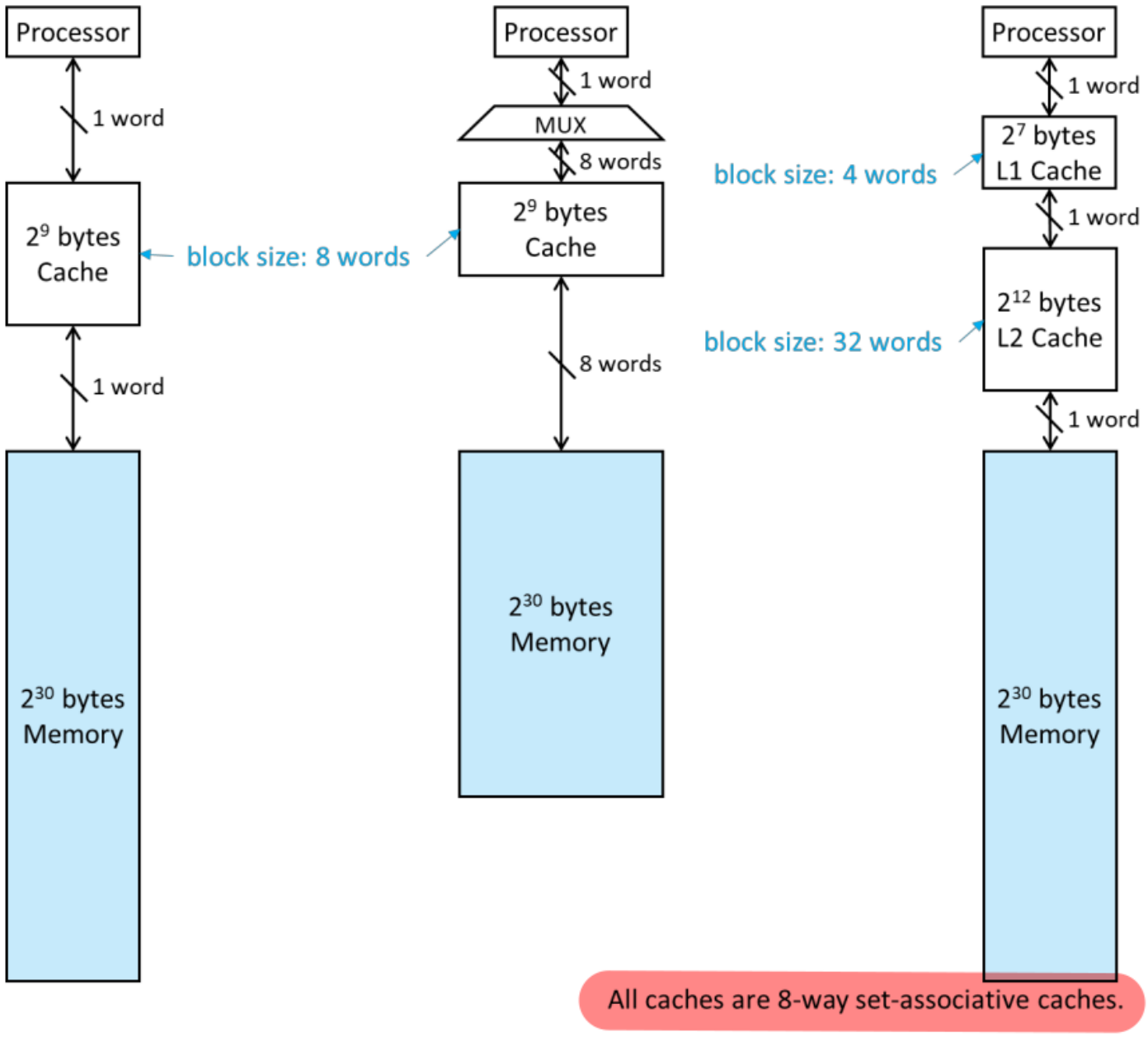
Computer Organiztion LAB5 Report

學號: 0711282邱頎霖 與 0716077 柯柏揚



Stall clock cycle計算:

A
Hit情況的stall clock cycle:
先從Processor送Address(1 clock cycle)，讀Cache內容花(2 clock cycle)，再送Data回去Processor(1 clock cycle)，不管是LW或是SW只要Hit就得Stall 4 clock cycle
算式為:1+2+1=4 clock cycle

Miss的情況:
先從Processor送Address(1 clock cycle)，讀Cache內容花(2 clock cycle)，因為Miss了，所以從Cache送Address到Memory(1 clock cycle)，再讀Memory的內b容花了100 clock cycle，再送Data回去Cache(花 1 clock cycle)，寫入Cache的花費為2 clock cycle，以上步驟要做8次(因為一次僅能1word但總共需要1個Blcok)，最後從Cache送回Processor花1 clock cycle
算式為:1+2+(1+100+1+2)*8+1 clock cycle

B
Hit的情況:
與A情況相同
算式為:1+2+1=4 clock cycle

Miss的情況:
與A的情況大致相同，，但需要注意的是Wider Memory Organization可以一次傳8個Word，所以上述在A中的乘以8可以不用，
算式為:1+2+(1+100+1+2)+1 clock cycle

C

- L1 Hit:
  與A情況大致相同，不同是讀取僅需1 clock cycle，所以算式為1+1+1=3 clock cycle
- L1 Miss:L1 Miss的話需要再往下看L2的反應為何者，分別為:
  - L2 Hit:

        - 如果在L2 Hit的話，整個步驟為:Processor送Address(1 clock cycle)，讀取L1的Cache內容花1 clock cycle，此時發現miss了，L1要送address往下到L2(1 clock cycle)，讀取L2內容10 clock cycle，送Data往上至L1花1 clock cycle，更新L1 Cache的內容花1 clock cycle，因為block size為4個word，故以上步驟需要乘上4，最後將word送回Processor花1 clock cycle
        - 算式為: 1+1+4*(1+10+1+1)+1
    - L2 Miss:
        - 與上述不同的是在L2也沒有Proccesor所需要的item，將繼續往下至Memory尋找，故承上再加上L2送Address往Memory花1 clock cycle，讀取Memory花費100 clock cycle，送資料到L2需要1 clock cycle，更新L2 Cache內容為10 clock cycle，因為此段block size為32words，所以要重複32次
        - 算式為: 1+1+4*(1+10+1+1)+32*(1 + 100 + 1 + 10)+1

Compare:

在開始使用程式模擬前，我們可以預測A與B的比較，一旦發生Miss，A付出的代價一定相對的高，因為透過增加Memory到Cache的頻寬，我們可以減少失誤所帶來的代價。

但我們很難在B與C發生Miss情況下保證誰付出的代價比較高，因為在C的機制下，L1我們在乎的是Hit rate，所以要L1 Cache容量可以小一點，盡可能縮短Access time，但L2相對的要盡可能降低Miss Rate，因此容量要大，但一旦在L2也Miss的時候就有可能L2停頓尋找資料並更新Cache的時間會花的比B多，所以我們可以總結B的目的較為偏向在Miss失誤時所帶來的代價，C則是降低Hit time與Miss rate。

所獲得的數據:
a1xb1:

(備註:Hit_cnt: Hit 的次數
Miss_cnt: Miss 的次數)

A	Hit_cnt	Miss_cnt
cnt	250	6

B	Hit_cnt	Miss_cnt
cnt	250	6

C_l1	Hit_cnt	Miss_cnt
cnt	244	12

C_L2	Hit_cnt	Miss_cnt
cnt	9	3

	A	B	C
Miss_penalty	6016	1648	12144

a2xb2:

A	Hit_cnt	Miss_cnt
cnt	1006	18

B	Hit_cnt	Miss_cnt
cnt	1006	18

C_l1	Hit_cnt	Miss_cnt
cnt	936	88

C_L2	Hit_cnt	Miss_cnt
cnt	81	7

	A	B	C
Miss_penalty	19072	5968	32736

a3xb3:

A	Hit_cnt	Miss_cnt
cnt	23759	9009

B	Hit_cnt	Miss_cnt
cnt	23759	9009

C_l1	Hit_cnt	Miss_cnt
cnt	22400	10368

C_L2	Hit_cnt	Miss_cnt
cnt	10325	43

	A	B	C
Miss_penalty	7626560	1068008	791552

a4xb4:

A	Hit_cnt	Miss_cnt
cnt	751422	297154

B	Hit_cnt	Miss_cnt
cnt	751422	297154

C_l1	Hit_cnt	Miss_cnt
cnt	718398	330178

C_L2	Hit_cnt	Miss_cnt
cnt	57344	272834

	A	B	C
Miss_penalty	251426432	35098320	998152040

我們可以在上述數據上清楚看到在a3xb3的情況下，C中的L2很少miss，其stall clock cycle就勝過於B了；然而在a4xb4的情況中，C中的L2的Miss數大增，一旦L2 miss就會付出大量的時間代價，因此花的Clock cycle相當的多，符合當初我們一開始的想法。