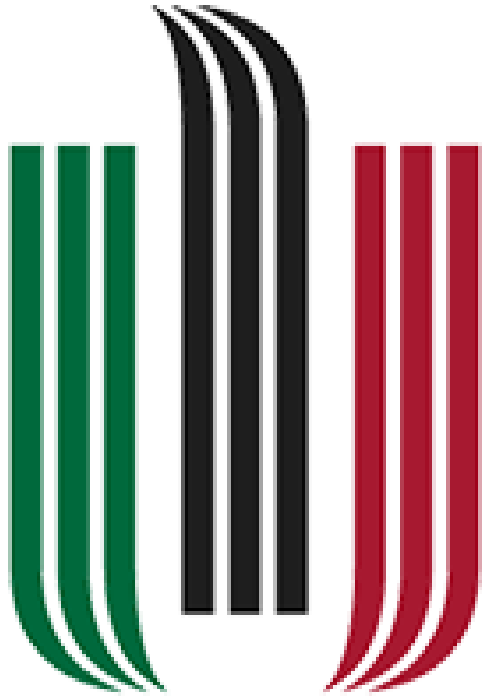


AGH

AGH University of Science and Technology

Introduction to CUDA and Open CL

Project

Wiktor Żychowicz

1. Project overview.

This project is a linear calculator which provide various operation that can be executed both on vectors and matrices. Program can be configured in numerous ways. Main features are possibility of CPU/GPU calculations, results comparing, performance measuring and terminal-based user interface.

2. Project interface.

It consist of Main Menu and five others sub Menus:

- Conf – Configuration Menu provides methods to change parameters of the program.
- Data Manage – Responsible for data manipulation (creating/destroying)
- Calculate Vectors – Implement operations on Vectors
- Calculate Matrices – Implement operations on Matrices
- Show – Is accessible from all others menus, grants tools for printing stored data

User can navigate the program by selecting appropriate numbers. Basic “Yeti” tolerance is included. Both successful actions and all sorts of errors provide feedback.

3. Project structure.

Project structure is made in CMake so it is portable and can be easily complied from source. When moving between systems you only need update appropriate flags.

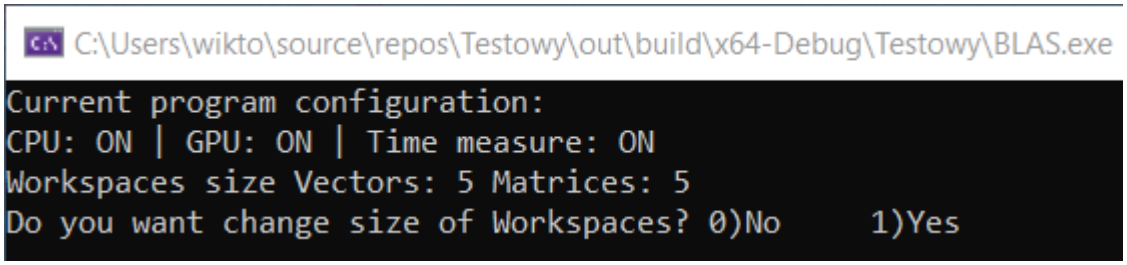
Through process of creating version control system named Git was used. Source code is available on GitHub:

https://github.com/Sightster/CUDA_Gr_5/tree/master/ProjektCUDA

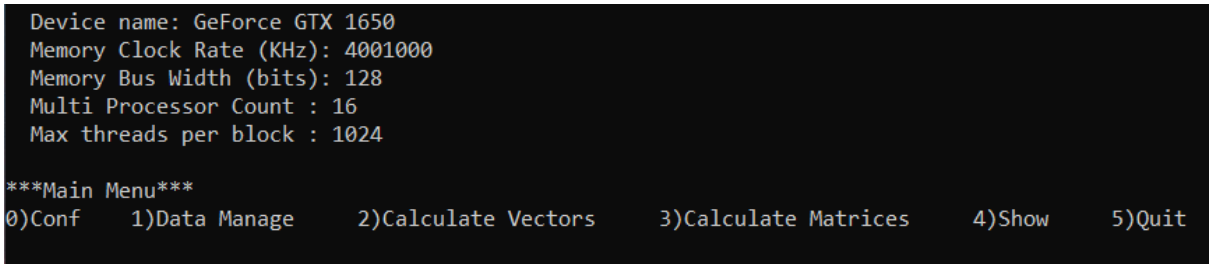
Project consist of various classes each with own header file(with documentation when needed) and source file. All CUDA code is separated thanks to that tools used in coding process didn’t have to support this NVIDIA technology.

4. Quick instruction.

When launched program gives us quick overview about its state and let user choose number od vectors/matrices he will be using (result of operations also must be stored in vector):



We can change other parameters in config menu but sizes of workspaces are constant during execution.
After that program scan our computer and optimizes itself to our GPU.
From now on various Menus guide us.



5. Failures and improvements possibilities.

Unfortunately it turned out that to use __shared memory its size must be hard-coded into program. As one of the design assumption was to provide user opportunity to configure this algebra calculator usage of algorithms with shared memory was not an option this problem remains unsolvable.

Program structure give a way to use cuBLAS library (similarly to cudaSamples code which was used in time measuring implementation) however algebra calculator use int vectors instead of floats (choice dictated by easier CPU/GPU comparisons). This problem could be easily solved by usage of C++ templates in the first place.

From time perspective implementation of vectors as a separate object instead of for example one dimensional matrices was a huge obstacle in terms of development time.

Development could be also improved by more frequently updating remote VCS servers.