# ELEC3441_Hw2

**JIANG Feiyu 3035770800**

## A.1 Resolving Branch Hazards

### 1.1

| Sequence | Instruction Address | Instruction | Branch Taken? (Y/N) |
|---|---|---|---|
| 1 | I10 | beqz | Y |
| 2 | I14 | bnez | Y |
| 3 | I10 | beqz | Y |
| 4 | I14 | bnez | Y |
| 5 | I10 | beqz | Y |
| 6 | I14 | bnez | Y |
| 7 | I10 | beqz | Y |
| 8 | I14 | bnez | Y |
| 9 | I10 | beqz | Y |
| 10 | I14 | bnez | Y |
| 11 | I10 | beqz | N |
| 12 | I14 | bnez | Y |
| 13 | I10 | beqz | N |
| 14 | I14 | bnez | Y |
| 15 | I10 | beqz | Y |
| 16 | I14 | bnez | Y |

| Sequence | Instruction Address | Instruction | Branch Taken? (Y/N) |
|---|---|---|---|
| 17 | I10 | beqz | Y |
| 18 | I14 | bnez | Y |
| 19 | I10 | beqz | N |
| 20 | I14 | bnez | N |

## 1.2

$Required\ cycles = 2 + 7 \times 15 + 2 \times (15 + 1) + 1 \times (15 + 1 + 3) = 158.$
When considering the initial latency, $11 + 158 = 169$ cycles are needed.

## 1.3

An 4 cycles delay will occur for misprediction of one branch. Thus, the $required\ cycles = 2 + 7 \times (11 + 4 + 4) + 2 \times (11 + 1 + 4) + 1 \times (11 + 1 + 3) = 182$

When considering the initial latency, $11 + 182 = 193$ cycles are needed.

## 1.4

There will be $20 \times 5$ mispreiction, thus, $Required\ cycles = 2 + 7 \times 11 + 2 \times (11 + 1) + 11 + 1 + 3 + 4 = 122.$

Considering about the initial latency, $11 + 122 = 133$ cycles are needed.

## 1.5

$S1\ time = 168 \times 1 = 169$

$S2\ time = 193 \times 1.5 = 289.5$

$S3\ time = 133 \times 2 = 266$

$S1 < S2 < S3$, thus, S1 is the best.

# A.2 Cache Access

## 2.1

| index | tag | valid | dirty | data | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 1 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 2 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 3 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 4 | 5D221F | 1 | 1 | BEEF0007 | BEEF0006 | 0EEE3441 | 0EEE3441 | BEEF0003 | BEEF0002 | BEEF0001 | 0EEE3441 |
| 5 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 6 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 7 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 8 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 9 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 10 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 11 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 12 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 13 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 14 | - | 0 | 0 | - | - | - | - | - | - | - | - |
| 15 | - | 0 | 0 | - | - | - | - | - | - | - | - |



Assume all data in main memory contain the value `0x0EEE3441`. According to the question,

- Capacity: 512B
- Line size: 8 words
- Organization: direct map
- Policy: write back, write allocate

$$offset\ size = log_2(line\ size \times 4) = 5\ bit$$

$$index\ size = log_2(cache\ capacity / line\ size) = 4bit$$

Thus, the tag size should be:

$$32 - offset\ size - index\ size = 23bit$$

As we know, the structure of each line is `tag 23bit + index 4bit + offset 5bit`.

**Cache access:**

| Address (hex) | Type | index | tag | Hit/Miss | Reason |
|---|---|---|---|---|---|
| BA443E98 | R | 0100 | 0x5d221f | H | |
| BA443FA8 | W | 1101 | 0x5d221f | H | Compulsory |

| Address (hex) | Type | index | tag | Hit/Miss | Reason |
|---|---|---|---|---|---|
| 54E99A88 | R | 0100 | 0x2a74cd | M | Compulsory |
| 54E99BB8 | R | 1101 | 0x2a74cd | M | Compulsory |
| 30013C80 | R | 0100 | 0x18009e | M | Compulsory |
| 54E99A84 | R | 0100 | 0x2a74cd | M | Conflict due to 54E99A88 |
| 54E99BA0 | R | 1101 | 0x2a74cd | H | |
| BA443FA4 | W | 1101 | 0x5d221f | M | Conflict due to BA443FA8 |
| 54E99A90 | W | 0100 | 0x2a74cd | H | |
| BA443E88 | W | 0100 | 0x5d221f | M | Conflict due to BA443E98 |

**file content**

| idx | tag | valid | dirty | data7 | data6 | data5 | data4 | data3 | data2 | data1 | data0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ... | | | | | | | | | | | |
| 4 | 5D221F | 1 | 0 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 |
| ... | | | | | | | | | | | |
| 13 | 5D221F | 1 | 1 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 11111111 | 11111111 | 0EEE3441 |
| ... | | | | | | | | | | | |



# 2.2

the structure of each line is `tag 24bit + index 3bit + offset 5bit`.

**Cache access:**

| Address (hex) | Type | index | tag | Hit/Miss | Reason |
|---|---|---|---|---|---|
| BA443E98 | R | 100 | 0xBA443E | H | |
| BA443FA8 | W | 101 | 0xBA443F | H | Compulsory |
| 54E99A88 | R | 100 | 0x54E99A | M | Compulsory |
| 54E99BB8 | R | 101 | 0x54E99B | M | Compulsory |
| 30013C80 | R | 100 | 0x30013C | M | Compulsory |
| 54E99A84 | R | 100 | 0x54E99A | H | |

| Address (hex) | Type | index | tag | Hit/Miss | Reason |
|---|---|---|---|---|---|
| 54E99BA0 | R | 101 | 0x54E99B | H | |
| BA443FA4 | W | 101 | 0xBA443F | H | |
| 54E99A90 | W | 100 | 0x54E99A | H | |
| BA443E88 | W | 100 | 0xBA443E | M | Conflict due to BA443E98 |

**file content**

| idx | tag | valid | dirty | data7 | data6 | data5 | data4 | data3 | data2 | data1 | data0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| … | | | | | | | | | | | |
| 4 | BA443E | 1 | 0 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | BEEF0003 | BEEF0002 | BEEF0001 | 0EEE3441 |
| 4 | 54E99A | 1 | 0 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 11111111 | BEEF0003 | BEEF0002 | BEEF0001 | 0EEE3441 |
| 5 | BA443F | 1 | 1 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 11111111 | 11111111 | 0EEE3441 |
| 5 | 54E99B | 1 | 1 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 |
| … | | | | | | | | | | | |



# 2.3

the structure of each line is `tag 27bit + offset 5bit`.

**Cache access:**

| Address (hex) | Type | tag | Hit/Miss | Reason |
|---|---|---|---|---|
| BA443E98 | R | 0x5D221F4 | H | |
| BA443FA8 | W | 0x5d221FD | H | Compulsory |
| 54E99A88 | R | 0x2a74cd4 | M | Compulsory |
| 54E99BB8 | R | 0x2a74cdD | M | Compulsory |
| 30013C80 | R | 0x18009e4 | M | Compulsory |
| 54E99A84 | R | 0x2a74cd4 | H | |
| 54E99BA0 | R | 0x2a74cdd | H | |
| BA443FA4 | W | 0x5d221fd | M | Compulsory |
| 54E99A90 | W | 0x2a74cd4 | H | |

| Address (hex) | Type | tag | Hit/Miss | Reason |
| --- | --- | --- | --- | --- |
| BA443E88 | W | 0x5d221f4 | H | |

**file content**

| idx | tag | valid | data7 | data6 | data5 | data4 | data3 | data2 | data1 | data0 | Sequence |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | | | | | | | | | | | 2,8 |
| 1 | 2A74CD4 | 1 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 11111111 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 3,6,9 |
| 2 | 2A74CDD | 1 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 4,7 |
| 3 | 18009E4, | 1 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 0EEE3441 | 5, |
| 4 | 5D221F4 | 1 | BEEF0007 | 0EEE3441 | 0EEE3441 | 0EEE3441 | BEEF0003 | BEEF0002 | BEEF0001 | 0EEE3441 | 1,10 |
| 5 | | | | | | | | | | | |
| … | | | | | | | | | | | |
| 15 | | | | | | | | | | | |



## 2.4

The minimum number of misses is 3. Here is one possible design:

| Cache Parameter | Range |
| --- | --- |
| capacity | 256 B |
| line size | 16 words |
| set associativity | 4 |
| write miss policy | write allocate |

# A.3 Simple Pipeline

## 3.1

According to the program, there are 100 loops in total, for each loop, $a_0 += (100 + 17), a1 += (100 - 17)$.

When the loop ends, $a_0 = 10485760 + 100 \times 117 = 10497460$, while $a_1 = 10559596$

Since there are four consecutive add out of the loop, $a_0 = 16 \times 10497460 = 167959360 = 0x0A02DB40$, and similarly, $a_1 = 0x0A1206C0$

## 3.2

$5 + 14 \times 100 + 8 = 1413$ cycles

## 3.3

load use hazard

```
lw t2, 0(s1)
add t3, t1, t2
```

## 3.4

For each Load, the Load Use hazard will cause 4 extra cycles as penalty, which happens for 100 times.

Branch misprediction will cause 3 more cycles as penalty, which happens for 99 times.

Thus,
$CPI = \frac{1413+99\times3+4\times100}{1413} = 1.49$

# A.4 Streaming Cache Performance

## 4.1

```
loop:
    RM * 2
    RH * 7 * 2
    WH * 8
```

$\frac{2^{10}}{8} = 128\ times$

## 4.2

The $Miss\ rate = \frac{2\times\frac{1024}{8}}{3\times1024} = 0.0833$

The $AMAT = Hit\ time + Miss\ rate \times\ Miss\ Penalty = (7 \times 2 + 8) \times 1 + \frac{1}{12} \times 300 = 26\ cycles$

# 4.3

$Total\ run\ time = (5 \times 1 + \times 4 + 26 \times 3) \times N = 91N$

# 4.4

Relacement policy: LRU is applied in the final $2^3$ loops.

$2^{23} > 2^{20}$, the size of array is greater than the cache capacity.

```
loop:
    RM * 2
    RH * 7 * 2
    WH * 8
```

Miss rate doesn't change.

# 4.5

graph LR CPU --read--> Cache CPU --write--> Buffer Buffer --> Memory Memory --write allocate-->Cache
graph LR CPU --read--> Cache Memory --write allocate-->Cache CPU --write--> Memory
All Read is before Write in the code. Therfore there is no difference in cache content except for the dirty bit.

| | **Read miss penalty** | **Write miss penalty** | **Overall performance of the above code** |
|---|---|---|---|
| with write buffer | no effect | no effect | no effect |
| without write | decrease | increase | unknow, almost balanced |

| | Read miss penalty | Write miss penalty | Overall performance of the above code |
|---|---|---|---|
| buffer | | | |

Explanation:

Without write buffer, cache fetches data from memory, thus, read miss penalty decrease; CPU needs to write the cache by the data from memory, thus, write miss penalty increase; the overall performance can't be determined if the read/miss penalty is unknown, the read and write miss penalty are almost balanced.