Formal methods for GPU Software Development using Ada SPARK

Dimitris Aspetakis[1], Matina Maria Trompouki[1], Leonidas Kosmidis[1]

[1]Barcelona Supercomputing Center (BSC)

Formal methods for Graphics processing units (GPUs) are increasingly considered for use in space, due to their high performance capabilities, in order to satisfy the ever increasing on-board data processing needs of future space missions. However developing software for GPUs, which are massively parallel accelerators, is more challenging than sequential CPU code. In particular, GPUs have a different address space than CPUs, and it is up to the programmer to manage explicit memory allocations and transfers through low-level, error prone operations using pointers. Moreover, parallel programming can lead to data races. These issues can challenge ECSS qualification of GPU software.

Formal methods such as the ones provided by Ada SPARK have been used for long time in CPU code to prevent programming mistakes and prove the absence of runtime errors. In the OSIP project "Formal methods for GPU Software Development", we are exploring the strengths and limitations of the application of Ada SPARK in GPU software code. In particular, AdaCore is currently developing a CUDA backend for their Ada compiler, which allows programming GPUs in Ada and its SPARK subset. In our work we show how Ada SPARK can prevent common GPU programming mistakes and can prove the absence of runtime errors such as numeric overflows or underflows, division by 0 as well as buffer overflows due to the pointer usage.

A major contribution of our work is the development of a programming methodology that conveys information from the host to the GPU code and vice versa, so that the prover has enough information to check the consistency between these two software parts. We have demonstrated our methodology by porting the open source GPU4S Bench (GPU for space) benchmarking suite [1][2][3] to Ada SPARK [4], achieving bronze SPARK adoption level. Despite the fact that the AdaCore GPU compiler is still under development and some GPU features are not yet available, our work demonstrates that Ada SPARK is a viable method to prove the correctness of GPU software for high assurance systems.

[1] I. Rodriguez et al, GPU4S Bench: Design and Implementation of an Open GPU Benchmarking Suite for Space On-board Processing, UPC Technical Report 2019, https://www.ac.upc.edu/app/research-reports/public/html/research_center_index-CAP-2019,en.html

[2] D. Steenari et al, OBPMark and GPU4S Bench repository, https://obpmark.github.io

[3] D. Steenari, L. Kosmidis, I. Rodriguez-Ferrandez, A. Jover-Alvarez, and K. Förster in 2nd European Workshop on On-Board Data Processing (OBDP2021), 2021. https://doi.org/10.5281/zenodo.5638577

[4] D. Aspetakis, GPU4S Bench Ada SPARK, https://gitlab.bsc.es/dimitris_aspetakis/gpu4s-bench-ada