**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №5**

**по дисциплине «Параллельные алгоритмы»**

Тема: Знакомство с программированием гетерогенных систем в стандарте Open CL

|  |  |  |
| --- | --- | --- |
| Студент гр. 0303 |  | Парамонов В.В. |
| Преподаватель |  | Сергеева Е. И. |

Санкт-Петербург

2023

## Цель работы.

Изучить фреймворк OpenCL для написания программ, связанных с параллельными вычислениями на различных графических и центральных процессорах. Реализовать расчёт фрактала Мандельброта на OpenCL.

**Постановка задачи.**

1. Реализовать расчет фрактала Мандельброта на OpenCL.
2. Визуализировать результат.
3. Произвести оценку производительности.

**Выполнение задач.**

1. **Реализация программы:**
2. В файлах *opencl\_handler.cpp* и *opencl\_handler.hpp* созданы функции для инициализации и работы с OpenCL: create\_device нужно для выбора конкретного устройства на компьютере для вычислений, create\_context - создает контекст для устройства, build\_program - компилирует программу для использования на выбранном устройстве, create\_buffer – создает буфер для передачи данных между хостом и устройством вычисления, fill\_buffer – заполняет буфер значениями, create\_queue – создает очередь команд для подачи на контекст, create\_kernel – создает ядро для выполняения программы на устройстве, set\_kernel\_arg – устанавливает параметр для ядра, send\_to\_execution – отправляет программу на ядре на выполнение на устройстве, read\_result\_from\_buf – чтение результата из буфера.
3. В utility.cpp, utility.hpp находятся дополнительные функции: read\_file – для чтения содержимого файла в строку, ppm\_draw – для записи вычисленного фрактала в файл ppm картинки.
4. mandelbrot.cl – содержит код для вычисления фрактала Мальденброта, который выполняется на устройстве под управлением OpenCL.
5. main.cpp – главный файл проекта, который запускает процесс вычисления фрактала и получение изображения.
6. **Визуализация полученного результата:**

После работы программы был получен следующий результат (см. рисунок 1):

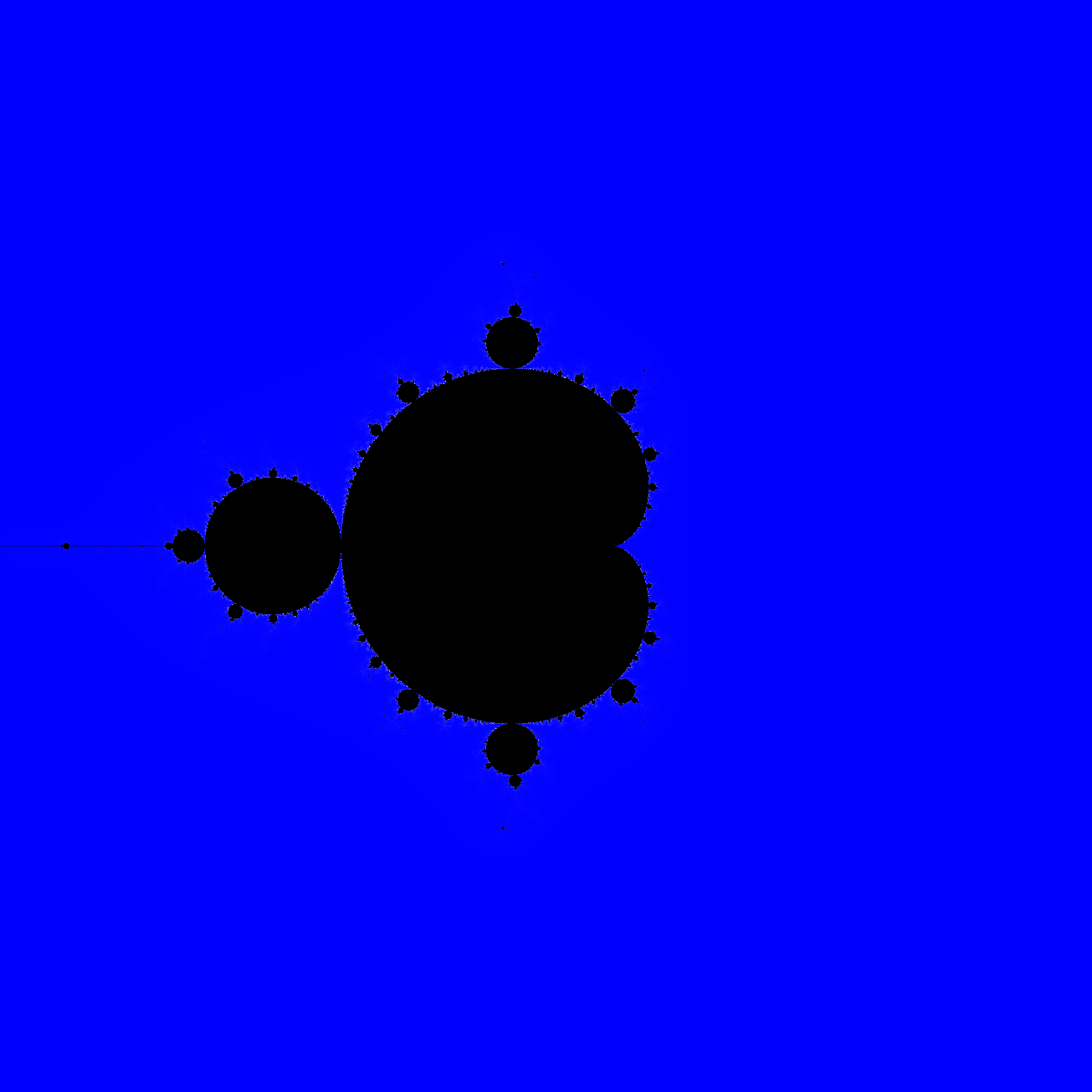


Рисунок 1 – Вычисленный фрактал Мальденброта

1. **Оценка производительности:**

* Зависимость времени выполнения от количества итераций для отрисовки фрактала (на фиксированных параметрах x\_max = 4; y\_max = 4; width = 2000; height = 2000) см. в таблице 1:

Таблица 1 – Время выполнения от числа итераций вычисления фрактала

|  |  |  |
| --- | --- | --- |
| Количество итераций | Intel встроенная, время выполнения (сек) | Nvidia дискретная, время выполнения (сек) |
| 100 | 0.008682 | **0.008642** |
| 200 | 0.031224 | **0.012403** |
| 300 | **0.01556** | 0.01665 |
| 400 | **0.021714** | 0.047249 |
| 500 | **0.030765** | 0.06369 |
| 600 | **0.044727** | 0.065868 |
| 700 | 0.056507 | **0.055117** |
| 800 | 0.04537 | **0.034338** |
| 900 | 0.053168 | **0.041994** |

* Зависимость времени выполнения от ширины и высоты обрабатываемого множества фрактала (на фиксированных параметрах x\_max = 4; y\_max = 4; iterations = 1000) см. в таблице 2:

Таблица 2 – Время выполнения от ширины и высоты

|  |  |  |
| --- | --- | --- |
| Ширина и высота (пиксели) | Intel встроенная, время выполнения (сек) | Nvidia дискретная, время выполнения (сек) |
| 800 | 0.035172 | **0.009678** |
| 1600 | 0.038852 | **0.034459** |
| 2400 | 0.076155 | **0.074201** |
| 3200 | 0.131717 | **0.130847** |
| 4000 | **0.200918** | 0.228936 |
| 4800 | **0.287071** | 0.289579 |
| 5600 | 0.391913 | **0.390787** |
| 6400 | **0.508236** | 0.512457 |
| 7200 | 0.641895 | **0.639952** |

* Зависимость времени выполнения от максимальных значений реальной и мнимой части для вычисления фрактала (на фиксированных параметрах width = 2000; height = 2000; iterations = 1000) см. в таблице 3:

Таблица 3 – Время выполнения от максимальных значений реальной и мнимой части для вычисления фрактала

|  |  |  |
| --- | --- | --- |
| x\_max, y\_max | Intel встроенная, время выполнения (сек) | Nvidia дискретная, время выполнения (сек) |
| 1 | **0.344504** | 0.350404 |
| 2 | **0.148138** | 0.14885 |
| 3 | **0.0820554** | 0.0820953 |
| 4 | **0.0541535** | 0.0545258 |
| 5 | **0.0435677** | 0.0436616 |
| 6 | **0.0329335** | 0.0389207 |
| 7 | **0.0286191** | 0.0286602 |
| 8 | 0.027294 | **0.0256272** |
| 9 | 0.026096 | **0.0237956** |

Исходя из данных таблиц встроенная видеокарта Intel и дискретная Nvidia показали примерно одинаковую производительность, однако испытания не чистые, так как и дискретная Nvidia параллельно с отрисовкой фрактала использовалась в отрисовке графики компьютера и нескольких приложений. По параметрам вычисления фрактала получились следующие оценки: чем больше число максимальное число итераций вычисления фрактала, тем выше время вычисления; так же с высотой и шириной множества; для максимальных значений реальной и мнимой части фрактала ситуация обратная, чем она больше, тем выше скорость вычисления (так как область, занимаемая множеством фрактала, уменьшается).

**Заключение.**

В ходе работы был изучен фреймворк OpenCL для написания программ, связанных с параллельными вычислениями на различных графических и центральных процессорах. Был реализован расчёт фрактала Мандельброта на OpenCL. Как показала оценка производительности в “не совсем справедливой” оценке встроенная видеокарта Intel и дискретная Nvidia показали примерно одинаковую производительность.