100 variants, η_{flop} shows the impact of the higher memory clock frequency. For pure energy efficiency, the A100 40GB wins by about 1%. The GH200 shows the benefits of high levels of integration. At $p_{\text{peak},32}$ its 150W baseline power and 330W π_{flop} yield an impressive 66% energy efficiency. The rising power limits therefore do not imply a stagnation of energy efficiency. Also interesting to note is that all GPUs consume a lot of power for memory operations.

5.3 Double Precision

For the FP64 benchmarks, the roofline and archline are shown in Fig. 8. As can be seen, under 100% occupancy, the roofline kernel tracks the roofline very accurately. Both the A100 40GB and A100 80GB GPUs are able to achieve their peak performance under all occupancies. Similar to the MI210 the GH200 GPU does not achieve its peak perfomance in time. Overall, the deviation from the roofline is lower, as the various overheads of the device's architecture are not as pronounced for double precision operations. However, all NVIDIA GPUs are able to achieve their peak performance in energy for the different occupancies. B_τ, B_ϵ and $\hat{B}_\epsilon(I)$ are shown. For the MI210 B_ϵ is significantly lower than B_τ. Furthermore, the A100 40GB and GH200 have a slightly lower B_ϵ than their B_τ. Only the A100 80GB has a higher B_ϵ than B_τ due to its higher

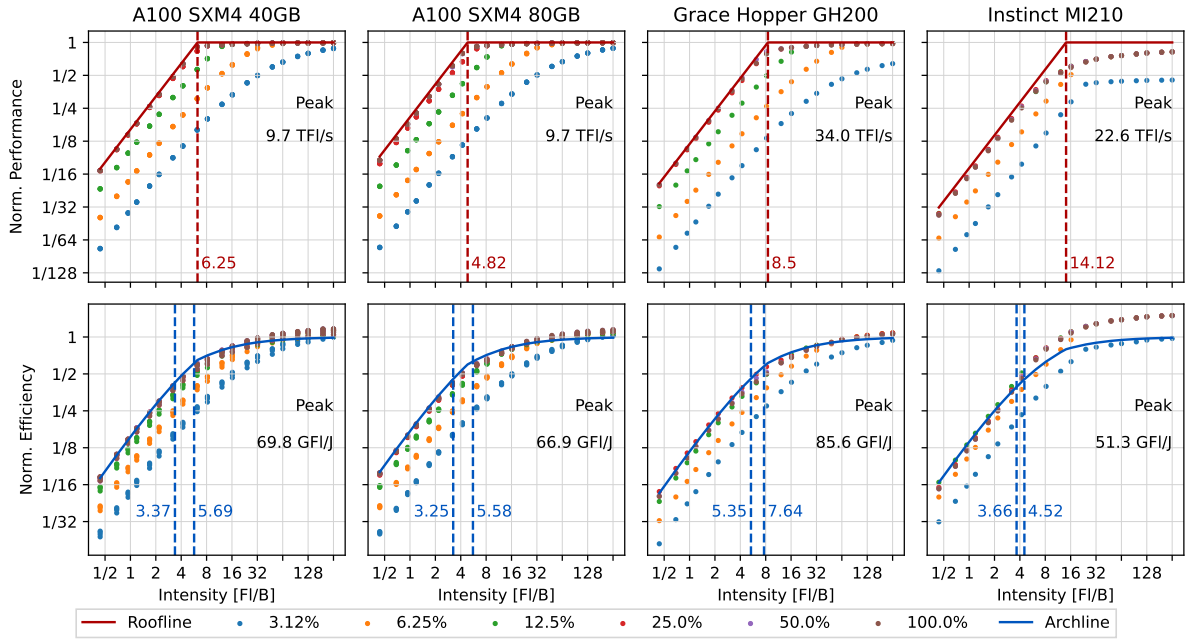


Figure 8 FP64 Rooflines and Archlines for the evaluated GPUs

bandwidth and therefore lowered B_τ. Interesting to note is that its B_ϵ is slightly lower than the 40GB variant's B_ϵ. The archline model does reflect the general trend in performance over energy, but underestimates the possible peak performance slightly.
On the MI210 the model completely misses the actual energy by a factor of 1.8 for higher intensities. This can be shown to be due to clock throttling as the GPU hits its power limit and reduces the clock frequencies starting on $I = 2$. All GPUs have $\hat{B}_\epsilon(I) < B_\tau$. For the A100 80GB the balance gap is centered around B_τ. For the other GPUs the balance gap is below B_τ. The powerline for the different GPUs is shown in

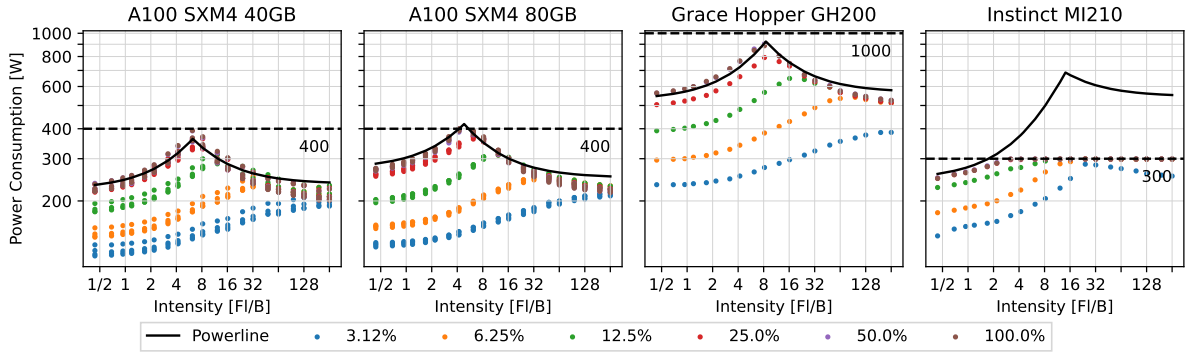


Figure 9 Powerlines for the evaluated GPUs FP64

Fig. 9. It accurately tracks the kernel's power measurements. Therefore, the spike in power consumption is pronounced for the NVIDIA GPUs. However, the MI210 does not develop such a spike. This is due to its low power limit of $300\mathrm{W}$. The GPU therefore

is unable to achieve peak performance. Through all occupancies, a drop in clock frequency can be observed. Since power consumption is largely dependant on clock frequencies this reduction can be attributed to the device enforcing its power limit. The different occupancies show the movement of B_τ very well as all GPUs can not satisfy their bandwidth for the lowered occupancies. Overall, the NVIDIA GPUs do not show a spike in power consumption for the lowest occupancy. The MI210 on the other hand does show such an inflection point. This is interesting, as it reaches the powerlimit even on this reduced occupancy. For higher intensities a drop can be seen, implying that the MI210 is hardware limited in terms of compute power. Unsurprisingly, the GH200's power consumption is higher than any other GPU. Both the memory bound and the compute bound power consumption are beyond the power consumption of the other GPUs. Looking at Tab. 8 the reason is obvious. While ϵ_{flop} has decreased a little, the rise in $p_{\text{peak,64}}$ and also the increase in b_{peak} necessarily lead to an increase in total board power draw. However, the models highest predicted power consumption stays below the power limits of the NVIDIA GPUs. From the measurements, no alteration of clock frequencies could be observed. As the plots show, the powerline almost exactly matches the power consumption of these cards. The fitted values are listed in

	ϵ_{flop} [$p\text{J/Fl}$]	ϵ_{mem} [$p\text{J/B}$]	ϵ_{mop} [$p\text{J/Op}$]	π_0 [W]
A100 40GB	14.33	81.59	652.7	96.08
A100 80GB	14.96	83.45	667.6	104.23
GH200	11.68	89.22	713.8	170.10
MI210	19.49	88.15	705.2	104.53

Table 8 Fitted parameters for FP64 instructions

Tab. 8. For the A100 40GB an ϵ_{flop} of $14.33p\text{J/Fl}$ and for the A100 80GB an ϵ_{flop} of $14.96p\text{J/Fl}$ are fitted. The difference between these is most probably due to the difference in memory clock speeds. The higher memory clock can be easily observed in π_0. Additionally, a small difference can be observed to the GH200's ϵ_{flop} at $11.68p\text{J/Fl}$. This shows that between these generations, a single floating point operation has decreased in energy usage. For ϵ_{mem}, these generational findings differ. Between the NVIDIA GPUs the A100 40GB has the lowest energy consumption and the GH200 has the highest. The MI210 GPU comes in at around $88.15p\text{J/Fl}$, slightly less than the GH200. Interestingly, between the A100 80GB's and MI210's π_0 nearly no difference is measured. The GH200 has by far the greatest π_0 at around 1.6 times the π_0 of the other GPUs. With its performance being significantly higher than the performance of the other cards this power consumption is not unsurprising as the larger transistor count also needs more energy to stay at peak clock frequencies. Tab. 9 shows the

	π_{flop} [W]	π_{mem} [W]	η_{flop}
A100 40GB	139.7	127.3	59.24%
A100 80GB	145.8	168.6	58.31%
GH200	397.0	356.9	70.00%
MI210	440.5	141.0	80.82%

Table 9 FP64 Flop Energy Efficiency of the measured GPUs

Flop energy efficiency and the peak power consumption of Mop and Flop. Of peculiar interest is η_{flop} of the MI210. The model assigns 80% energy efficiency, which would be an incredible feat, given that the GH200 achieves 70%. However, as stated in 3.1.2, the proposed model does not take power caps into account.
The total Flop power is 440W, together with π_0 this would rise to around 540W. As Fig. 9 shows, the MI210's power consumption never rises above its power limit at 300W. Therefore, it can never actually achieve this flop efficiency, invalidating this efficiency at around $I = 2$. However, the reduced clock frequencies due to clock throttling allow the GPU lower π_0, ϵ_{flop} and ϵ_{mem} at the cost of peak performance in time. Also important to note is the deviation in total power spent on memory transfers. For the NVIDIA GPUs a nearly equal amount of power is needed for memory transfers compared to floating point operations. The MI210 changes this relationship with the power spent on memory transfers at around one third of the power consumption of Flop. This carries quite some significance, since it allows the machine's B_ϵ for $\pi_0 = 0$ to sit well below B_τ.

5.4 Discussion

GTX 580	ϵ_{flop} [$p\text{J/Fl}$]	ϵ_{mem} [$p\text{J/B}$]	ϵ_{mop} [$p\text{J/Op}$]	π_0 [W]
FP32	99.7	513	2052	122
FP64	212	513	2052	122

Table 10 Reference Values for the NVIDIA GTX 580 [5]

The fitted values for all of these GPUs show their theoretical performances under maximum clock frequencies. Tab. 10 lists the fitted parameters for a NVIDIA GTX 580 as evaluated by [5]. Comparing these results to current GPUs, a significant reduction of both ϵ_{flop} and ϵ_{mem} can be seen. This is in line with the expectations of higher efficiency from better and smaller physical transistor designs. Additionally, GPUs pack a much higher transistor count, leading to similar π_0 values in comparison to the GTX 580. This is paired with a similar increase in computing performance. Overall, looking

	π_{flop} [W]	π_{mem} [W]	η_{flop}
FP32	157.6	73.9	56.38%
FP64	41.8	73.9	25.56%

Table 11 Flop Energy Efficiency of the NVIDIA GTX 580 GPUs [5]

at η_{flop}, the GTX 580 achieves 56.44% FP32 and 25.56% FP64 efficiency. While the FP32 efficiency can compete with modern GPUs, the FP64 efficiency certainly can not. This can partly be attributed to the GTX 580's intended use for consumer applications which typically rely on FP32 performance. This is interesting as it shows that efficiency has not necessarily gone up over the years. The demand for increased performance and therefore increasingly higher clock frequencies hides the architecture's better energy designs. For FP32, the increase in performance and the reduction of computation time lead to about similar η_{flop}.

Altogether, modern GPUs have very similar η_{flop} for both FP32 and FP64. Since data center GPUs typically have half the performance in FP64 compared to FP32, the baseline energy per flop $\epsilon_{0,\text{flop}}$ for FP64 is two times that of the FP32 instructions. Similarly, the energy per flop ϵ_{flop} is doubled, therefore it is expected that η_{flop} is approximately the same. In practice this means the energy efficiency stays consistent. With the MI210 having the same performance in both FP64 and FP32, an increase in power efficiency for FP64 performance can be seen. The influence of π_0 on the FP64 power consumption is not as significant. This is interesting, as the MI210 uses the same execution units for both precisions, with FP32 at half the register width. This allows for a packed instruction, where two FP32 values are computed simultaneously in one execution unit. For such instructions, a specific load pattern has to be achieved. This raises computing performance up to $45T$Flop. In this case π_{flop} rises to approximately 370W, similar to the double precision power consumption. However, as stated in Sec. 5.3, the MI210 is severely limited by its power limit and can therefore achieve this performance only on very small intensities. All GPUs do not require high intensity to become compute bound in energy for FP64. This can be considered as an advantage, since increasing a code's intensity proves very difficult. The important goal of energy efficiency is therefore easier to achieve than that of time efficiency. Additionally, all GPUs are compute bound in energy well before being compute bound in time. This means that race-to-halt will be a valid strategy for power conservation and, except for the A100 80GB, the only strategy in FP64.

When looking at memory architecture differences, a key difference is in the GH200 using newer HBM3 memory. The advancements over HBM2e are increased stack size and bus clock frequency. While the architecture itself got a little more energy efficient,

the increased clock frequency overshadows any energy gain. Compared to the A100 variants, in FP32 ϵ_{mem} is similar but for FP64 ϵ_{mem} is higher than for the older HBM2e standard. In both cases the data transfers are more or about similar expensive compared to the MI210's ϵ_{mem}. The increase in bandwidth performance can therefore be largely attributed to an increase in the number of memory chiplets. The effect of power reduction by clock throttling can be seen well for the MI210.
Overall, the energy required for floating point operations has decreased by a factor of 12 to 15 and the energy required for memory operations has decreased by a factor of 5 for FP32. In FP64 a reduction factor of 10 to 12 for floating point operations and a factor of 6 for transferred Bytes can be observed.

6 Conclusion

The archline and powerline models can be used to make an accurate estimation of the peak performance in energy and power consumption of modern GPUs. While the architectures have become increasingly more performant, the key findings of [5] remain valid. However, modern GPUs display a significantly smaller balance gap than the GTX 580. Furthermore, for FP64 even B_ϵ is lower than B_τ for most GPUs. Therefore, race-to-halt strategies are gaining in importance in order to conserve power on modern architectures. The models can be used to analyze and predict a kernels performance and to guide the development of a given kernel.

Overall, the energy requirements for data transfers and floating point operations have dropped significantly. But there exist differences in the extent of the drops for each of these properties. While the energy for Flops has decreased by a factor of approximately 12 the energy required for data transfer has dropped only by a factor of 5. The performance gains for Flop mostly relate to advancements in transistor sizing but also increased transistor count and clock frequencies. The performance gain for data transfers rely much more on an increase in clock frequencies and in the number of concurrently accessible memory modules, but only to a lesser degree in transistor advancements. This can be seen through the difference between High Bandwidth Memory (HBM)2e and HBM3, the latter's increased stack size and clock frequencies corresponding to the performance gain over the former's. Additionally, the cost to transfer data to main memory can be assumed to consist to a large degree of the energy required to travel the distance between the different chips on the GPU.

Similar to the GTX 580 some modern GPUs, like the MI210, can not support their peak performance due to power limitations. The effect of power limitations and how they can be integrated into the model are the key to the evaluation of such GPUs with higher accuracy. In recent years hardware designs have become increasingly integrated, meaning more transistors and dies are build in close vicinity and within the same board. This allows for a lower baseline power while increasing the peak power draw of the GPUs. In the near future this trend will probably continue as increasing the density of the GPU dies is becoming more difficult. Therefore, integrating power limit awareness into the model may prove useful for evaluating the optimum operating frequencies of the device. Additionally, most GPUs support a range of operating frequencies. As can be seen through the impact of clock frequency reduction on the MI210, these frequencies can have a large impact on total device power consumption. Exploration towards hardware performances under different frequencies is necessary to gather a complete picture of a GPU. Possible other topics for exploration include a more detailed evaluation of the memory hierarchy, i.e., the cost to access different caches apart from just DRAM transfers.

6.1 Usability

The proposed measurement loop is written in such a way to permit usage for arbitrary GPU kernels. It is set into a library file exposing three wrapper functions to the programmer. Additionally, it is thread-safe, i.e., the library is usable for different GPUs on multiple threads simultaneously. To initialize the library, a call to “initMeasurement” must be made after initializing the CUDA or ROCm runtime. This will create the events. Optionally, a stream can be specified to which the measurement should be synchronized. Immediately before launching the kernel, a call to “startMeasurement” records the begin event. In case no stream was specified during initialization, the default stream will be used. After the kernel launch, a call to “endMeasurement” will record the end event. This function will block the calling thread until the measured section has completed computation and the gathered data is returned in a custom data structure. The first value in the returned data structure is the execution time in milliseconds. The other two entries are vectors, the first containing the measured power consumption in milliwatt and the second containing the corresponding clock frequencies in megahertz. The tool will be integrated into the *gpu-benches* repository [11].

7 Acknowledgements

Scientific support and the High-Performance Computing (HPC) resources for this thesis were kindly provided by the Erlangen National High-Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU). Erlangen National High-Performance Computing Center (NHR@FAU) funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG) – 440719683.

The benchmarks for this paper were written as an extension to the GPU analysis tool suite created by Dominik Ernst [11]. All analysis tools are available under GPL-3.0 and the extensions will be integrated into the gpu roofline microbenchmark. The energy measurement tool will be made available as an extension to the “gpu–stats.h” library header.

References

[1] Muhammad Ali Awan and Stefan M. Petters. “Race-to-halt energy saving strategies”. In: *Journal of Systems Architecture* 60.10 (2014), pp. 796–815. ISSN: 1383-7621. DOI: https://doi.org/10.1016/j.sysarc.2014.10.001. URL: https://www.sciencedirect.com/science/article/pii/S1383762114001295.

[2] Daniel Bedard et al. “PowerMon: Fine-grained and integrated power monitoring for commodity computer systems”. In: *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*. 2010, pp. 479–484. DOI: 10.1109/SECON.2010.5453824.

[3] Robert A. Bridges, Neena Imam, and Tiffany M. Mintz. “Understanding GPU Power: A Survey of Profiling, Modeling, and Simulation Methods”. In: *ACM Comput. Surv.* 49.3 (Sept. 2016). ISSN: 0360-0300. DOI: 10.1145/2962131. URL: https://doi.org/10.1145/2962131.

[4] Victoria Caparrós Cabezas and Markus Püschel. “Extending the roofline model: Bottleneck analysis with microarchitectural constraints”. In: *2014 IEEE International Symposium on Workload Characterization (IISWC)*. 2014, pp. 222–231. DOI: 10.1109/IISWC.2014.6983061.

[5] J. Choi et al. “A Roofline Model of Energy”. In: *Parallel and Distributed Processing Symposium, International*. Los Alamitos, CA, USA: IEEE Computer Society, May 2013, pp. 661–672. DOI: 10.1109/IPDPS.2013.77. URL: https://doi.ieeecomputersociety.org/10.1109/IPDPS.2013.77.

[6] NVIDIA Corporation. *CUDA Toolkit*. URL: https://developer.nvidia.com/cuda-toolkit.

[7] NVIDIA Corporation. *NVIDIA A100 Tensor Core GPU*. 2021.

[8] NVIDIA Corporation. *NVIDIA GH200 Grace Hopper Superchip*. 2024.

[9] NVIDIA Corporation. *NVidia Management Library*. URL: https://developer.nvidia.com/management-library-nvml.

[10] NVIDIA Corporation. *Server PCI Express Module*. URL: https://en.wikipedia.org/wiki/SXM_(socket).

[11] Dominik Ernst. *gpu-benches*. URL: https://github.com/te42kyfo/gpu-benches.

[12] Rong Ge et al. “PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications”. In: *IEEE Transactions on Parallel and Distributed Systems* 21.5 (2010), pp. 658–671. DOI: 10.1109/TPDS.2009.76.

[13] *Green500*. 2024. URL: https://top500.org/lists/green500/.

[14] AMD Inc. *AMD Instinct MI210 Accelerator*. 2022.

[15] AMD Inc. *AMD Instinct MI300A APU*. 2023.

[16] AMD Inc. *ROCm Software*. URL: https://www.amd.com/de/products/software/rocm.html.

[17] Jens Lang and Gudula Rünger. “High-Resolution Power Profiling of GPU Functions Using Low-Resolution Measurement”. In: *Euro-Par 2013 Parallel Processing*. Ed. by Felix Wolf, Bernd Mohr, and Dieter an Mey. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 801–812. ISBN: 978-3-642-40047-6.

[18] André Lopes et al. “Exploring GPU performance, power and energy-efficiency bounds with Cache-aware Roofline Modeling”. In: *2017 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. 2017, pp. 259–268. DOI: 10.1109/ISPASS.2017.7975297.

[19] PCI-SIG. URL: https://pcisig.com/specifications.

[20] Open Compute Project. *Open Accelerator Infrastructure Open Accelerator Module*. URL: https://www.opencompute.org/wiki/Server/OAI.

[21] scikit-learn. *Linear Regression*. URL: https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LinearRegression.html.

[22] Dhanabalan Thangam et al. “Impact of Data Centers on Power Consumption, Climate Change, and Sustainability”. In: Mar. 2024, pp. 60–83. ISBN: 9798369315521. DOI: 10.4018/979-8-3693-1552-1.ch004.

[23] *Top500*. 2024. URL: https://top500.org/lists/top500/.

[24] Samuel Williams, Andrew Waterman, and David Patterson. “Roofline: an insightful visual performance model for multicore architectures”. In: *Commun. ACM* 52.4 (Apr. 2009), pp. 65–76. ISSN: 0001-0782. DOI: 10.1145/1498765.1498785. URL: https://doi.org/10.1145/1498765.1498785.

Declaration of Academic Integrity

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde.

Die Stellen der Arbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Nürnberg, 5th December 2024

Jonah Holtmann