For Distribution

Кэш в MIPS microAptiv UP / Microchip PIC32MZ

Зачем нужны кэши?

- В 1960-е годы процессоры были медленнее, чем память
- С 1980-х годов скорость процессоров росла быстрее, чем скорость памяти
- За одно обращение к памяти современный процессор для десктопа может выполнить сотни арифметических инструкций; без кэша он будет простаивать
- Кэш полезен уже у PIC32MX (процессор - 80 MHz, флэш – 30 MHz)
- Для 200 MHz PIC32MZ кэш становится небходим

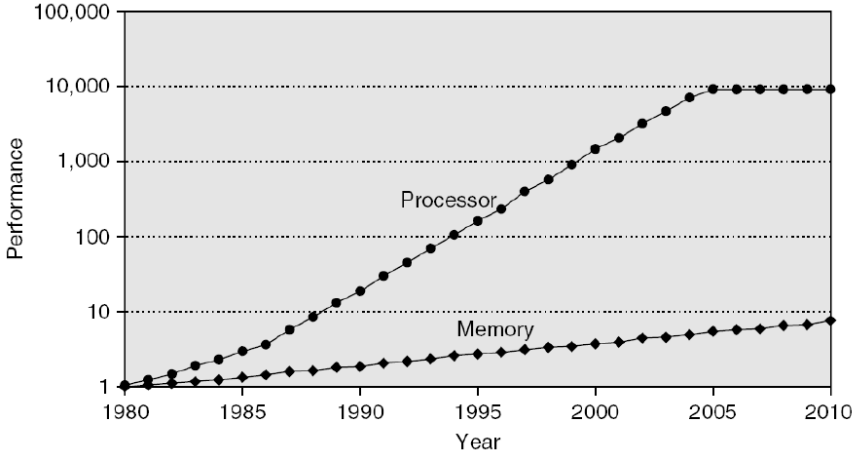
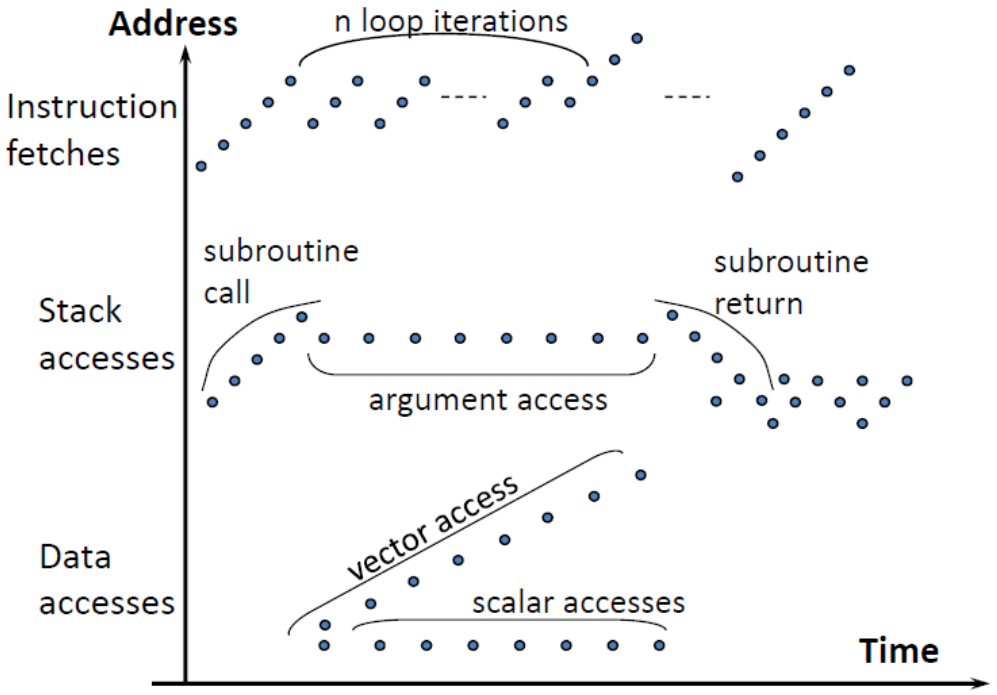


График из Hennessy & Patterson 2011,
через David Wentzlaff 2011 из Princeton University

Какие свойства программ использует кэш?



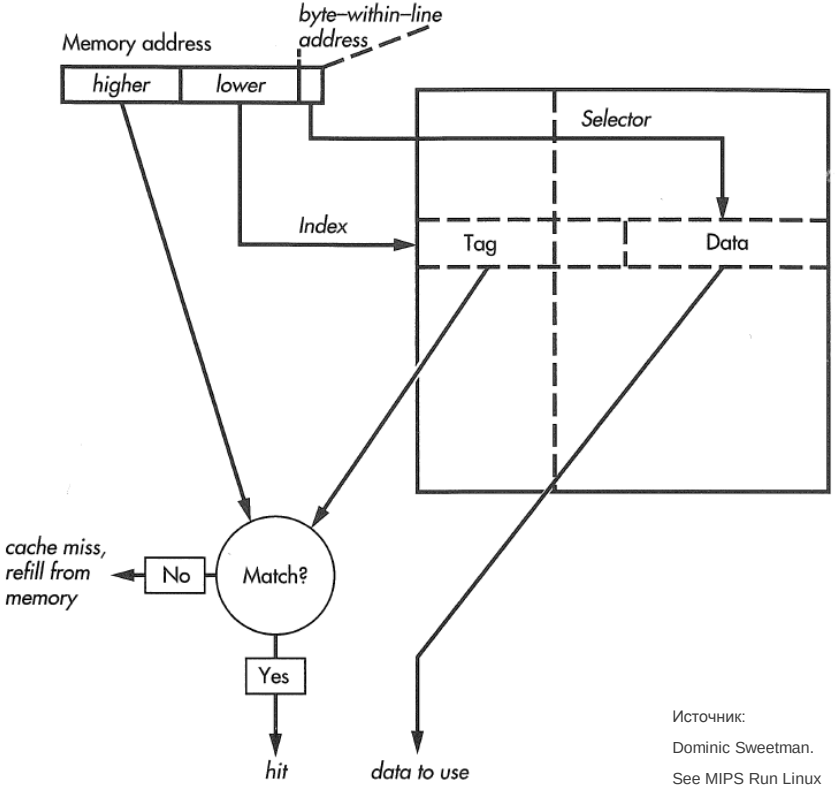
- **Временная локальность**
  - Если произошел доступ к адресу, он вероятно скоро повторится
- **Пространственная локальность**
  - Если произошел доступ к адресу, вероятно скоро будет доступ к соседнему адресу

График из David Wentzlaff 2011, Princeton University

Принцип работы кэша прямого отображения

Direct-mapped cache

- **Терминология**
  - Адрес — Address
  - Индекс — Index
  - Тэг — Tag
  - Строка — Line
  - Индекс байта — Byte index
  - Попадание — Hit
  - Промах — Miss
  - Вытеснение — Eviction



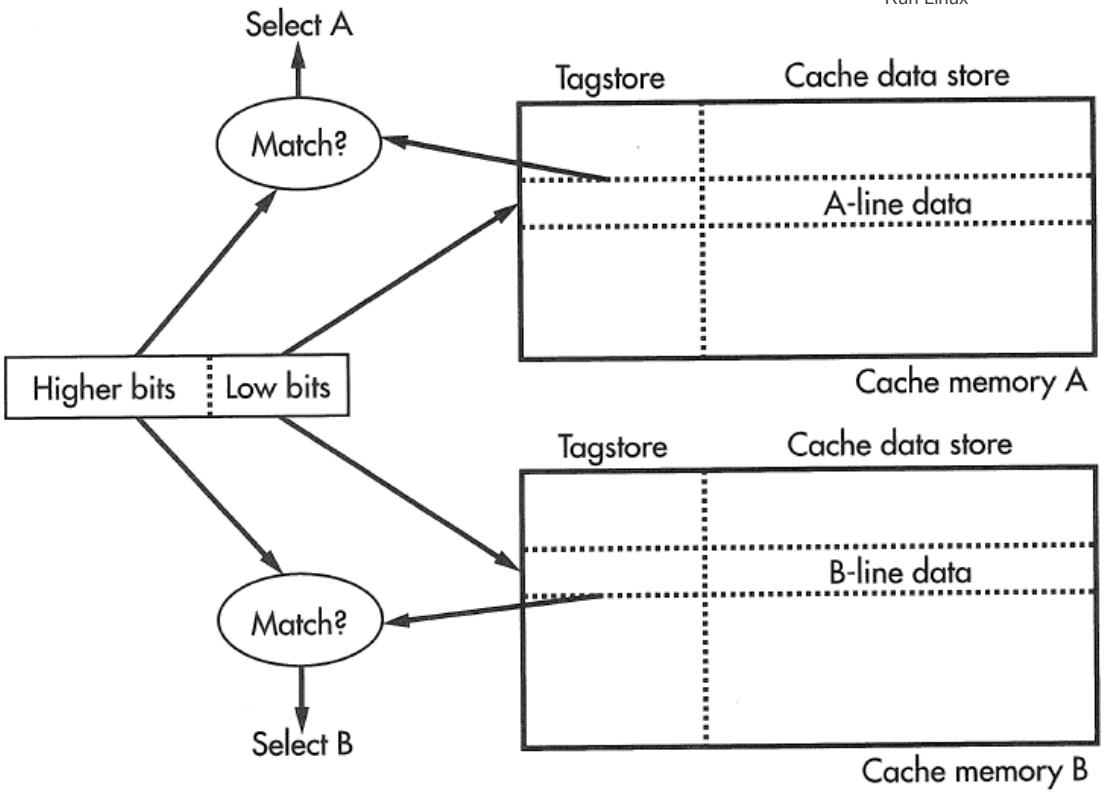
Источник:
Dominic Sweetman.
See MIPS Run Linux

Двухканальный множественно-ассоциативный кэш

Two-way set associative cache

Источник: Dominic Sweetman. See MIPS Run Linux

- **Терминология (продолжение)**
  - Канал
    - Way
  - Политика замещения
    - Replacement policy
  - Наименее последний используемый
    - Least Recently Used (LRU)

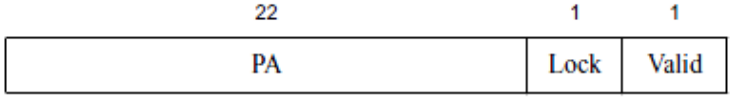


Параметры кэшей в PIC32MZ

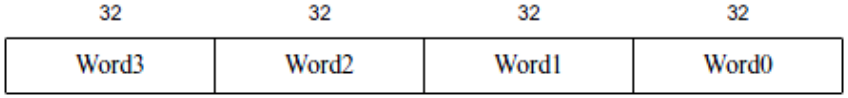
Частный случай конфигураций ядра MIPS microAptiv UP

- **L1 I-Cache – 16KB, 4-канальный множественно-ассоциативный кэш инструкций**
- **L1 D-Cache – 4KB , 4-канальный множественно-ассоциативный кэш данных**

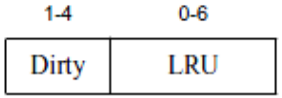
Tag (per way):



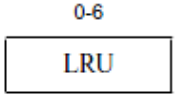
Data (per way):



D Way-Select:



I Way-Select:



Политика записи и выделения линии кэша

Write policy and write allocation

- **Политика сквозной записи**
  - Write-through
  - Любая запись в кэш (с попаданием или без) приводит к записи в память
- **Два варианта действия при промахе во время сквозной записи**
  - Выделять строку в кэше (write allocation) или нет
- **Политика отложенной записи**
  - Write-back
  - Иногда называют некорректно «обратная запись»
  - При попадании в кэш во время записи - запись в память не производится
  - Запись в память производится при вытеснении строки другой строкой (с другим тегом)
  - При политике отложенной записи выделение строки при промахе происходит всегда

Атрибуты кэшируемости и когерентности

Cache Coherency Attributes (CCA)

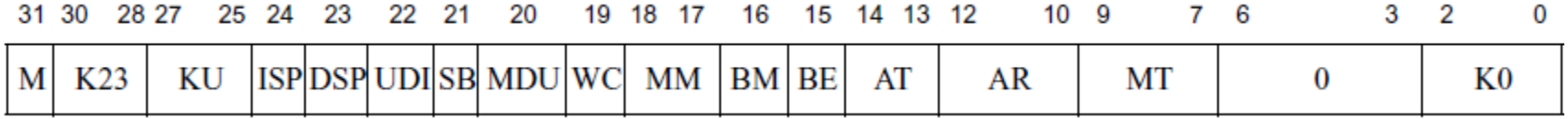
- **В PIC32MZ / MIPS microAptiv UP поддерживаются четыре аттрибута**
  - Uncached
  - Cacheable, noncoherent, write-through, no write-allocate
  - Cacheable, noncoherent, write-through, write-allocate
  - Cacheable, noncoherent, write-back, write-allocate

- **В PIC32MX / MIPS M4K поддерживаются только два аттрибута**
  - Uncached и cacheable
  - Используется для контроля внешнего (по отношению к M4K) кэша префетчера флэша

- **CCA может выставляться в двух местах ядер MIPS**
  - Если ядро сконфигурировано для MMU с TLB (случай PIC32MZ) – в атрибутах страницы
  - Если ядро сконфигурировано для MMU с FMT (случай PIC32MX) – в регистре Cop0 *Config*

Атрибуты кэшируемости и когерентности в Cop0 *Config*

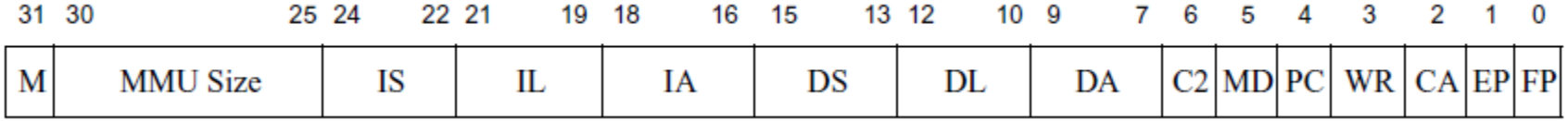
Частично относится к PIC32MX, не используется в PIC32MZ



- **Cop0 Config.K0 содержит атрибуты кэшируемости для сегмента kseg0**
  - 2 - uncached и 3 - cacheable
  - Включает внешний (по отношению к ядру MIPS M4K) кэша префетчера флэша
- **Другие подобные поля *Config***
  - KU для kuseg/useg
  - K23 для kseg2/kseg3 (не используется в PIC32)

Параметры кэшей в регистре Cop0 Config1

Полезно для написания кода, портабильного между несколькими ядрами MIPS



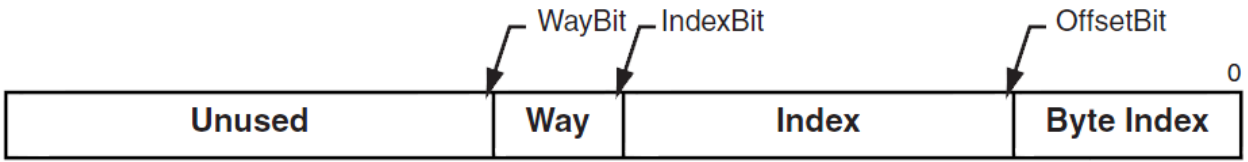
- **Cop0 Config1.IS – кодирует количество строк на канал в L1 кэше инструкций**
  - 0 – 64 строки, 1 – 128, 2 – 256, 3 – 512, 4 – 1024, кодировки 5-6 не используются
  - Кодировка 7 – 32 строки. Особый случай в других ядрах (не MIPS microAptiv UP)
- **IL – кодирует количество байтов в строке в L1 кэше инструкций**
  - 0 – кэш отсутствует, 3 – 16 байтов, 4 – 32 байта (в других ядрах), другие для L1 кэша инструкций не используются
- **IA - кодирует количество каналов в L1 кэше инструкций**
  - 0 – кэш прямого отображений, 1 – двух канальный, 2 – трехканальный, 3 – четырехканальный , другие кодировки для I-L1$ не используются
- **DS, DL, DA – аналогично для L1 кэша данных**
- **Для других ядер MIPS есть также параметры для L2 и L3 в регистре Config2**

Инструкция CACHE op, offset (base)

Для «ручного» управления кэшем

- ***op* состоит из кода операции с кэшем, а также кода кэша**
  - Код кэша – L1 для инструкций, L1 для данных, для других ядер также L2, L3
- **Коды операций с операндом-индексом**
  - Index Invalidate / Index Writeback Invalidate, Load Tag, Store Tag, Store Data
- **Коды операций с операндом-адресом**
  - Hit Invalidate, Fill / Hit Writeback Invalidate, Hit Writeback, Fetch and Lock

Формат операнда-индекса:



Инструкция CACHE – продолжение

Использование и регистры

- **Для инициализации кэшей нужно использовать CACHE StoreTag**
  - Поставить регистр Cop0 TagLo0.V (valid) = 0 и записать invalid tag во все строки
- **Для начального заполнения кэша инструкций можно делать CACHE Fill**
- **Для отладочной проверки состояния кэша CACHE LoadTag**
  - В зависимости от Cop0 ErrCtl.WST чтение будет происходить либо из Tag Array, либо из Way Select Array
  - Если Cop0 ErrCtl.WST= 0, тогда Cop0 TagLo0 будет содержать Tag, Valid, Dirty, Locked
  - Если Cop0 ErrCtl.WST= 1, тогда Cop0 TagLo0 будет содержать LRU
  - LRU = Least Recently Used. Отражает, какой канал использовался последним

Инструкция CACHE – продолжение

Другие случаи использования

- **До начала DMA из кэшируемой памяти – записать все из кэша**
  - CACHE HitWriteback или CACHE HitWritebackInvalidate
- **До начала DMA в кэшируемую память – убрать информацию из кэша**
  - CACHE HitInvalidate
- **Локировать линию – полезно для обработчиков прерываний**
  - CACHE Fetch and Lock

Инструкция CACHE – окончание

Использование

- **Очистить весь кэш – бывает нужно довольно редко**
  - CACHE IndexInvalidate
- **Для синхронизации кэша данных и кэша инструкций вместо последовательноси из CACHE можно использовать SYNCI offset(base)**
- **Для обычного чтения и записи кэш работает автоматически**
  - Инструкция CACHE – исключение, а не правило

Виртуально индексируемые и физически тэгируемые

Virtually Indexed and Physically Tagged (VIPT)

- **Свойство всех L1 кэшей ядер MIPS от MIPS Technologies**
- **Индекс берется от адреса до его трансляции MMU**
- **Тэг берется от адреса после его трансляции MMU**
- **Преимущество – скорость: кэш может работать параллельно с MMU**

- **Недостаток VIPT в некоторых ядрах - cache aliases**
  - Возникает, когда размер кэша для одного канала больше, чем размер страницы в MMU
  - Может привести к нескольким разным копиям данных для одного физического адреса
  - Требует специальных мер (программных и/или аппаратных) для корректной работы кэша

- **В PIC32MZ проблема cache aliases не возникает**

For Distribution

Спасибо!