МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования
«**Московский технический университет связи и информатики**»
(**МТУСИ**)

Кафедра «Системное программирование»

ОТЧЕТ

по лабораторной работе №1
Вариант №27

по дисциплине «**Системное программное обеспечение**»

Выполнил:
студент гр. БФИ2202
_____ Сидорук Д. В.
«____» _____ 2026 г.

Проверил:
старший преподаватель
_____ Алексанян Д. А.
«____» _____ 2026 г.

Москва, 2026 г.

Содержание

1 Введение

Развитие технологий параллельных вычислений привело к широкому применению графических процессоров (GPU) не только для обработки графики, но и для решения ресурсоёмких вычислительных задач общего назначения. Архитектура современных графических ускорителей ориентирована на массовый параллелизм и высокую пропускную способность памяти, что делает их эффективным инструментом для научных расчётов, обработки больших объёмов данных и задач машинного обучения.

Платформа NVIDIA CUDA предоставляет программную модель и инструменты для разработки приложений, использующих вычислительные возможности GPU. Эффективное применение CUDA требует понимания аппаратных характеристик устройства, таких как вычислительная способность (compute capability), размеры блоков и сеток потоков, объёмы различных типов памяти, а также параметры выполнения, включая размер варпа.

Перед началом разработки параллельных алгоритмов важно уметь получать и анализировать информацию о доступных графических устройствах. Эти сведения определяют ограничения на конфигурацию запуска ядер, объём используемых ресурсов и потенциальную производительность приложения. Исследование аппаратных параметров GPU позволяет сформировать базовое представление о среде выполнения и подготовить основу для дальнейших лабораторных работ, связанных с программированием и оптимизацией вычислений на графических ускорителях.

2 Цель работы

Целью данной лабораторной работы является ознакомление с аппаратными ресурсами CUDA-совместимого устройства и процессом компиляции и запуска программы, выполняющей запрос характеристик GPU.

3 Задание

Скомпилировать и запустить код, получающий информацию о доступных устройствах CUDA.

4 Ход работы

В листинге ниже приведено содержание файла rvs_laboratories_0/main.cu, который содержит код, получающий информацию о доступных устройствах CUDA (лист. 1).

Лист. 1 – Содержание rvs_laboratories_0/main.cu

```
#include "wb.h"

int main(int argc, char **argv) {
  int deviceCount;

```

```
6     wbArg_read(argc, argv);
7
8     cudaGetDeviceCount(&deviceCount);
9
10    wbTime_start(GPU, "Getting GPU Data.");
11
12    for (int dev = 0; dev < deviceCount; dev++) {
13      cudaDeviceProp deviceProp;
14
15      cudaGetDeviceProperties(&deviceProp, dev);
16
17      if (dev == 0) {
18        if (deviceProp.major == 9999 && deviceProp.minor == 9999) {
19          wbLog(TRACE, "No CUDA GPU has been detected");
20          return -1;
21        } else if (deviceCount == 1) {
22          wbLog(TRACE, "There is 1 device supporting CUDA");
23        } else {
24          wbLog(TRACE, "There are ", deviceCount,
25                "  devices supporting CUDA");
26        }
27      }
28
29      wbLog(TRACE, "Device ", dev, " name: ", deviceProp.name);
30      wbLog(TRACE, " Computational Capabilities: ",
        ↪  deviceProp.major, ".",
31            deviceProp.minor);
32      wbLog(TRACE, " Maximum global memory size: ",
33            deviceProp.totalGlobalMem);
34      wbLog(TRACE, " Maximum constant memory size: ",
35            deviceProp.totalConstMem);
36      wbLog(TRACE, " Maximum shared memory size per block: ",
37            deviceProp.sharedMemPerBlock);
38      wbLog(TRACE, " Maximum block dimensions: ",
39            deviceProp.maxThreadsDim[0], " x ",
              ↪  deviceProp.maxThreadsDim[1],
40            " x ", deviceProp.maxThreadsDim[2]);
41      wbLog(TRACE, " Maximum grid dimensions: ",
        ↪  deviceProp.maxGridSize[0],
42            " x ", deviceProp.maxGridSize[1], " x ",
43            deviceProp.maxGridSize[2]);
44      wbLog(TRACE, " Warp size: ", deviceProp.warpSize);
45    }
46
47    wbTime_stop(GPU, "Getting GPU Data.");
48
49    return 0;
50  }
```

В листинге ниже приведено содержание файла CMakeLists.txt, определяющего правила сборки программы (лист. 2).

Лист. 2 – Содержание CMakeLists.txt

```
project(rvs_laboratories_0)

find_package(CUDA REQUIRED)

cuda_add_executable(rvs_laboratories_0 rvs_laboratories_0/main.cu)
```

В листинге ниже приведен результат сборки и выполнения программы (лист. 3)

Лист. 3 – Результат работы программы

```
eoanermine@eoanermine:~/Repositories/rvs_laboratories_0/build$ cmake ..
-- The C compiler identification is GNU 13.3.0
-- The CXX compiler identification is GNU 13.3.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD - Success
-- Found Threads: TRUE
-- Found CUDA: /usr/local/cuda (found version "13.1")
-- Configuring done (4.2s)
-- Generating done (0.1s)
-- Build files have been written to: ~/Repositories/rvs_laboratories_0/build
eoanermine@eoanermine:~/Repositories/rvs_laboratories_0/build$ make
[ 50%] Building NVCC (Device) object CMakeFiles/rvs_laboratories_0.dir/rvs_laboratories_0/rvs_laboratories_0_generated_main.cu.o
[100%] Linking CXX executable rvs_laboratories_0
[100%] Built target rvs_laboratories_0
eoanermine@eoanermine:~/Repositories/rvs_laboratories_0/build$ ./rvs_laboratories_0
Trace main::22 There is 1 device supporting CUDA
Trace main::29 Device 0 name: NVIDIA GeForce RTX 5060
Trace main::30  Computational Capabilities: 12.0
Trace main::32  Maximum global memory size: 8546484224
Trace main::34  Maximum constant memory size: 65536
Trace main::36  Maximum shared memory size per block: 49152
```

```
Trace main::38  Maximum block dimensions: 1024 x 1024 x 64
Trace main::41  Maximum grid dimensions: 2147483647 x 65535 x
 ↪  65535
Trace main::44  Warp size: 32
[GPU    ] 0.000106752 Getting GPU Data.
```

Заключение

В ходе выполнения лабораторной работы были изучены основные аппаратные характеристики графического процессора, доступные через API CUDA. Получены сведения о вычислительной способности устройства, объёмах глобальной, константной и разделяемой памяти, максимальных размерностях блока и сетки, а также размере варпа.

Анализ полученных параметров позволяет оценить ограничения и возможности конкретного GPU при запуске параллельных вычислений. Понимание этих характеристик является необходимым условием для корректной конфигурации потоков, рационального распределения ресурсов и предотвращения ошибок при выполнении программ.

Освоенные в работе приёмы компиляции и запуска CUDA-приложений формируют практическую основу для дальнейшего изучения параллельного программирования на графических ускорителях и перехода к более сложным задачам оптимизации и анализа производительности.