**Ключи компиляции GCC:**

Таблица 5.5 - Опции компиляции GNU GCC C/C++

|  |  |
| --- | --- |
| Скалярная версия | -O3 -ffast-math -fivopts -march=native -fno-tree-vectorize |
| Автовекторизованная версия | -O3 -ffast-math -fivopts -march=native -fopt-info-vec -fopt-info-vec-missed |
| Векторизованная версия | -O3 -ffast-math -fivopts -march=native -fopt-info-vec -fopt-info-vec-missed -fno-tree-vectorize |

**-Wall** - Вывод сообщений о всех предупреждениях или ошибках, возникающих во время компиляции программы.

**-O** - включить оптимизацию

**-O0 -** Этот уровень (буква "O" и ноль за ней) отключает оптимизацию полностью и является уровнем по умолчанию, если никакого уровня с префиксом -O не указано в переменных CFLAGS или CXXFLAGS. Это сокращает время компиляции и может улучшить данные для отладки, но некоторые приложения не будут работать должным образом без оптимизации. Эта опция не рекомендуется, за исключением использования в целях отладки.

**-O1** - Это наиболее простой уровень оптимизации. Компилятор попытается сгенерировать быстрый, занимающий меньше объема код, без затрачивания наибольшего времени компиляции. Он достаточно простой, но должен всегда выполнять свою работу.

**-O2** - Шаг вперед от -O1. Рекомендуемый уровень оптимизации, до тех пор, пока не понадобится что-то особенное. -O2 активирует несколько дополнительных флагов вдобавок к флагам, активированных -O1. С параметром -O2, компилятор попытается увеличить производительность кода без нарушения размера, и без затрачивания большого количества времени компиляции.

**-O3** - Это наибольший возможный уровень оптимизации. Включает оптимизации, являющейся дорогостоящей с точки зрения времени компиляции и потребления памяти. Компиляция с -O3 не является гарантированным способом повышения производительности, и на самом деле во многих случаях может привести к замедлению системы из-за больших двоичных файлов и увеличения потребления памяти. -O3 известен также тем, что ломает несколько пакетов. Использование -O3 не рекомендуется.

**-mavx** – включение поддержки инструкций AVX

**-ffast-math** - Этот параметр не включается никаким опцией -O, кроме -Ofast, поскольку это может привести к некорректному выходу для программ, которые зависят от точной реализации правил / спецификаций IEEE или ISO для математических функций. Однако он может дать более быстрый код для программ, для которых не требуются гарантии этих спецификаций

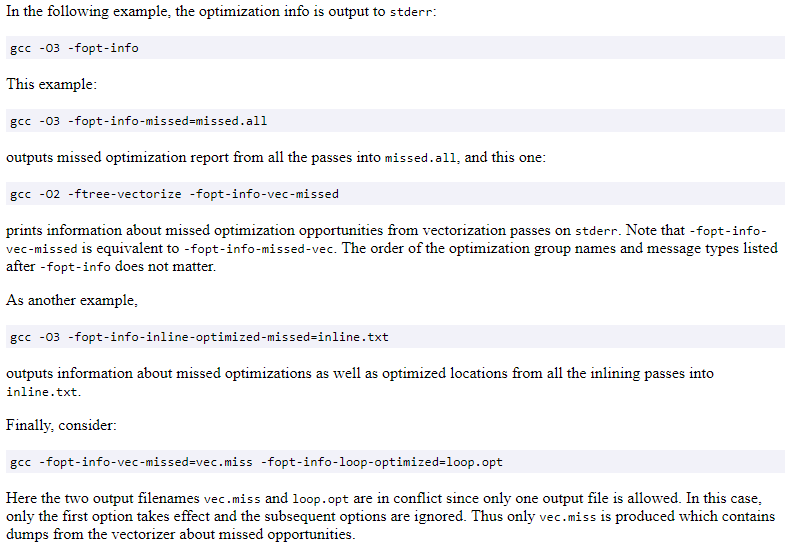
**-fivopts** - влияют только на компиляцию, а не на связь, но я уверен, что они ничего не делают, если вы не включите оптимизацию, по крайней мере, с -O. Вам, вероятно, не нужно их использовать, просто используйте -O или -O2 или -O3 вместо того, чтобы настраивать индивидуальные или более конкретные параметры, если вы точно не знаете, чего вы пытаетесь достичь, и измеряете точные эффекты.

**-march=native** - Самым первым и наиболее важным параметром является -march. Он сообщает компилятору, какой код генерировать для архитектуры процессора (соответствующие английские термины: “architecture” или “arch”. — Прим. пер.); он сообщает GCC компилятору, что тот должен генерировать код для определенного типа CPU. Разные типы CPU имеют разные возможности, поддерживают различные наборы команд и обладают разными способами исполнения кода. Флаг -march проинструктирует компилятор генерировать специфичный код для архитектуры CPU, со всеми доступными возможностями, особенностями, наборами команд, интересными функциями и так далее. Если тип процессора все еще нее определен, либо не знаете какую настройку выбрать, то можно воспользоваться параметром -march=native. Когда используется этот флаг, GCC попытается распознать процессор и автоматически установит для него подходящие флаги.

**-fopt-info-vec-missed** - Печатает информацию о пропущенных возможностях оптимизации от векторизации на stderr. Обратите внимание, что -fopt-info-vec-missed эквивалентен -fopt-info-missed-vec. Порядок имен групп оптимизации и типов сообщений, перечисленных после -fopt-info, не имеет значения.

**-fopt-info-vec** - Печатает информацию проделанной векторизации на stderr.

-**fno-tree-vectorize** – Отключает векторизацию.



**-c** - создать только объектный файл (source.o) из исходного (source.f, .source.for)

**-o** - создать загрузочный файл с именем file (по умолчанию создается файл с именем a.out)

**Ключи компиляции ICC:**

Таблица 5.6 - Опции компиляции Intel C++ compiler

|  |  |
| --- | --- |
| Скалярная версия | -O3 -xHost -no-vec |
| Автовекторизованная версия | -O3 -xHost -qopt-report3 -qopt-report-phase=vec,loop -qopt-report-embed |
| Векторизованная версия | -O3 -xHost -qopt-report3 -qopt-report-phase=vec,loop -qopt-report-embed |

**-xHost** – флаг «-xHOST» означает оптимизацию для того процессора, на котором запущен компилятор.

**-qopt** – cообщает компилятору генерировать отчет об оптимизации. Опция -qopt-report - это опция замены для -opt-report, которая устарела

**-qopt-report3** – (Необязательно) n = 3, указывает уровень детализации в отчете. Вы можете указать значения от 0 до 5. Если вы укажете ноль, отчет не будет сгенерирован. Для уровней n = 1 - n = 5 каждый уровень включает всю информацию предыдущего уровня, а также потенциально некоторую дополнительную информацию. Уровень 5 дает наибольший уровень детализации. Если вы не укажете n, по умолчанию используется уровень 2, который создает средний уровень детализации.

**-qopt-report-phase=vec,loop** -

**-qopt-report-embed** – Определяет, будут ли внедрены специальные аннотации информации цикла в файл объекта и / или файл сборки при его создании.

**Ключи компиляции LLVM/Clang:**

Таблица 5.7 - Опции компиляции LLVM/Clang

|  |  |
| --- | --- |
| Скалярная версия | -O3 -ffast-math -fno-vectorize |
| Автовекторизованная версия | -O3 -ffast-math -fvectorize -Rpass=loop-vectorize -Rpass-missed=loop-vectorize -Rpass-analysis=loop-vectorize |
| Векторизованная версия | -O3 -ffast-math -fvectorize -Rpass=loop-vectorize -Rpass-missed=loop-vectorize -Rpass-analysis=loop-vectorize |

**-Rpass=loop-vectorize** - идентифицирует циклы, которые были успешно векторизованы.

**-Rpass-missed=loop-vectorize** – идентифицирует циклы, которые не были успешно векторизованы.

**-Rpass-analysis=loop-vectorize** – определяет из-за чего он не смог векторизовать тот или иной цикл