Лабораторная работа №3	ФИО: Кинзябулатов Эдуард Шамилевич
Название работы: Кэш	Группа: M3137

Цель работы: моделирование системы “процессор-кэш-память”

Ссылка на репозиторий: https://github.com/skkv-itmo/itmo-comp-arch-2023-cache-Tortik3000

Инструментарий и требования к работе: Python (3.11.6)

Результат работы:

LRU: hit perc. 99.5437% time: 3680504

pLRU: hit perc. 99.5312% time: 3685167

Вариант – 2

Кэш

MEM_SIZE	1 Мбайт
ADDR_LEN	20 бит
Конфигурация кэша	look-through write-back
Политика вытеснения кэша	LRU и bit-pLRU
CACHE_WAY	4
CACHE_TAG_LEN	10 бит
CACHE_IDX_LEN	4 бит
CACHE_OFFSET_LEN	6 бит
CACHE_SIZE	4 Кбайт
CACHE_LINE_SIZE	64 байт
CACHE_LINE_COUNT	64
CACHE_SETS_COUNT	16

ADDR_LEN = log2(MEM_SIZE)

CACHE_LINE_SIZE = 2 ^ CACHE_LOFFSET_LEN

CACHE_LINE_COUNT = CACHE_SIZE / CACHE_LINE_SIZE

CACHE_SETS_COUNT = 2 ^ CACHE_IND_LEN

CACHE_WAY = CACHE_LINE_COUNT / CACHE_SETS_COUNT

CACHE_TAG_LEN = ADDR_LEN - CACHE_IDX_LEN – CACHE_OFFSET_LEN

Размерность шин

Шина	Обозначение	Размерность
A1, A2	ADDR1_BUS_SIZE ADDR2_BUS_SIZE	20 бит 14 бит
D1, D2	DATA1_BUS_SIZE DATA2_BUS_SIZE	16 бит
C1, C2	CTR1_BUS_SIZE CTR2_BUS_SIZE	3 бита 2 бита

ADDR1_BUS_SIZE = ADDR_LEN

ADDR2_BUS_SIZE = CACHE_TAG_LEN + CACHE_IDX_LEN

8 команд CTR1_BUS_SIZE = 3 бита

3 команды CTR2_BUS_SIZE = 2 бита

Кэш – look-through. В этой системе процессор и память не связаны, когда процессор подает запрос в кэш, если кэш-попадание, то кэш отправляет ответ, иначе отправляет запрос в память.

Write-back – измененные данные записываются сначала в кэш, а при вытеснении из кэша записываются в память.

Кэш в коде представлен массивом блоков линий.

Когда процессор передает команду чтения, сначала идет проверка на наличие линии с нужным тэгом, если кэш попадание, то передаем процессору нужные байты.

Иначе выбираем в кэш блоке линию к которой дольше всего не было обращения

и если флаг dirty = 1, то передаем это линию на запись в память, потом из памяти в кэш записывается нужная кэш-линия.

При записи, сначала идет проверка на наличие линии с нужным тэгом, если такая найдена

то в нее записываются измененные байты и значение флага dirty = 1. Иначе ищется самая старая линия, если dirty = 1, то пред тем как записать строку из память эта строка записывается в память.

Реализация LRU:

После обращения к кэш-линии ее возраст становится равным нулю а остальные возраста в блоке увеличиваются на 1.

```
def timeLRU(indexBlock, indexLine):
    block = cache[indexBlock]
    for line in block:
        line[1] += 1
    block[indexLine][1] = 0
    block.sort(key=lambda x: x[1])
    for i in range(CACHE_WAY):
        block[i][1] = i
```

Реализация pLRU:

После обращения к кэш-линии ее бит возраста становится равным единице и если количество

кэш-линий в блоке с битом возраста равным единице равно CACHE_WAY, то все остальные биты возраста переходят в 0

```python
def timePLRU(indexBlock, indexLine):
    block = cache[indexBlock]
    count = 0
    for line in block:
        if(line[1] == 1):
            count += 1
    if count == CACHE_WAY:
        for line in block:
            line[1] = 0
    block[indexLine][1] = 1
```