SCALE–Ahead-Of-Time Compilation of CUDA for AMD GPUs

Manos Pavlidakis, Chris Kitching, Nicholas Tomlinson, and Michael Søndergaard
Spectral Compute, United Kingdom
{manos,chris,nstomlinson,ms}@spectralcompute.co.uk

Abstract

SCALE is a new solution by Spectral Compute that empowers developers to write code once and deploy it across a range of GPU hardware platforms without modifying the original code. Designed to extend CUDA's capabilities to AMD GPUs, SCALE maintains CUDA compatibility while introducing novel features that streamline GPU programming. This demo paper presents SCALE's functionalities and impact, particularly its implications for lowering development costs, enhancing flexibility, and addressing supply chain concerns within the GPU ecosystem.

CCS Concepts

• **Computer systems organization → Single instruction, multiple data**; **Multicore architectures**; **Heterogeneous (hybrid) systems**; • **Software and its engineering → Compilers**; • **General and reference → Cross-computing tools and techniques**; • **Computing methodologies → Parallel programming languages**;

Keywords

SCALE, CUDA, ROCm, AMD GPUs, Ahead-of-Time Compilation, Cross-Platform Development, Code Portability

1 Introduction

Graphics Processing Units (GPUs) have become essential for accelerating a wide range of applications, including machine learning (ML), deep learning (DL), scientific simulations, data analytics, computational biology, and computer vision [5–8]. However, the GPU market relies heavily on NVIDIA's CUDA platform, the standard for GPU programming, due to its robust libraries, wide adoption, and extensive developer community. Yet, CUDA's exclusivity to NVIDIA GPUs creates a significant limitation, restricting developers to a single hardware vendor. For organizations and developers, this vendor lock-in introduces various challenges, including increased costs, limited hardware flexibility, and risks associated with supply chain dependencies.

Several alternatives attempt to address CUDA's platform exclusivity, but each has limitations. HIP [1] and SYCL [3], AMD's and Intel's counterparts to CUDA, introduce new GPU languages and APIs, each with tools [1, 3] for partial code conversion. However, these tools present several key challenges. First, HIPify and SYCLomatic cannot fully convert all CUDA code, leaving developers to manage significant manual modifications. Second, to achieve optimal performance on NVIDIA GPUs, developers often need to maintain a CUDA version of the code, which undercuts the goal of a unified, cross-platform solution. Finally, the CUDA dialect issue compounds these challenges: the CUDA language lacks a formal specification, with its "standard" being loosely defined by NVIDIA's `nvcc` compiler behavior. While `clang` can compile CUDA, it operates with a different CUDA dialect, which can lead to subtle semantic inconsistencies. Additionally, HIP and SYCL do not support inline PTX assembly blocks in CUDA code, meaning these must be manually removed or guarded with macros, further complicating code maintenance.

SCALE, developed by Spectral Compute, addresses these issues head-on by enabling seamless CUDA compatibility on AMD GPUs. This innovative platform allows developers to run their CUDA applications on AMD hardware without any code modifications, unlocking the freedom to select the best GPU for their needs regardless of the hardware vendor. Beyond simple compatibility, SCALE introduces powerful language extensions that make GPU programming more accessible and efficient, ensuring that developers benefit from modern coding practices and improved code maintainability. SCALE effectively addresses the following challenges:

CUDA Platform Exclusivity. CUDA applications can only run on NVIDIA GPUs, locking developers into a single vendor and limiting hardware flexibility. SCALE enables CUDA applications to run on AMD GPUs without modifications, breaking the dependency on NVIDIA hardware and allowing developers to select GPUs based on performance and feature requirements rather than compatibility.

CUDA Portability and Maintenance. HIP and SYCL, require developers to rewrite or heavily modify their CUDA code to achieve compatibility with non-NVIDIA GPUs, leading to fragmented codebases and high maintenance costs. SCALE does not introduce a new language. Instead, it allows programs written using the widely popular CUDA language to be directly compiled for AMD GPUs using SCALE's "`nvcc`"-compatible compiler. Our compiler directly compiles PTX assembly to the AMD equivalent. As a result, users do not need to have multiple codebases or compromise on performance to support multiple GPU vendors.

Avoid Dynamic-linking library interception. CUDA call interception leads to security concerns and is unstable because NVIDIA can always make it unfeasible. SCALE includes a complete re-implementation of CUDA's runtime and driver APIs for AMD GPUs, which provides native performance and supports CUDA's features, such as inline PTX, enabling optimized and fully functional execution on AMD hardware. Regarding CUDA accelerated libraries, such as cuBLAS and cuSOLVER, SCALE uses the ROCm equivalent.

2 The SCALE Platform

SCALE is designed as an extension to CUDA, maintaining full compatibility with the CUDA programming model while adding optional features that enhance programming productivity. SCALE's platform offers the following key features:

- **Full CUDA Compatibility with AMD GPUs:** SCALE allows CUDA code to be executed on AMD GPUs without requiring any modifications. This is achieved through a

complete re-implementation of CUDA's runtime and driver APIs.

- **Enhanced Language Extensions:** SCALE introduces language extensions that simplify GPU programming by adding modern C++ features, such as RAII (Resource Acquisition Is Initialization) and exception handling, to the CUDA API. These extensions improve code readability, maintainability, and expressiveness, allowing developers to write cleaner and more robust GPU code that leverages advanced C++ paradigms.
- **Compatibility with AMD's SoTA GPUs:** Currently, SCALE ensures performance optimization across state-of-the-art AMD GPU micro-architectures, including gfx1030 and gfx1100.
- **Extensive Testing Across Open-Source Projects:** SCALE has been rigorously tested on a broad set of open-source CUDA projects, including NVIDIA Thrust, Blender Cycles, AMGX, llama-cpp, faiss, xgboost, GOMC, stdgpu, and hashcat, confirming its robustness and versatility in various GPU computing applications.

3 Demonstration

Our demo session highlights SCALE's real-world applications and showcases its flexibility. Attendees will experience SCALE's ability to run CUDA-based applications on AMD GPUs with minimal setup, offering a hands-on perspective on how SCALE translates CUDA's programming model into an AMD-compatible framework without compromising performance. In particular, we will demonstrate how projects implemented in CUDA, like LLaMA [2] and AMGx [4], run without any modifications on AMD GPUs. It is worth noting that, although AMGx lacks native ROCm support, it can run seamlessly on AMD GPUs through SCALE. Additionally, we will showcase the cuBLAS library and the execution of inline PTX code on AMD GPUs, highlighting SCALE's robust support for CUDA libraries and low-level assembly, further underscoring its compatibility and performance advantages.

4 Impact on the GPU Ecosystem

The introduction of SCALE marks a significant shift in the GPU market. Traditionally, developers have faced the challenges of vendor lock-in and high porting costs when working with CUDA on non-NVIDIA hardware. By providing CUDA compatibility for AMD GPUs, SCALE alleviates these constraints, offering several transformative advantages:

- **Freedom in Hardware Selection:** SCALE enables developers to select GPU hardware based on performance and feature set, rather than compatibility constraints, thus fostering a more flexible and dynamic development environment.
- **Cost and Time Savings:** Avoiding the need to rewrite or maintain multiple versions of CUDA code reduces development costs and speeds up deployment cycles.
- **Reduced Supply Chain Concerns:** SCALE addresses potential supply chain issues by making AMD GPUs a viable alternative for CUDA-based applications, thus broadening access to compatible hardware.
- **Optimization for AMD GPUs:** The SCALE platform leverages AMD's GPU architectures, allowing CUDA applications to perform efficiently on AMD hardware.

5 Conclusion

SCALE offers a powerful new paradigm for GPU development by extending CUDA compatibility to AMD GPUs. Through its innovative platform, SCALE eliminates the need to port away from CUDA, providing GPU developers with unprecedented flexibility, cost savings, and hardware options. SCALE is poised to become a critical tool in the GPU market, empowering developers to choose the best hardware for their applications while remaining within the CUDA ecosystem.

Acknowledgments

We would like to express our sincere gratitude to the team members at Spectral Compute who contributed to the development of SCALE. Special thanks go to Francois Souchay, Matthew Ireland, Niki Järvinen, Ruben van Dongen, Andrei Stepanenko, Finlay Hudson, Vytautas Mickus, Patrick Ville, Allen Philip, Vanessa Oxley, Miron Zadora, and Justine Khoo for their invaluable efforts and expertise.

References

[1] Advanced Micro Devices. 2020. HIPify. Retrieved May 2023 from https://docs.amd.com/bundle/HIPify-Reference-Guide-v5.1/page/HIPify.html
[2] Georgi Gerganov. 2024. Inference of Meta's LLaMA model (and others) in pure C/C++. Retrieved November 2024 from https://github.com/ggerganov/llama.cpp
[3] Robert Mueller-Albrecht. 2024. SYCLomatic: SYCL Adoption for Everyone - Moving from CUDA to SYCL Gets Progressively Easier: Advanced Migration Considerations. In *IWOCL '24.*
[4] NVIDIA. 2023. Algebraic Multigrid Solver (AmgX) Library. Retrieved November 2024 from https://github.com/NVIDIA/AMGX
[5] Manos Pavlidakis, Stelios Mavridis, Antony Chazapis, Giorgos Vasiliadis, and Angelos Bilas. 2022. Arax: A Runtime Framework for Decoupling Applications from Heterogeneous Accelerators. In *SoCC '22.*
[6] Manos Pavlidakis, Giorgos Vasiliadis, Anargyros Argyros, Stelios Mavridis, Antony Chazapis, and Angelos Bilas. 2024. Guardian: Safe GPU Sharing in Multi-Tenant Environments. In *MIDDLEWARE '24.*
[7] Foteini Strati, Xianzhe Ma, and Ana Klimovic. 2024. Orion: Interference-aware, Fine-grained GPU Sharing for ML Applications. In *EuroSys '24.*
[8] Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. 2022. MLaaS in the wild: Workload analysis and scheduling in Large-Scale heterogeneous GPU clusters. In *NSDI '22.*