

# Теория Параллелизма

## Отчет

### Уравнение теплопроводности (орепасс)

Выполнил 21932, Бабенко Егор Степанович

02.03.2022

## Цель работы

Реализовать решение уравнение теплопроводности методом Якоба (пятиточечный шаблон) для двумерной сетки. Произвести профилирование программы для GPU и оптимизацию кода. Произвести сравнение времени работы на CPU и GPU.

Используемый компилятор: *pgc++*

Для компиляции различных версий использовались команды:

- `pgc++ t2.cpp -o t2_MultiCore.pg -fast -acc=multicore -O2 -Mconcur=allcores`
- `pgc++ t2.cpp -o t2_OneCore.pg -fast -O2`
- `pgc++ t2.cpp -o t2_GPU.pg -fast -acc=gpu -O2 -D OPENACC__`

Для дополнительной профилировки: `-D NVPROF_`

Используемый профилировщики: *nvprof*, *nsys*

Nsys использовался с OpenACC trace, а также NVTX trace.

Замер времени работы всей программы производился с помощью команды `time`. Время высчитывалось как среднее между пятью запусками.

## CPU-onecore

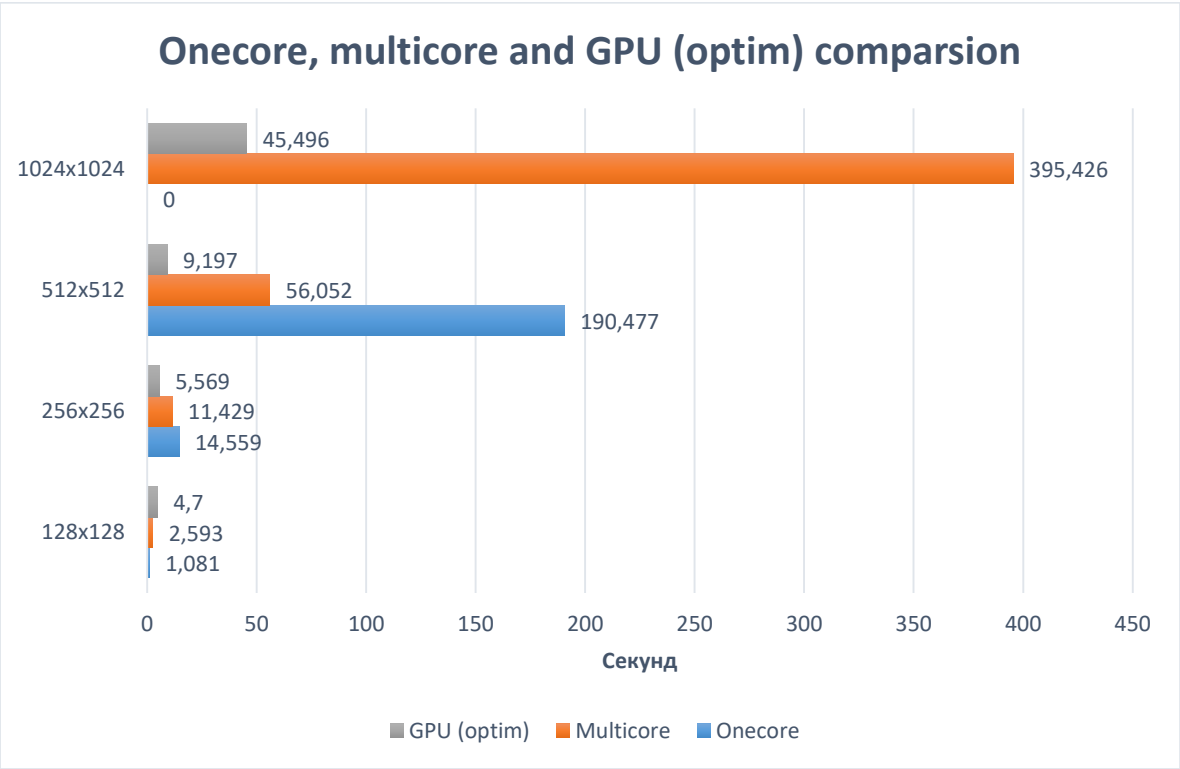
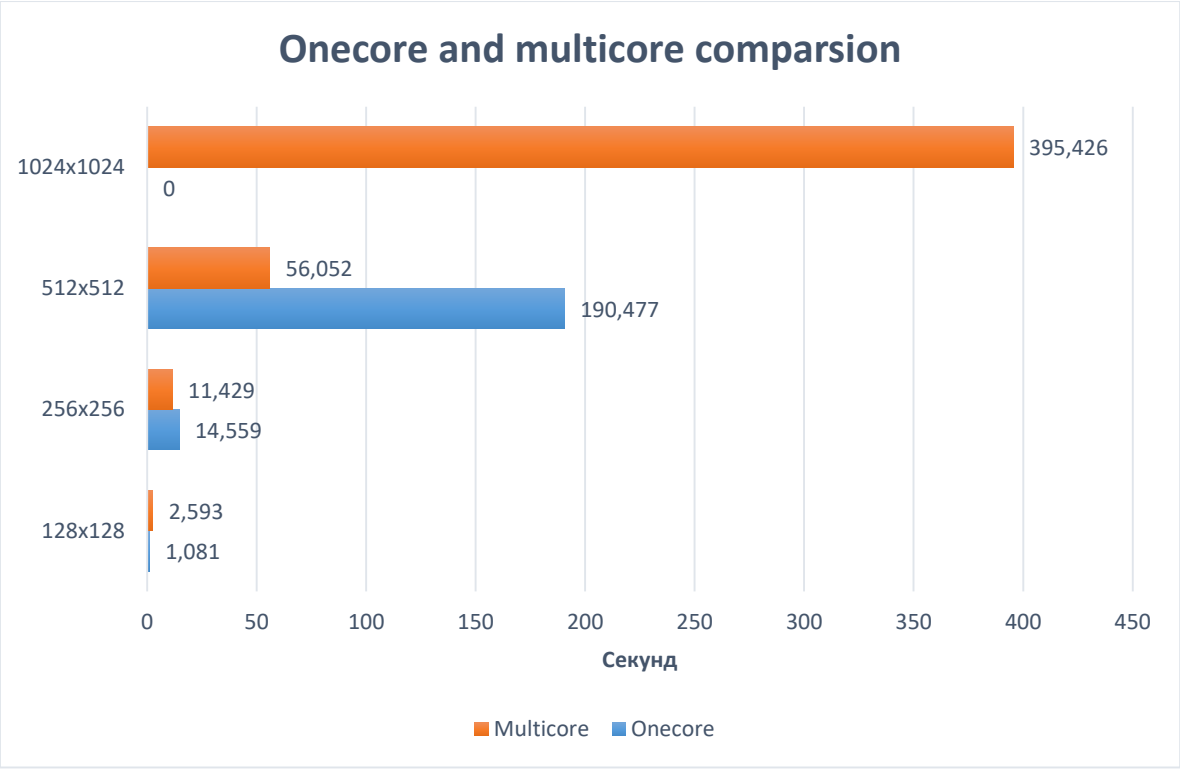
Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	1.081	9.9998e-07	30074
256x256	14.559	9.9993e-07	102885
512x512	190.477	9.99984e-07	339599
1024x1024	2285.354	1.36929e-06	1000000

## CPU-multicore

Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	2.593	9.9998e-07	30074
256x256	11.429	9.9993e-07	102885
512x512	56.052	9.99984e-07	339599
1024x1024	395.426	1.36929e-06	1000000

## GPU-optimized

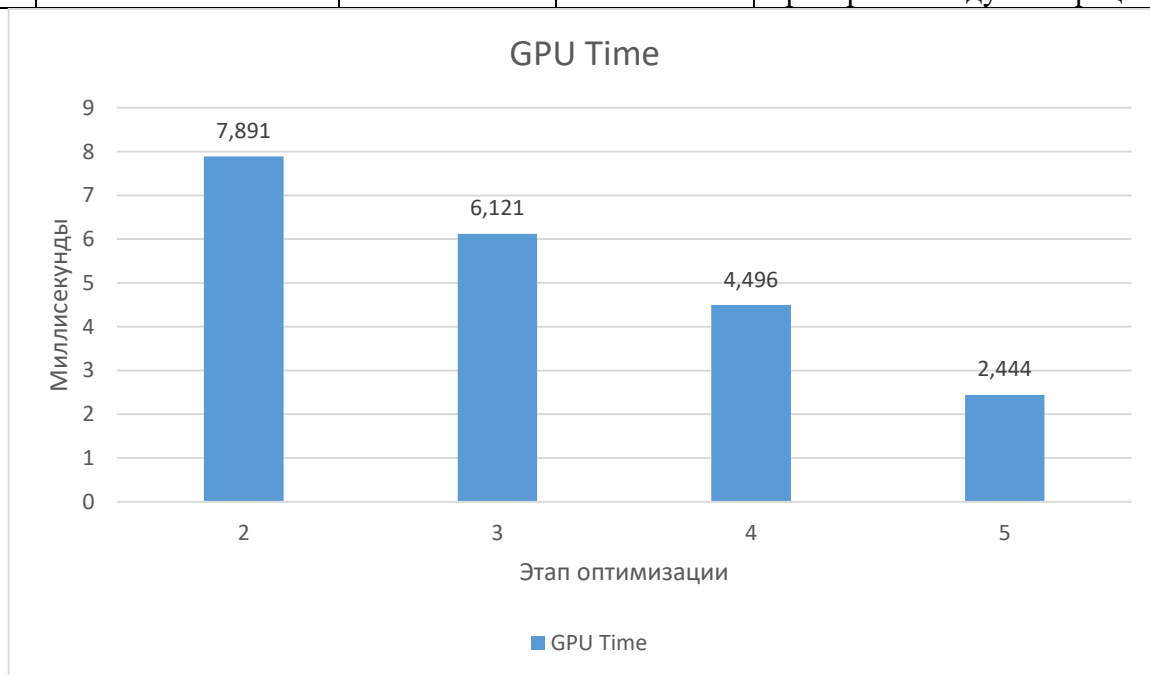
Размер сетки	Время выполнения (сек)	Ошибка	Количество итераций
128x128	4.700	9.98757e-07	30078
256x256	5.569	9.99703e-07	102888
512x512	9.197	9.99965e-07	339600
1024x1024	45.496	1.36929e-06	1000000



# Этапы оптимизации кода на GPU

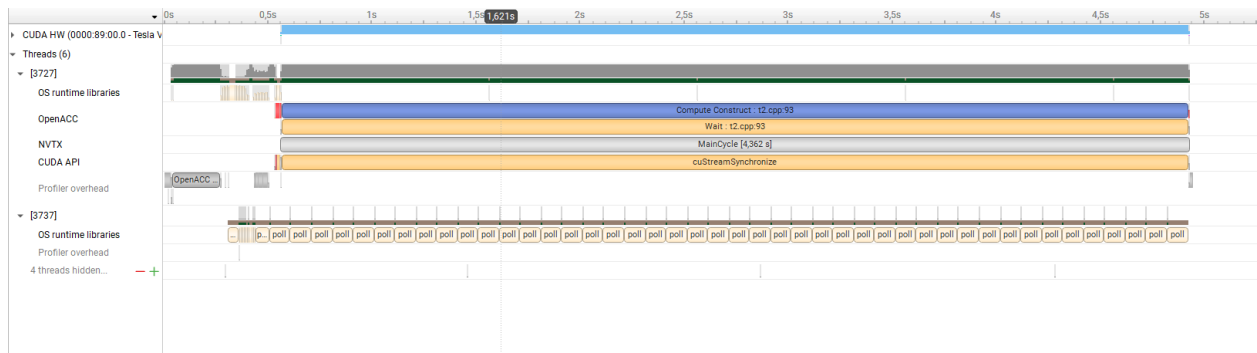
## Этапы оптимизации на сетке 512x512

№	Время выполнения основного цикла	Ошибка	Кол-во итераций	Комментарий
1	4.362 с	0.107043	100	Попытка распараллелить весь цикл while (не получилось)
2	7.891 мс	0.107043	100	Распараллеливание только внутреннего цикла
3	6.121 мс	0.107043	100	Reduction на error, ручное управление памятью
4	4.496 мс	0.107043	100	Замена копирования DtoD на swar указателей внутри GPU, а также асинхронный запуск внутреннего цикла
5	2.444 мс	0.107043	100	Обновление error на хосте каждые 5 итераций (т.к. ошибка постоянно уменьшается, нет смысла проверять каждую итерацию)

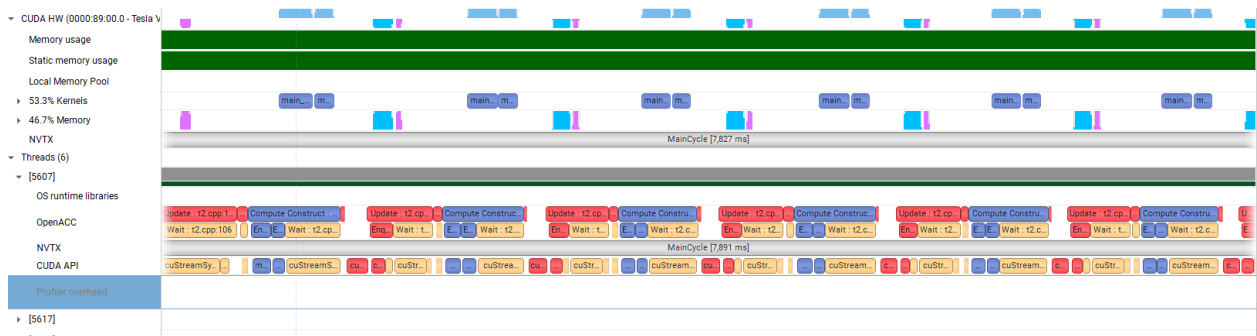


Возможной оптимизацией кода на GPU еще может быть уменьшение количества обменов данных между CPU и GPU. Т.е. делать проверку на ошибки и количества итераций на GPU, тем самым мы уменьшим количество синхронизаций до минимального количества.

## Первый этап:

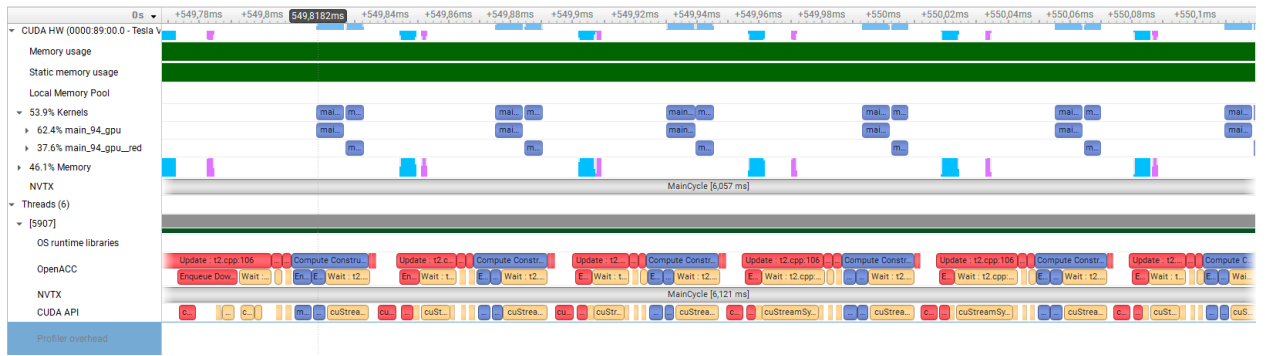


## Второй этап:

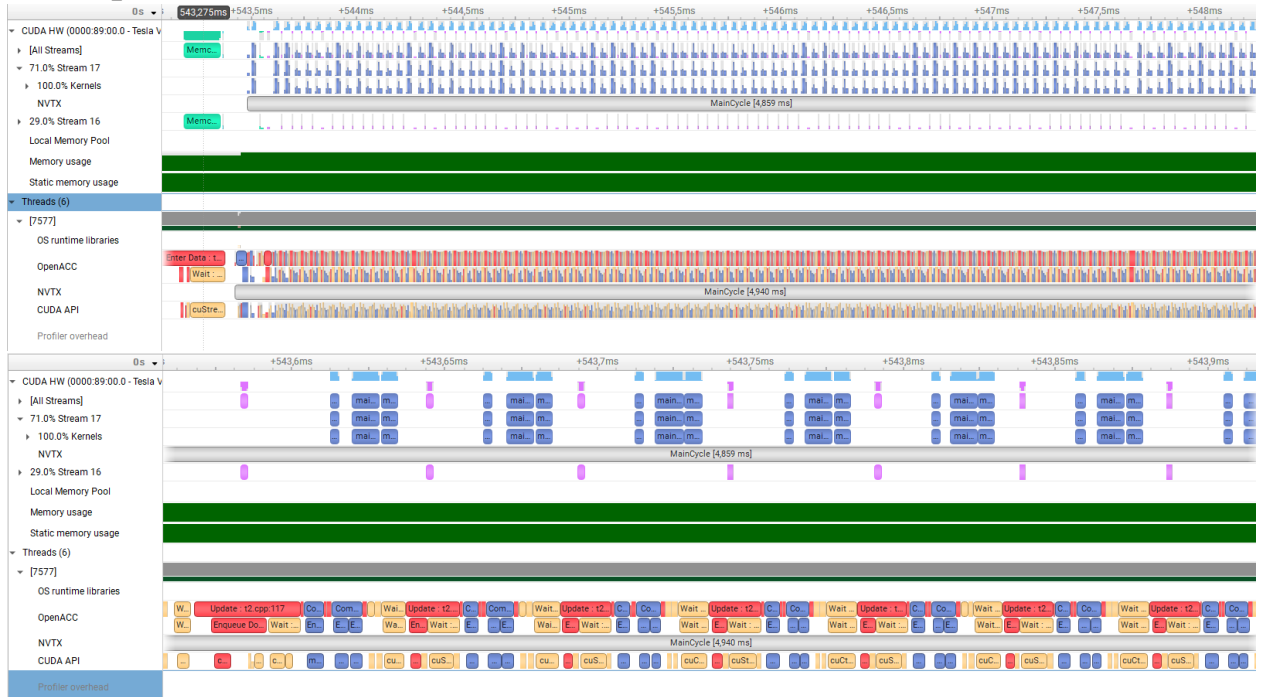


### Третий этап:





## Четвертый этап



## Пятый этап:



## Вывод:

Для обработки сетки небольшого размера будет целесообразнее использовать однопоточный вариант, т.к. издержки на инициализацию видеокарты занимают время. Но при увеличении размера сетки видно, что распараллеливание задачи более эффективно, чем выполнение в одном потоке. Т.к. среднее время выполнения одного цикла на GPU, сильно меньше, чем на CPU.



Github: <https://github.com/JooudDoo/Parallelism-Tasks>

