26.10.2013

Суперконвейерные машины – (пример – MIPS R400) когда есть машинный такт и внутренний этапный такт (из-за внутреннего этапного такта мы можем делать полторы работы за такт)

Гиперконвейерный – Pentium4 – 20 этапов в конвейере

Суперконвейер – 10 ступеней (увеличение частоты внутри этапов)

Конвейерный – 5 ступеней

Основные этапы работы ЦП: Чтение, Дешифровка, Исполнение, Запись

Каждая 7-я команда – это команда передачи управления по статистике.

Безусловный переход:

Чтобы конвейер не умирал при передачи команды управления – заводят регистр Branch Target Buff (BTB), который фактически является таблицей из 2-х записей – на какой команде перехода мы были (адрес) и куда мы при этом перешли (какой мы сделали выбор) (фактически мы собираем статистику) по таблице мы ориентируемся по теговской части (часть адреса, по которому находится команда перехода). (Параметр goto – не всегда фиксирован) Потом мы на конвейер записываем ту, куда указала таблица. Так повышается вероятность угадывания. На неудачном переходе для конвейера мы можем потерять до 30 тактов. В BTB хранится последнее направление перехода. При условном переходе мы достаём направление из BTB после чего мы грузим конвейер одной из веток, но с запретом на запись.

Условный переход:

Мы можем использовать предсказатели, куда мы перейдём. Бывают статически (угадывают с 75%) и динамические (93%).

В статическом – при создании трансляции, мы специально программируем так программу, чтобы можно было угодать (чаще всего - цикл). Обычно сравнение на равно и сравнение с 0 – предпочтительнее. Статические предсказания фиксируются следующими способами: Блок, по которому мы скорее всего будем переходить – пишется сразу после команды перехода. Сейчас даже существуют специальные флаги. Ещё способ – в некоторых системах программирования – есть специальный указатель, который указывает вероятность прохода по той или другой ветке. Если много циклов – то статический предсказатель – отличный выбор.

Динамический – это просто счётчик на каждую команду goto – угадали +1 к счётчику, не угодали -1 к счётчику. Примеры счётчиков:

Счётчик – однобитовый (В таблице BTB – появляется этот бит, после чего если 1 – ты в BTB указано правильное направление, если 0 – то в другом направлении)

Счётчик 2-х битовый – записываем 1 или 0 с вытеснением, т.е. со сдвигом (shift “>>”) Если есть хоть одна 1, то переход будет, если нету, то не будет.

Счётчик (алгоритм или схема Смита), который инвертируется, только тогда, когда произошло 2 одинаковых события подряд. Если первый элемент =1, то переход будет

1. 11 –Strongly taken
2. 10 – taken
3. 01 – weakly taken
4. 00 – strongly not taken (часто не выполняется)

Чтобы предсказание изменилось, необходимо 2 промаха

По факту – алгоритм Смита – это мы ходим по диапазону от 0 до 3, вычитая и добавляя единицу при успехе, или не успехе.

Расширение счётчика Смита, который может принимать от 0 до 2^n-1 – если первый бит 1 – то прогноз выполняется, иначе не выполняется, дальше мы делаем +1 или -1, в зависимости от того, угадали, или не угадали. За границы конечно же переходить нельзя. Корректирование на +-1 происходит, когда решение о переходе на конвейере принято.

Методом научного тыка было выяснено, что счётчик должен иметь размер 2-3 бита. (Тут кстати можно использовать не арифметику, а просто обычный автомат мура (т.е. оперировать не на уровне цифр))

Для 2-х бит 99 – 82% удачи

Размер таблицы BTB – буфер ёмкости больше 4K работает так же, как и 4K. Поэтому кажется часто пользуют буфер BTB, в котором находится 4K таблиц.

Т.к. в BTB происходит индексация по части адреса, то разные команды могут иметь один и тот же адрес в той части, которую мы смотрим и тогда у нас получится один и тот

Другой динамический способ предсказания

Регистр глобальной истории Global History Request – мы записываем в некоторый массив значения, которые нам дал прыжок да – или нет, туда записывается со сдвигом. В итоге у нас есть история ответов – да или нет. Этот регистр (наш массив) потом используется как адрес в некоторой таблице.

Регистр Local History Request – мы как в Global History Request, но на каждый тип команды перехода у нас отдельный регистр.

2-х уровневая кореллированная схема – у нас есть и local и global регистры, но мы берём из global последние несколько бит, ищём по ним нужный local, и по этой паре мы в двумерной таблице смотрим, какой выбор нам сделать.

Предиктор предикторов – когда у нас несколько предикторов, и мы выбираем при помощи ещё одного предиктора какой из них слушать

Агрессивное спикулятивное дополнение. – когда пытаются выполнить обе ветки сразу.

ISA

CISC (Complex Instruction Set Computing) – набор команд, потом появился RISC (Restricted (Reduced) Instruction Set Computing) (сокращённый набор команд) (уменьшает скорость декодирования и выполнения (т.к. команды имеют небольшую длинну))

CISC помимо арифметических и логических операций так же есть ещё и команды передачи управление. CISC нельзя было разместить на одном кристалле, поэтому и засада.

В RISC – есть только команды, для работы с регистрами и команда загрузки из памяти на регистр. Смешанных команд – нету.

Pentium Pro – это RISC машинf, но на вход поступает CISC, для совместимости создана возможность автоматического конвертирования команд из CISC в RISC.

Логические команды – булева алгебра, сдвиги, сборка и разборка (взятие по маске а потом конкатенация выбранных бит)

Косвенная адресация – это когда мы берём байт не по адресу, а по адресу, записанному по адресу. (бывает до 3-х итераций)

Для ускорения выполнения операции, в 3-х адресной команде могут быть не более 1-го операнда где указана память (ускорение чтения из памяти)

Т.к. в 3-х адресной машине могут использоваться не все операнды, то может быть применено совмещение двух операций в одну, если одной нужен один операнд, а другой 2

Существуют бесформатные команды.

SSE – Streaming SIMD Extension – (это сопроцессор) есть 128 битные регистры, если туда записать несколько чисел, а потом выполнить сложение или ещё, что, то выполниться SIMD – что приведёт к успеху по времени.

Существует много версий SSE 1-4, и в каждом появлялись какие-то дополнительные команды.

MMX (Multi Media Extension) – (это сопроцессор) около 6 регистров по 8 байтов можно делать всякие операции (сложение без переполнения) – расширение этого будет графический процессор.

Графический процессор когда работает, то он данные выбирает только локально.