P5 Cache

《计算机组织结构》课程助教组

作业要求

- 实现cache中fetch方法查询数据块在cache中的行号

```
private int fetch(String pAddr)
```

- 实现cache中map方法根据数据在内存中的块号映射

```
private int map(int blockNO)
```

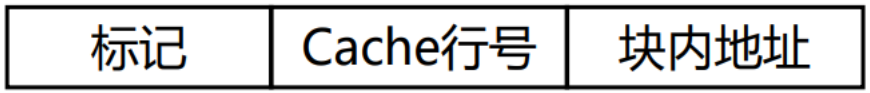
- 基于写回和写直达完善write方法

```
public void write(String pAddr, int len, byte[] data)
```

- 完成FIFO、LFU、LRU三种cache替换策略

映射策略

直接映射　　32　主存地址

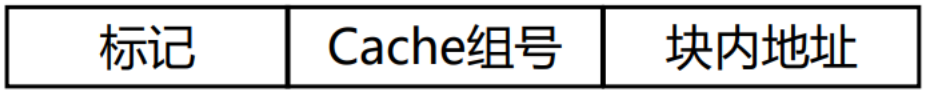


9组 每组1行

全关联映射　　主存地址

1组 每组9行

组关联映射　　主存地址



n组 每组512/n行

映射策略-map

1. 根据内存中blockNO获取在cache中的组号以及对应的tag

   获取组号：blockNO % SETS 块号mod组数

   获取tag：由于不同映射策略有不同的块号位数，我们统一采用26位，不够的使用前导0补齐（不改变大小）。可以调用Transformer.intToBinary方法获得tag。

2. 在组内搜索每一个CacheLine是否有匹配的tag。有直接返回cache行号，没有返回-1。

   注意：

   在访问行号时，请注意判断validBit，即该行存储的是否为有效数据。

   此时如果有匹配上的行号可以调用替换策略中的hit方法表示命中。

映射策略-fetch

1. 调用getBlockNO方法将物理地址转化为对应的内存块号
2. 调用map方法获取对应的行号
3. 命中（即非-1）直接返回，未命中从内存中加载

   加载步骤：

   1. 计算映射到cache组号、tag

   2. 如果是直接映射，调用update方法（在update方法中根据行号直接更新）

   3. 如果不是，调用替换策略的replace方法实现加载

   4. 返回行号

替换策略-FIFO

- hit: do nothing
- replace：使用CacheLine中的timeStamp属性记录该行数据进入时间，替换时使用update方法替换掉时间最小一行。

注意：可以使用`System.currentTimeMillis()`来获取当前时间，得到的是Long类型的数据。

替换策略-LRU

- hit: 重置时间戳，即将被访问的一行的timeStamp更新为当前时间
- replace：替换时和FIFO中一样，使用update方法替换掉时间最小一行。

注意：可以使用`System.currentTimeMillis()`来获取当前时间，得到的是Long类型的数据。

替换策略-LFU

- hit: 被访问的Cache行visited字段值加1
- replace：替换时使用update方法替换掉visited次数最小一行。

【总结】update方法中需要完成的内容：

- 重置标记位：

```
validBit = true;
visited = 1;
timeStamp = System.currentTimeMillis();
```

- 更新tag：`System.arraycopy(tag, 0, this.tag, 0, tag.length);`
- 更新数据：`System.arraycopy(input, 0, this.data, 0, input.length);`

写策略-写直达

写直达策略直接在调用write方法时同时写入cache和memory，写入cache的方法框架代码中已经给出，此处只需调用memory中的write方法。

写入memory时需要物理地址，但此处调用了fetch方法只能获取到行号，因此需要根据行号获取到物理地址。

```
/**
 * 根据行号计算该行首地址对应的物理地址
 *
 * @param rowNO 行号
 * @return 对应的物理地址
 */
public String calculatePAddr(int rowNO) {
    //取出tag有效位
    //计算行号位于的组数
    //加上“000000”表示块内位置的起始位置
    return new String(tag).substring(offset, tag.length) + setNo + "000000";
}
```

写策略-写回

写直达策略直接在调用write方法时只写入cache，只有在发生替换时才会写回memory。因此，在write方法中，只把对应脏位置为true。

在替换策略的replace方法中，检查被替换行的有效位和脏位，只有当数据行有效且脏位被置为true时，才如写直达中一样计算物理地址并向memory写入。