

Оптимизация xxHash для RISC-V с использованием различных реализаций RVV

Николай Пономарев

Математико-механический факультет СПбГУ

20 сентября 2024 г.

xxHash

xxHash — современная библиотека для быстрого хеширования

Особенности:

- Поддерживает вычисление 32-, 64- и 128-битных хешей
- Скорость работы может превышать пропускную способность RAM
- Имеет поддержку SIMD и векторных расширений процессоров
 - SSE2, AVX2, AVX512
 - NEON, SVE
 - Но нет поддержки RVV
- Использование intrinsic функций для оптимизации

Векторное расширение RISC-V

- Векторное расширение RISC-V \equiv RVV
- В природе встречается три версии:
 - Урезанный RVV 0.7.1 — реализован в ядрах Xuantie C906 (Sipeed Lichee RV, MangoPi MQ), **НЕ** поддерживает 64-битные элементы
 - Полноценный RVV 0.7.1 — в ядрах Xuantie C910 (Sipeed Lichee Pi 4A)
 - RVV 1.0 — в ядрах Xuantie C908 и C920, SpacemiT X60 (Banana Pi BPI-F3)
- Программные инструменты не готовы поддерживать RVV 0.7.1
 - Нужно использовать патченные производителем инструменты

Урезанный RVV 0.7.1

- Испытуемый: Sipeed Lichee RV с урезанным RVV 0.7.1 и VLEN = 128 бит
- **Проблема:** Алгоритм работает над 64-битными числами
 - ⇒ будем использовать 32-битные элементы
- **Проблема:** Битовые операциям всё равно, но что со сложением?
 - ⇒ будем дополнительно обрабатывать перенос

Полноценный RVV 0.7.1 (осень 2023)

- Испытуемый: Sipeed LicheePi 4A с полноценным RVV 0.7.1 и VLEN=128
- Здесь хочется объединить реализации для RVV 0.7.1 для RVV 1.0
- Это возможно, **но**
 - в апстриме компиляторов у intrinsic функций появился префикс `__riscv`
⇒ придётся повозиться с макросами
 - можно попасться на несуществующую в одной версии инструкцию
- А ещё в GCC 14 есть поддержка XTheadVector, но со сломанной кодогенерацией 😞

RVV 1.0!

- BPI-F3 с RVV 1.0 и VLEN = 256 бит
- Кажется, что счастье!
 - Полная поддержка в апстриме компиляторов
 - Автовекторизация

Бенчмарки

	Размер регистра, бит	Скалярная версия, Мб/с	Векторная версия, Мб/с	Ускорение, раз
Урезанный RVV 0.7.1	128	169.3	116.0	0.69
Полноценный RVV 0.7.1	128	645.3	472.6	0.73
RVV 1.0	256	516.3	2036.0	3.94
AVX2	256	21374.8	46864.3	2.19

Выводы

- Оформлен Pull Request в библиотеку
 - ссылка

Спасибо!