09.11.2013

Супер-скалярные микропроцессоры

До 86 года все процессоры были скалярными. С обычным конвейером, и с тремя устройствами вещественное АЛУ, целочисленное, то, что работает с цифрами (видимо предиктор). Но проблема в том, что всегда эти устройства работают не все, а лишь одно из них.

Предлагается ввести параллелизм на уровне команд. Чтобы, например, можно было считать одновременное вещественную и целочисленную инструкцию. (Конвейер на некотором этапе раздваивается) Поток команд один – тут ничего не изменилось, просто дорога конвейера раздвоилась. Несовместимые команды (если есть зависимости) нужно отправлять на главный конвейер.

Конвейер можно не только раздваивать, но и расстраивать, ...

Отдельно на каждом «подковейере» уже RISC.

Табло – там лежит информация о декодированных командах. При разветвлении происходит выбор, что и куда. Табло большим быть не должно, чтобы скорость не падала.

Чтобы не тратить время на сортировку, предлагается делать это при трансляции программы.

Если машина с сиcтемой длинных команд (VLIW – very long instruction word).

Тут подразумеваются специальные компиляторы, в которых происходит перевод обычной программы в vliw команды. Для vliw – чётко всё определено, с точностью до такта (Определено время выполнения операции). Несколько функциональных устройств. Единое управляющее устройство. Каждая vliw команда имеет специальные поля, в которых можно выполнять только целочисленное, или вещественное, или загрузка в память (а дальше мы можем либо заполнить поля, либо нет). Озу должно иметь большое расслоение на банки памяти.

!!! Ну сколько можно рассматривать раритет, а????????

Нужно также помнить, что глубина конвейера для разных полей – может быть разная, а потому мы результат vliw команды должны ловить в разных местах для разных полей. (т.е. разные поля закончат выполняться через разное количество других инструкций)

Это также порождает проблему с кэшем. Потому что, если у нас половина команды есть в кэше, а другой нету, то придётся ждать, пока подгрузятся данные.

Банки памяти идентифицируются при программирования. А дальше предлагается отказаться от кэша, а данные разложить так, чтобы это было в разных банках памяти.

Vliw – умудрялась когда-то давать прирост в 12 раз (12 миллионов операций)

Почему vliw – сейчас не используются?

Потому что очень тяжело писать программы.

(думаю, что ещё иметь полноценный 2-й конвейер иметь гораздо удобнее, и потому что не часто в программе встречаются куски, где целочисленных и вещественных операций – одинаковое количество)

В 2000 году выпустили Grucoe

В процессоре была «Программная эмуляция кодов в VLIW формат».

Процессор успевает собрать команды в vliw, сколько успеет, пока не отнимут на выполнение (даже если vliw заполнена не полностью)

И хотя оптимизация происходит достаточно рыхло, всё равно класс.

Оттранслированные команды помещаются в кэш, и при доп использовании команды vliw формата поднимаются из буфера.

10% - кода выполняется 90% времени.

Потом придумали дополнительное поле шаблона, позволяющее задать зависимости оставшихся слотов, там задаётся, как можно выполнять инструкции в полях vliw команды.

Уровень вычислительной мощности весьма высок.

Была классная попытка совместить vliw и risc. Компилятор занимается оптимизацией, процессор распараллеливает.

//==================================================================================

Мультитредовые машины

При выполнении нескольких потоков, много времени занимает переключение контекста, поэтому полезно сделать аппаратную реализацию, этого дела.

Каждый тред имеет свои регистры и т.д.

На каждом такте переключение на новый тред, получается, что как бы у процессора есть много входов, у каждого входа есть всё своё – память регистры и прочее, а всё, что делает поцессор – это выполняет по очереди то, что ему подсовывают.

EPIC

1. Выявление зависимостей (поиск микропроцессов)
2. Динамическая работа между функциональными устройствами
3. Агрессивная реализация условного оператора (параллельность – дело компилятора, проводящего оптимизацию) (более высокая производительность при низких степенях сложности)
4. Эффективные способности повышения производительности (у реализации EPIC)