



Вычисления на графических ускорителях в экосистеме RISC-V

Семён Григорьев

Санкт-Петербургский Государственный Университет

29 сентября 2026г.

Семён Григорьев

- Доцент кафедры системного программирования Санкт-Петербургского Государственного Университета
- Руководитель исследовательской группы в лаборатории YADRO в СПбГУ
 - ▶ Топ-ИТ: Полина Угарова, Екатерина Арзамасцева, Даниил Правкин, Глеб Запорожец,
 - ▶ Шамиль Шаехмитов, Владимир Кутуев, ...
- Области интересов
 - ▶ **Высокопроизводительная линейная алгебра** для анализа графов
 - ★ **Обобщённая**: матрицы и вектора параметризованы типом элемента, операции над ними могут быть заданы пользователем
 - ★ **Разреженная**: специализированные структуры для хранения матриц и векторов, специализированные алгоритмы для их обработки
 - ★ В том числе, с использованием **графических ускорителей**



- Email: s.v.grigoriev@mail.spbu.ru
- GitHub: [gsvgit](#)
- Google Scholar: [Semyon Grigorev](#)
- DBLP: [Semyon V. Grigorev](#)

Контекст: вычисления, а не графика

- «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
 - ▶ Классические оптимизации
 - ▶ Удобно анализировать «отзывчивость» GPGPU при классические техники

¹<https://github.com/CNugteren/CLBlast>

²<https://github.com/SparseLinearAlgebra/spla>

³<https://graphblas.org/>

Контекст: вычисления, а не графика

- «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
 - ▶ Классические оптимизации
 - ▶ Удобно анализировать «отзывчивость» GPGPU при классические техники
- **CLBlast**¹: вычислительная линейная алгебра (плотная)
 - ▶ Переносимое решение между GPGPU разных вендоров (за счёт OpenCL)
 - ▶ Оптимизированные, в том числе под мобильные GPGPU, ядра
 - ▶ Неплохой тест на поддержку классических оптимизаций

¹<https://github.com/CNugteren/CLBlast>

²<https://github.com/SparseLinearAlgebra/spla>

³<https://graphblas.org/>

Контекст: вычисления, а не графика

- «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
 - ▶ Классические оптимизации
 - ▶ Удобно анализировать «отзывчивость» GPGPU при классические техники
- **CLBlast**¹: вычислительная линейная алгебра (плотная)
 - ▶ Переносимое решение между GPGPU разных вендоров (за счёт OpenCL)
 - ▶ Оптимизированные, в том числе под мобильные GPGPU, ядра
 - ▶ Неплохой тест на поддержку классических оптимизаций
- **Spla**²: *алгоритмы анализа графов на основе разреженной обобщённой линейной алгебры*, реализация подмножества GraphBLAS³-подобного API на C++ и OpenCL C
 - ▶ Переносимое решение между GPGPU разных вендоров
 - ▶ Запускается на CPU благодаря POCL
 - ▶ На «взрослых» GPGPU конкурентно с аналогами
 - ▶ Неплохой тест на «зрелость» поддержки OpenCL на устройстве

¹<https://github.com/CNugteren/CLBlast>

²<https://github.com/SparseLinearAlgebra/spla>

³<https://graphblas.org/>

Контекст: вычисления, а не графика

- «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
 - ▶ Классические оптимизации
 - ▶ Удобно анализировать «отзывчивость» GPGPU при классические техники
- **CLBlast**¹: вычислительная линейная алгебра (плотная)
 - ▶ Переносимое решение между GPGPU разных вендоров (за счёт OpenCL)
 - ▶ Оптимизированные, в том числе под мобильные GPGPU, ядра
 - ▶ Неплохой тест на поддержку классических оптимизаций
- **Spla**²: *алгоритмы анализа графов на основе разреженной обобщённой линейной алгебры*, реализация подмножества GraphBLAS³-подобного API на C++ и OpenCL C
 - ▶ Переносимое решение между GPGPU разных вендоров
 - ▶ Запускается на CPU благодаря POCL
 - ▶ На «взрослых» GPGPU конкурентно с аналогами
 - ▶ Неплохой тест на «зрелость» поддержки OpenCL на устройстве

Работоспособность, не борьба за производительность

¹<https://github.com/CNugteren/CLBlast>

²<https://github.com/SparseLinearAlgebra/spla>

³<https://graphblas.org/>

- **RISC-V CPU + Imagination Technologies GPU**: самая распространённая (из доступных) конфигурация

- **RISC-V CPU + Imagination Technologies GPU:** самая распространённая (из доступных) конфигурация
- **RISC-V CPU + AMD GPU:** работает и есть официальная поддержка ([Milk-V Pioneer](#), [Milk-V Megrez](#), [RuyiBook](#))
- **RISC-V CPU + Intel GPU:** [ходят слухи, что можно](#), но официальной поддержки пока нет
- **RISC-V CPU + Nvidia GPU:** [анонсировано](#)

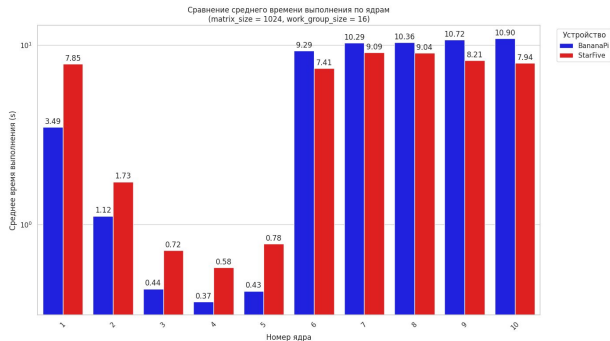
- **RISC-V CPU + Imagination Technologies GPU:** самая распространённая (из доступных) конфигурация
- **RISC-V CPU + AMD GPU:** работает и есть официальная поддержка ([Milk-V Pioneer](#), [Milk-V Megrez](#), [RuyiBook](#))
- **RISC-V CPU + Intel GPU:** [ходят слухи, что можно](#), но официальной поддержки пока нет
- **RISC-V CPU + Nvidia GPU:** [анонсировано](#)
- **RISC-V GPU** (открытые проекты с полным стеком)
 - ▶ [Vortex](#)
 - ▶ [Ventus](#)

- OpenCL: почти всё работает
 - ▶ Spla запускается и работает
 - ▶ CLBlast запускается и работает
 - ▶ Вполне ожидаемо отзывается на многие типичные оптимизации

- OpenCL: почти всё работает
 - ▶ Spla запускается и работает
 - ▶ CLBlast запускается и работает
 - ▶ Вполне ожидаемо отзывается на многие типичные оптимизации
- НО
 - ▶ Флаги доступа к памяти не совсем по стандарту
 - ▶ В Spla странные поведения при использовании приватной памяти (пока не наблюдали на других устройствах)

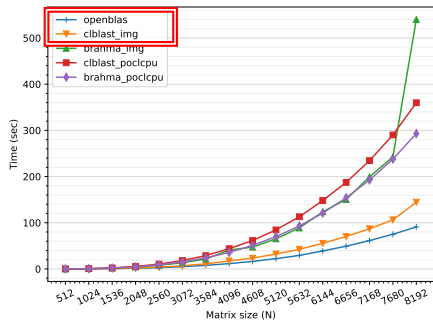
Умножение плотных матриц на IMG GPU

IMG BXE-2-32 (Banana Pi) и IMG BXE-4-32 (StarFive)



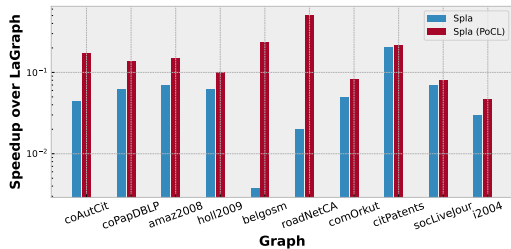
- Ожидается реакция на ряд оптимизаций
- Сложно конкурировать с ЦПУ
- Надо смотреть на более свежем «железе»

IMG BXE-2-32

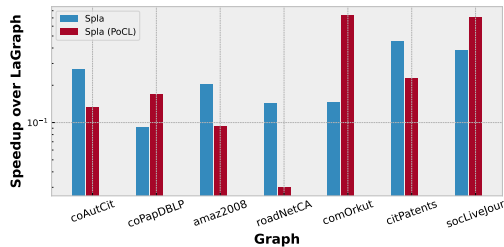


Spla на SpacemiT M1 (RISC-V) с IMG BXE-2-32 GPU

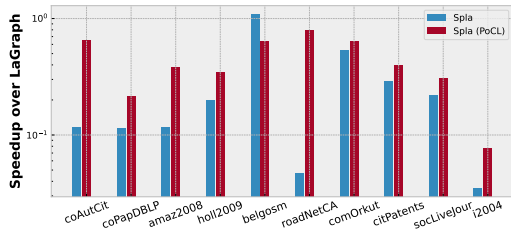
BFS



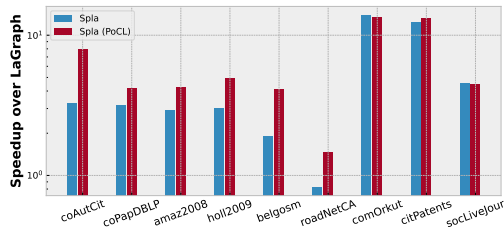
Triangle Count



Single Source Shortest Path



PageRank



Запуск GPGPU на Milk-V (SpacemiT M1)

✓ Старый AMD (Radeon HD 6450)

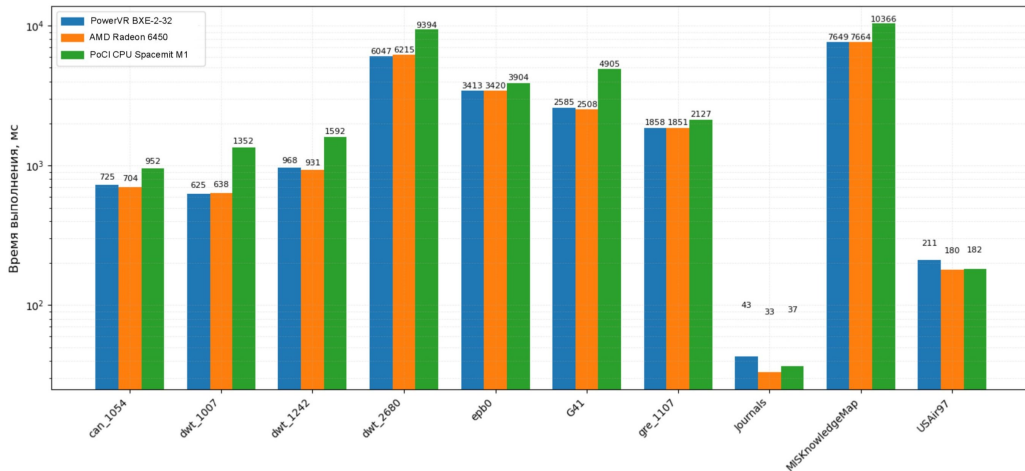
- ▶ Запускается «из коробки»
- ▶ Работают прикладные алгоритмы на основе Spla

⚙ Должно быть возможно, но нужно конфигурировать и собирать ядро

- ▶ Новый AMD (RX 6750)
 - ★ Работа идёт: <https://www.phoronix.com/news/AMDKFD-RISC-V-Linux-6.16>
 - ★ Команда Yadro смогла подключить AMD GPGPU к RISC-V ядру на ПЛИС:
<https://habr.com/ru/companies/yadro/articles/946950/>
- ▶ Intel ARC
 - ★ Работа идёт: <https://habr.com/ru/companies/selectel/articles/966828/>
- ▶ Nvidia, скорее всего, через *nouveau* [nuvo]

Boruvka MST (Spla) на Milk-V + AMD Radeon HD 6450

Сравнение производительности Spla на трёх платформах

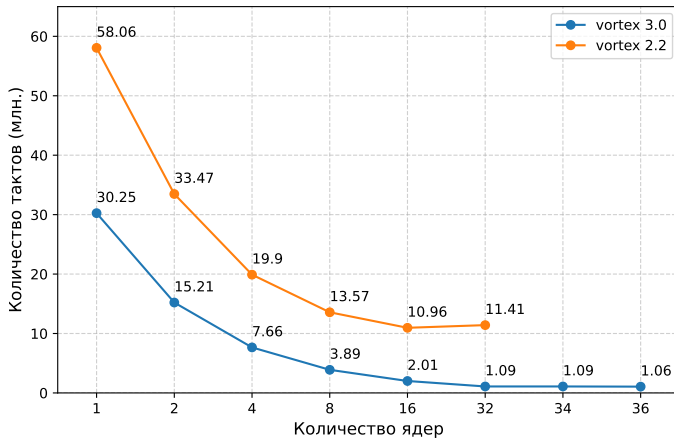


- Georgia Tech, UCLA
- Набор инструкций, основанный на RISC-V ISA
- Поддержка OpenCL через POCL (LLVM)
- Полностью открытая реализация полного стека
 - ▶ RTL-дизайн (System Verilog)
 - ▶ Компиляторный стек (POCL, LLVM)
 - ▶ Компоненты времени исполнения
 - ▶ Стек разработчика (симуляторы и т.д.)
- Для ПЛИС с HBM
 - ? Бонус для обработки слабоструктурированных данных
- Недавно вышел релиз v3.0:
 - ▶ Обновили все компоненты: RTL, рантайм, инструменты разработчика, ...
 - ▶ Ключевые (для нас) обновления
 - ★ Улучшена поддержка OpenCL, поддержка атомиков в компиляторе и рантайме
 - ★ Атомики в RTL
 - ★ Обновление до LLVM 20 и POCL 7.0



Надо обновлять и всё перепроверять

Наивное умножение матриц на Vortex (в simx)



Конфигурация: clusters=1, warps=8, threads=64, l1cache, l2cache, l3cache

Проблемы со сбросом регистров

- Регистры сбрасываются в глобальную память

Проблемы со сбросом регистров

- Регистры сбрасываются в глобальную память
- Были выявлены нами в версии v2.0
 - ▶ Issue 1
 - ▶ Issue 2
 - ▶ Были закрыты разработчиками как исправленные
- Недавно открыт новый [issue](#) со ссылками на наши и описанием всё той же проблемы

Проблемы со сбросом регистров

- Регистры сбрасываются в глобальную память
- Были выявлены нами в версии v2.0
 - ▶ Issue 1
 - ▶ Issue 2
 - ▶ Были закрыты разработчиками как исправленные
- Недавно открыт новый [issue](#) со ссылками на наши и описанием всё той же проблемы
- Типичные оптимизации не работают

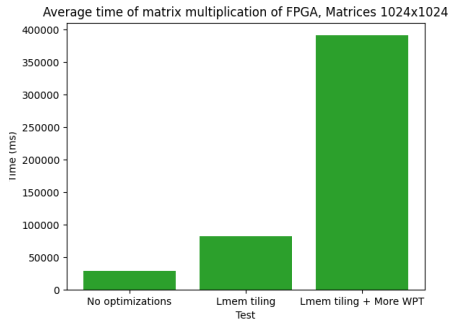
Эксперименты с оптимизациями умножения плотных матриц

- По мотивам «[Tutorial: OpenCL SGEMM tuning for Kepler](#)» от Cedric Nugteren
- Подгрузка блоков в локальную память и более агрессивное использование регистров

Эксперименты с оптимизациями умножения плотных матриц

- По мотивам «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
- Подгрузка блоков в локальную память и более агрессивное использование регистров

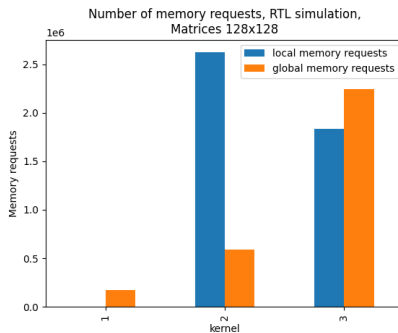
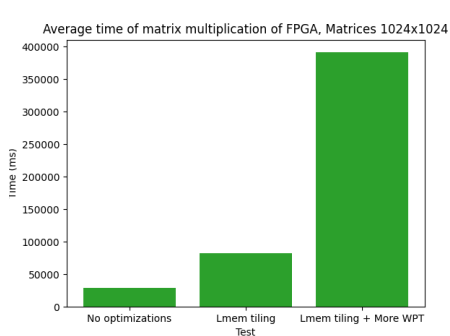
Для Vortex v2



Эксперименты с оптимизациями умножения плотных матриц

- По мотивам «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
- Подгрузка блоков в локальную память и более агрессивное использование регистров

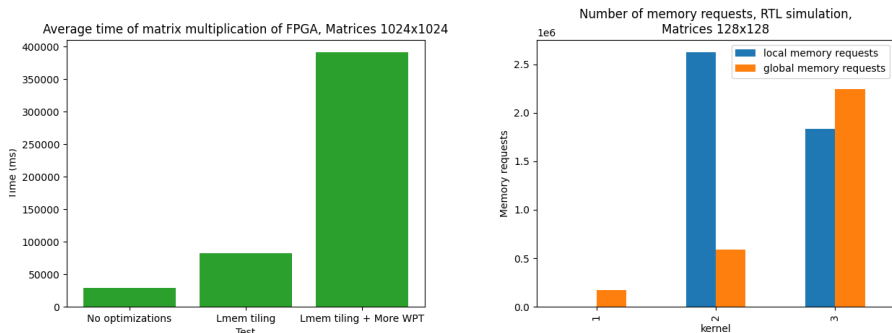
Для Vortex v2



Эксперименты с оптимизациями умножения плотных матриц

- По мотивам «Tutorial: OpenCL SGEMM tuning for Kepler» от Cedric Nugteren
- Подгрузка блоков в локальную память и более агрессивное использование регистров

Для Vortex v2



- Мы не заметили изменения поведения на Vortex v3.0
- Продолжаем расследование

- ✓ Проходят тесты и работают алгоритмы анализа графов в функциональном симуляторе `simx`
 - ▶ BFS, SSSP, PageRank, Triangle Count
- ✓ Проходят тесты, в том числе на алгоритмы анализа графов, в RTL-симуляторе (на свежем `main-e`, `v3.0+`)
- ⚙ Исследуем масштабируемость и влияние параметров (конфигурация ядер, размеры кэшей) на производительность алгоритмов

- DSP Lab of Tsinghua University
- Набор инструкций, основанный на RISC-V ISA (32-битный)
- Поддержка OpenCL через POCL
- Полностью открытая реализация полного стека
 - ▶ RTL-дизайн (Chisel HDL)
 - ▶ Компиляторный стек (POCL, LLVM)
 - ▶ Компоненты времени исполнения
 - ▶ Стек разработчика (симуляторы и т.д.)
- Относительно активно развивается

- DSP Lab of Tsinghua University
 - Набор инструкций, основанный на RISC-V ISA (32-битный)
 - Поддержка OpenCL через POCL
 - Полностью открытая реализация полного стека
 - ▶ RTL-дизайн (Chisel HDL)
 - ▶ Компиляторный стек (POCL, LLVM)
 - ▶ Компоненты времени исполнения
 - ▶ Стек разработчика (симуляторы и т.д.)
 - Относительно активно развивается
-  Мы только начинаем его изучать

- Есть проблемы с поддержкой атомарных операций, но мы помогаем их преодолеть
 - ✓ Execute 32-bit AMOs as per-thread vector operations (spike)
 - ⚙ Align by-value kernel arguments like the Ventus POCL driver (compiler, llvm)
 - ⚙ Use signed 12-bit immediate for vadd12.vi/vsub12.vi (compiler, llvm)
 - ⚙ Keep materialized frame address in eliminateFrameIndex (compiler, llvm)

- Есть проблемы с поддержкой атомарных операций, но мы помогаем их преодолеть
 - ✓ Execute 32-bit AMOs as per-thread vector operations (spike)
 - ⚙ Align by-value kernel arguments like the Ventus POCL driver (compiler, llvm)
 - ⚙ Use signed 12-bit immediate for vadd12.vi/vsub12.vi (compiler, llvm)
 - ⚙ Keep materialized frame address in eliminateFrameIndex (compiler, llvm)
- Если собрать всё в кучу, то Spla работает в симуляторе (spike)

Заключение

- ✓ RISC-V CPU + IMG GPGPU работает⁴, но пока не для высокопроизводительных вычислений
- ⚙ RISC-V CPU + AMD, Intel, Nvidia GPGPU активно развивается

⁴Почти

⁵По большей части. Vortex работает над поддержкой Cuda

Заключение

- ✓ RISC-V CPU + IMG GPGPU работает⁴, но пока не для высокопроизводительных вычислений
- ⚙ RISC-V CPU + AMD, Intel, Nvidia GPGPU активно развивается
- ⚙ Открытые архитектуры GPGPU на базе RISC-V ISA активно развиваются
 - ▶ Vortex, Ventus
 - ▶ Не только RTL-дизайн но полный стек
 - ▶ Вокруг OpenCL-стека⁵

⁴Почти

⁵По большей части. Vortex работает над поддержкой Cuda

- ✓ RISC-V CPU + IMG GPGPU работает⁴, но пока не для высокопроизводительных вычислений
- ⚙ RISC-V CPU + AMD, Intel, Nvidia GPGPU активно развивается
- ⚙ Открытые архитектуры GPGPU на базе RISC-V ISA активно развиваются
 - ▶ Vortex, Ventus
 - ▶ Не только RTL-дизайн но полный стек
 - ▶ Вокруг OpenCL-стека⁵

**Пока боремся за стабильность
Но уже можно ставить интересные эксперименты**

⁴Почти

⁵По большей части. Vortex работает над поддержкой Cuda