計算機組織 Lab 5

CPU 快取行為模擬

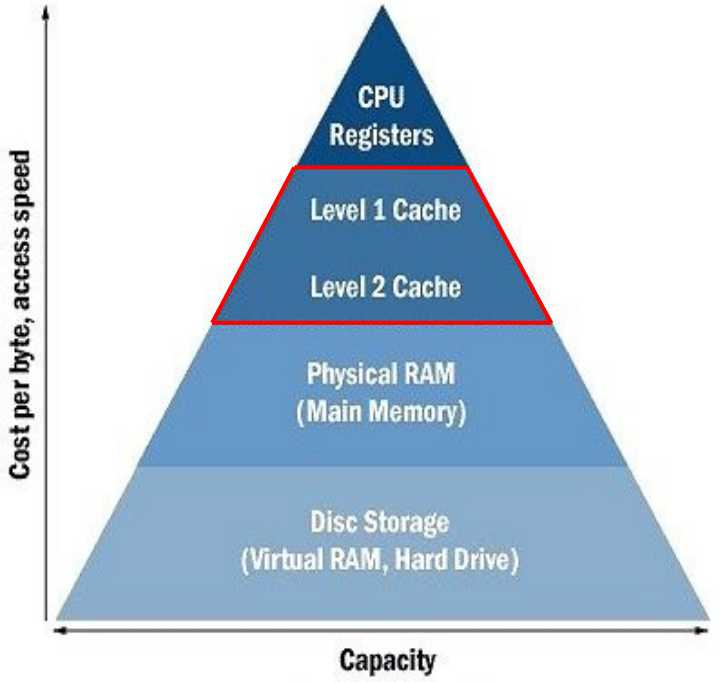
蘇宗埔
2017/12/27

Outline

- Introduction of Cache
  - Cache Memory
  - Direct Mapped Cache
  - Associative Cache
  - Replace Policy
- 實驗目的
- Homework
- 評分標準

Cache Memory

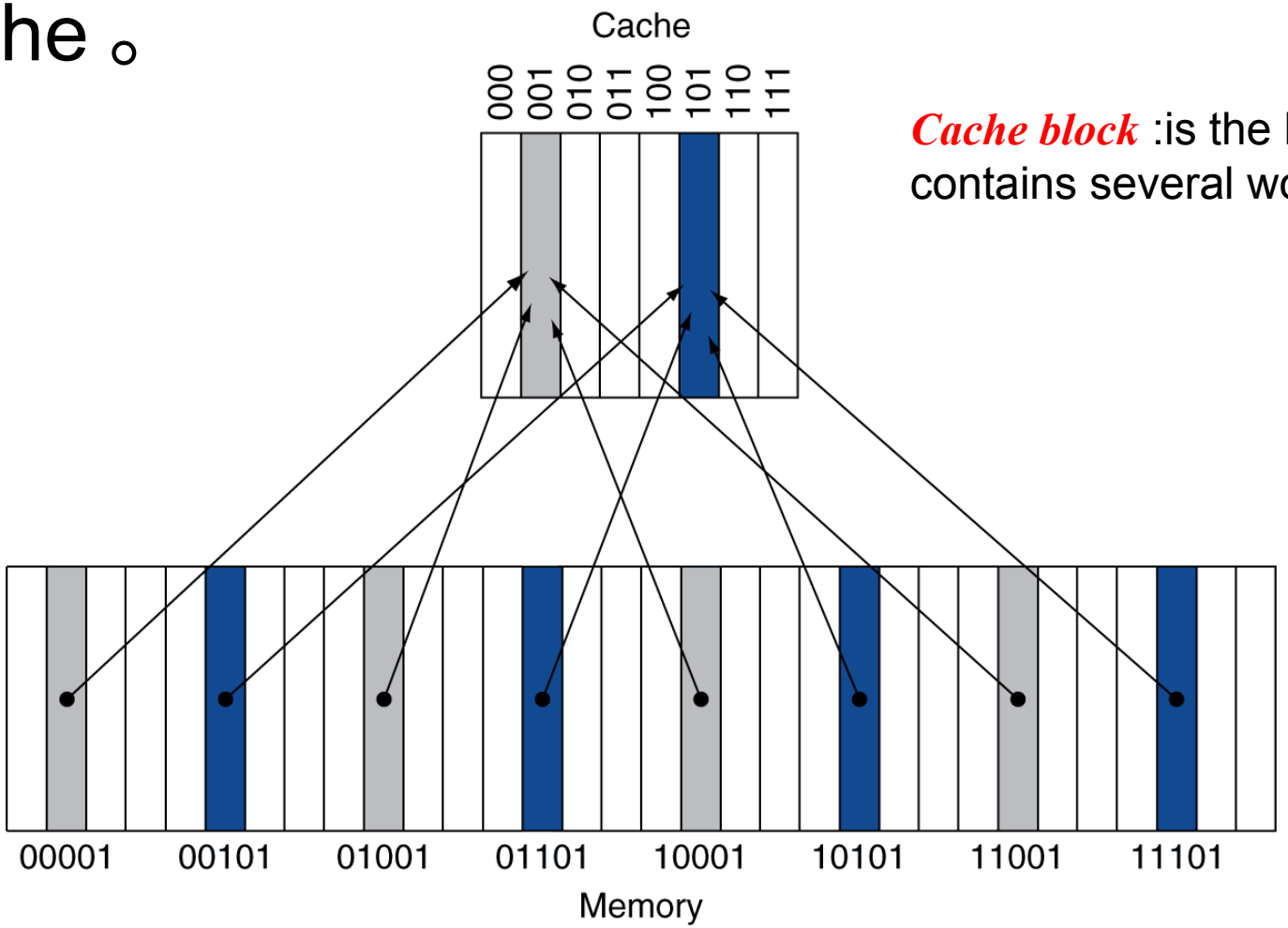
- Cache Memory
  - The level of the memory hierarchy closest to the CPU
- Computer Memory Architecture
  - 越往上層，記憶體的速度越快，容量越小。
  - 　資料只能在相鄰的階層中移動。



computer memory architecture

Direct Mapped Cache

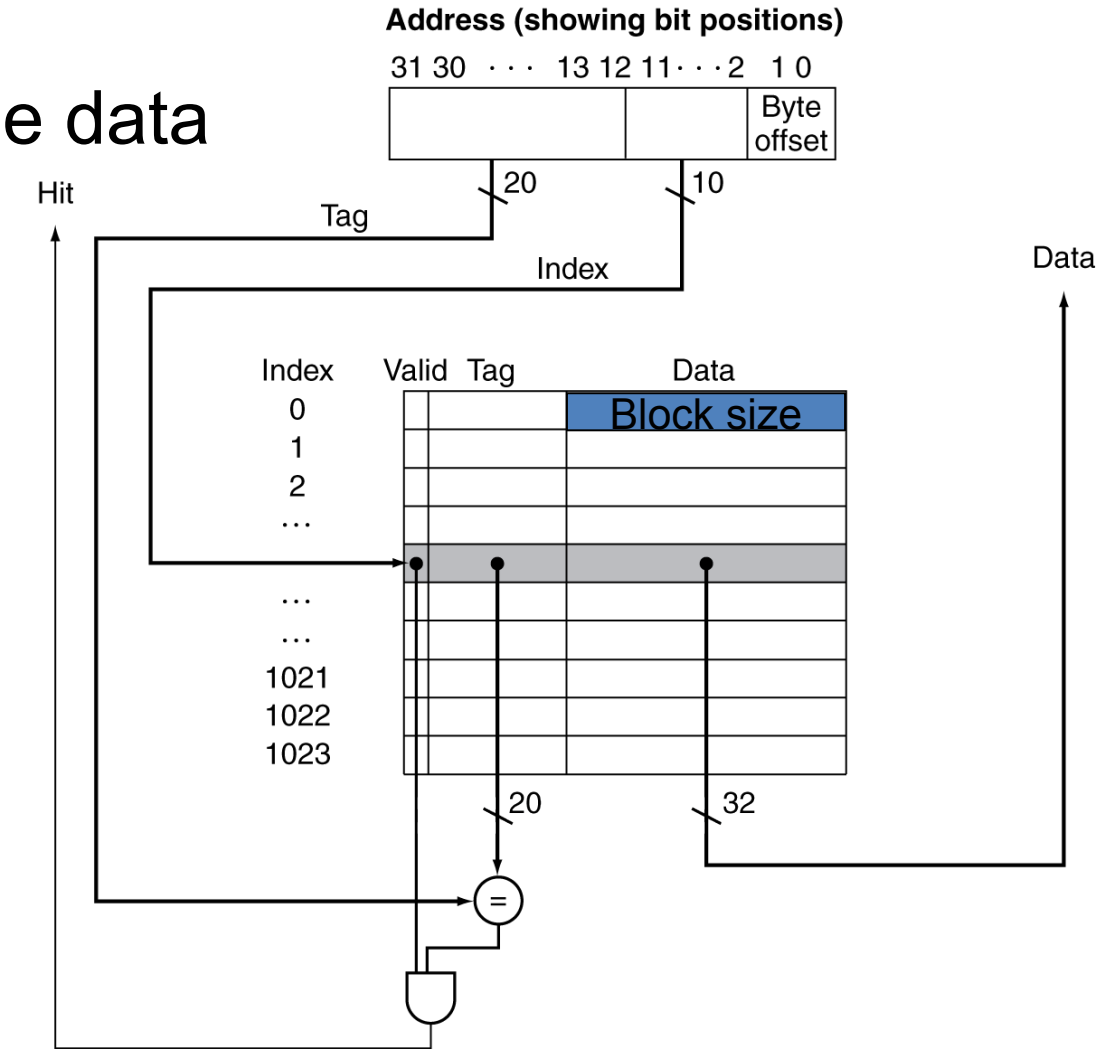
- 根據 Memory 位置，把所有區塊分配給 cache 。



Cache block :is the basic mapping unit, which usually contains several words

Address Subdivision

- Tag
  - Store block address as well as the data
- Vaild bit
  - Cache 中資料是否有效
- Block size
  - 1 words
- Cache size
  - 4 KB (1024 blocks)
    - 4KB =1Kwords = 1K blocks



Direct-mapped Cache Example (1/6)

- 8-blocks, 1 word/block, direct mapped
- Initial state

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

Direct-mapped Cache Example (2/6)

Word addr	Binary addr	Hit/miss	Cache block
22	10 110	Miss	110

Index	V	Tag	Data
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Direct-mapped Cache Example (3/6)

Word addr	Binary addr	Hit/miss	Cache block
26	11 010	Miss	010

Index	V	Tag	Data
000	N		
001	N		
010	**Y**	**11**	**Mem[11010]**
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Direct-mapped Cache Example (4/6)

Word addr	Binary addr	Hit/miss	Cache block
22	10 110	Hit	110
26	11 010	Hit	010

Index	V	Tag	Data
000	N		
001	N		
010	Y	11	Mem[11010]
011	N		
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Direct-mapped Cache Example (5/6)

Word addr	Binary addr	Hit/miss	Cache block
16	10 000	Miss	000
3	00 011	Miss	011
16	10 000	Hit	000

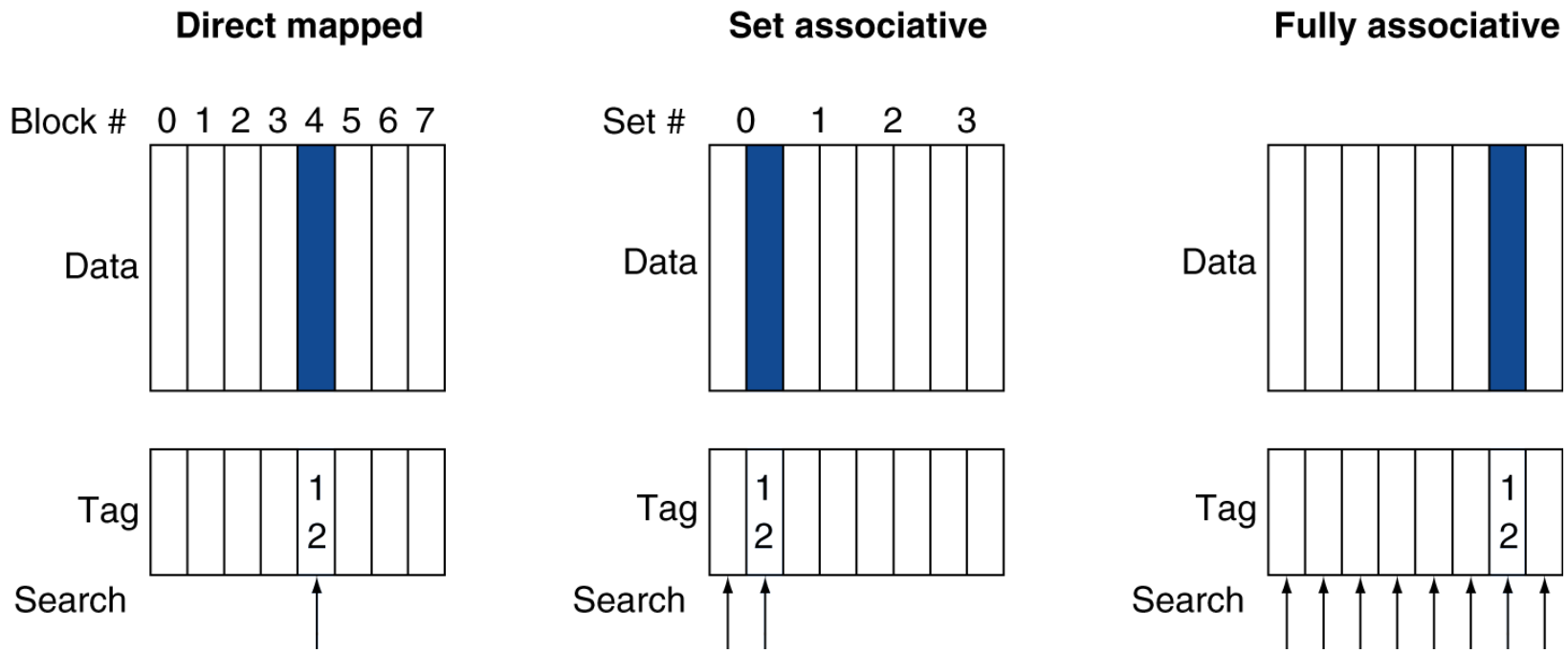
Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	Y	11	Mem[11010]
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Direct-mapped Cache Example (6/6)

Word addr	Binary addr	Hit/miss	Cache block
18	10 010	Miss	010

Index	V	Tag	Data
000	Y	10	Mem[10000]
001	N		
010	**Y**	**10**	**Mem[10010]**
011	Y	00	Mem[00011]
100	N		
101	N		
110	Y	10	Mem[10110]
111	N		

Associative Cache Example (1/3)



Associativity Example (2/3)

- Compare 4-block caches
  - Direct mapped, 2-way set associative,fully associative
  - Block access sequence: 0, 8, 0, 6, 8
- Direct mapped
  - Cache index = Block address % block numbers

Block address	Cache index	Hit/miss	Cache content after access			
			0	1	2	3
0	0	miss	Mem[0]			
8	0	miss	Mem[8]			
0	0	miss	Mem[0]			
6	2	miss	Mem[0]		Mem[6]	
8	0	miss	Mem[8]		Mem[6]	

Cache index

Associativity Example (3/3)

- 2-way set associative

Block access sequence: 0, 8, 0, 6, 8

Block address	Cache index	Hit/miss	Cache content after access			
			Set 0			
0	0	miss	**Mem[0]**			
8	0	miss	Mem[0]	**Mem[8]**		
0	0	hit	**Mem[0]**	Mem[8]		
6	0	miss	Mem[0]	**Mem[6]**		
8	0	miss	**Mem[8]**	Mem[6]		

LRU(least recently used) Policy

Cache index = Block address % set numbers

- Fully associative

Block access sequence: 0, 8, 0, 6, 8

Block address		Hit/miss	Cache content after access			
0		miss	**Mem[0]**			
8		miss	Mem[0]	**Mem[8]**		
0		hit	**Mem[0]**	Mem[8]		
6		miss	Mem[0]	Mem[8]	**Mem[6]**	
8		hit	Mem[0]	**Mem[8]**	Mem[6]	

實驗目的

- 以撰寫程式的方式來模擬 cache 行為，讓大家對 cache 更為熟悉。

Homework 簡介

- LAB5 壓縮檔內容
  - spice.din & gcc.din
  - 內含約十萬筆資料，以模擬 CPU 到 cache 找資料的行為
- 作業內容
  - 利用撰寫一個程式讀取 din 檔，來完成 cache 的行為模擬
    - 本次作業可以選擇使用 C 、 C++ 來撰寫

作業 (Overview)

- CO_Lab5 壓縮包裡面有 .din 檔
  裡面有資料 模擬 CPU Cache 行為時需要的指令資料

- 作業內容
  寫一個程式讀取並處理 din 檔，完成 Cache 的行為模擬

- 壓縮包
  臉書社團上有放

作業（執行檔）

- 把編譯完的執行檔命名為 cache
- 輸入語法為

cache cache_size block_size associativity replace_policy file

1. cache_size 8, 16, …, 256 (KB)
2. block_size 4, 8, 16, …, 128 (B)
3. associativity 1 (direct-mapped), 2, 4, 8, f (fully associative)
4. replace-policy FIFO, LRU
5. file 輸入檔名

- 輸入範例

cache 8 32 2 LRU test.din

作業 (.din 檔內容)

- .din 檔內容格式
  - Label
    - 0 : data read
    - 1 : data write
    - 2 : instruction (cache 只需將 lable2 資料寫入)
  - Label 2 Address
    - 由 Tag 、 Index 、 Offset 所組成且以 16 進位表示

Tag	Index	Offset

Lable Addresss

```
2 408ed4
0 10019d94
2 408ed8
1 10019d88
2 408edc
0 10013220
2 408ee0
2 408ee4
1 100230b8
2 408ee8
0 10013220
2 408eec
2 408ef0
2 408ef4
```

作業（輸出格式）

- 輸出要要有這些東西

Input file
Demand fetch
Cache hit
Cache miss
Miss rate
Read data
Write data
Bytes from Memory
Byte to memory

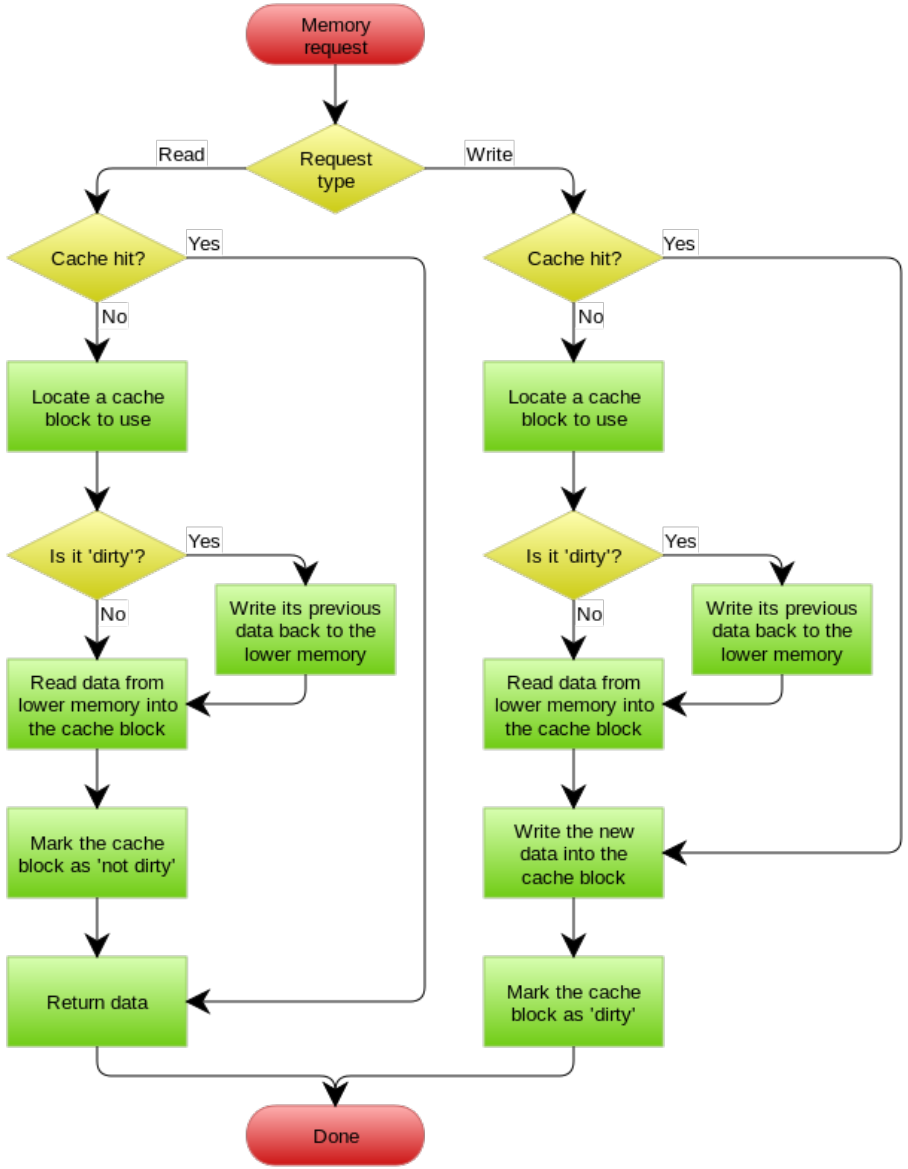
cache 8 32 2 LRU gcc.din

```
Input file = gcc.din
Demand fetch = 1000002
Cache hit = 940024
Cache miss = 59978
Miss rate = 0.0600
Read data = 159631
Write data= 83030
Bytes from memory = 1919296
Bytes to memory = 231424
```

cache 16 16 1 FIFO spice.din

```
Input file = spice.din
Demand fetch = 1000001
Cache hit = 970983
Cache miss = 29018
Miss rate = 0.0290
Read data = 150699
Write data= 66538
Bytes from memory = 464288
Bytes to memory = 71216
```

write-back 是將資料量儲存到一定的量之後，會依據同區塊的資料一次整批寫回去．裡面有提到 dirty，他是在記憶體裡面 cache 的一個 bit 用來指示這筆資料已經被 CPU 修改過但是尚未回寫到儲存裝置中．



作業 (評分標準)

- 作業總共 8 分
- 總共會有八組測資
  一個測資 0.5 分
- 只要有一組測資結果正確就有 2 分 (全錯 0 分 對一個直接 2.5 然後 0.5 0.5 加上去)
- 八組測資全對的話有另外 2 分
- 以助教寫的 Code 的測試結果為基準

作業 （繳交相關）

- 上傳有程式原始碼的壓縮檔到教學平台上
- 不要太奇怪的語言都可以（可用 C, C++, Java, Basic, Python…）非 C/C++ 請附使用說明文件
- Deadline = 1/10

關於作業繳交（所有作業）

- 不管現在補交有沒有分數 都要交作業
  真的有沒有分數請向助教詢問

- 不要繳交不相關的東西
  不接受懺悔書、不接受完全不會動的屍體

- 只要有缺作業，可能會影響學期末計算成績結果
- 在結算期末考前都可以補交

END