DSD HW4 report

B07505026 田昀曜

(a) The cycle time (no timing violation)

- DM

  Cycle time in cache_syn.sdc = 5.0ns (slack=0.02)

  Cycle time in tb_cache.v = 5.0ns

  total cell area = 67054^2um^2

```
Inferred memory devices in process
        in routine cache line 118 in file
                '/home/raid7_2/userb07/b7505026/DSD/hw4/HW4/cache_dm.v'.
=================================================================================
|    Register Name    |   Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
=================================================================================
|      state_reg      | Flip-flop |   2   |  Y  | N  | N  | N  | N  | N  | N  |
|      cache_reg      | Flip-flop | 1240  |  Y  | N  | N  | N  | N  | N  | N  |
=================================================================================
Statistics for MUX_OPs
=======================================================
| block name/line  | Inputs | Outputs | # sel inputs |
=======================================================
|     cache/53     |   8    |    2    |      3       |
|     cache/55     |   8    |   25    |      3       |
|     cache/58     |   8    |   128   |      3       |
|     cache/90     |   8    |   153   |      3       |
=======================================================
```

- 2Way (LRU)

  Cycle time in cache_syn.sdc = 5.0ns (slack=0.0)

  Cycle time in tb_cache.v = 5.5ns

  total cell area = 99530^2um^2

```
syn
Inferred memory devices in process
        in routine cache line 177 in file
                '/home/raid7_2/userb07/b7505026/DSD/hw4/HW4/cache_2way.v'.
=================================================================================
|    Register Name    |   Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
=================================================================================
|       lru_reg       | Flip-flop |   4   |  Y  | N  | N  | N  | N  | N  | N  |
|      cache_reg      | Flip-flop | 1248  |  Y  | N  | N  | N  | N  | N  | N  |
|      state_reg      | Flip-flop |   2   |  Y  | N  | N  | N  | N  | N  | N  |
=================================================================================
Statistics for MUX_OPs
=======================================================
| block name/line  | Inputs | Outputs | # sel inputs |
=======================================================
|     cache/60     |   4    |    5    |      2       |
|     cache/64     |   4    |   52    |      2       |
|     cache/68     |   4    |   256   |      2       |
|    cache/107     |   4    |   312   |      2       |
=======================================================
```

(b) Specification of the cache unit

- DM

  The cache has 8*155bits cache registers. 1 bit for valid, 1 bit for

dirty, 25 bits for tags, 128 bits for data (4 words)

- 2Way

  The cache has 4*312bits cache registers. 1 bit for each valid, 1 bits for each dirty, 26 bits for each tag, 128 bits for each data (4 words), (1+1+26+128)*2 = 312.

  Since there only two elements in each set, there 4 bits for each set. If the lru bit is high, then it means the first space of the set is the least recently used space.

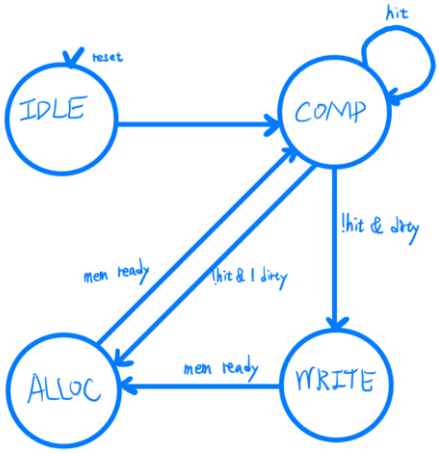
(c) Read/write policy

I use write-back policy for both designs.

(d) Design architecture or the finite state machine

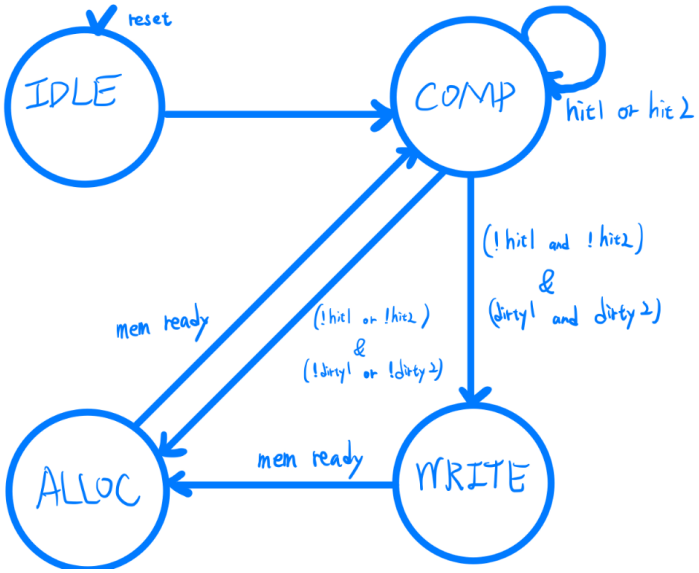
- DM

  There are 4 states in the FSM.



- 2Way

  There are 4 states in the FSM.



(e) Performance evaluation

- DM

	Read 1	Write	Read 2	Total
Count miss	256	256	1024	1536
Execute cyc	1024	1024	1024	3072
Count op	1024	1024	1024	3072
Stall cyc	1792	3528	7224	12544
Total cycle	2816	4552	8248	15616
Miss rate	0.25	0.25	1	0.5

- 2Way

Least Recently Used (LRU) (best)

	Read 1	Write	Read 2	Total
Count miss	256	256	256	768
Execute cyc	1024	1024	1024	3072
Count op	1024	1024	1024	3072
Stall cyc	1792	3556	1820	7168
Total cycle	2816	4580	2844	10240
Miss rate	0.25	0.25	0.25	0.25

(if two rooms in the block are possible, choose the room randomly with proc_addr[0] == proc_wdata[0])

	Read 1	Write	Read 2	Total
Count miss	256	252	384	892
Execute cyc	1024	1024	1024	3072
Count op	1024	1024	1024	3072
Stall cyc	1792	3500	2772	8064
Total cycle	2816	4524	3796	11136
Miss rate	0.25	0.246	0.375	0.29

(f) Compare the performance

"Read initial data": reads sequentially from address 0 to address 1023, so the miss rate of dm is 0.25, and since the write data is also in the order of 0, 1, 2, … , 1023, the miss rate of my random 2way design is also 0.25. The LRU design is also 0.25.

"Write new data" writes sequentially from address 0 to address 1023, so the miss rate is also 0.25, the miss rate of my random 2way design is also close to 0.25. The LRU design is also 0.25.

"Read new data" reads in the order of 0, 32, 1, 33, … , 64, 96, 65, 97, …. Since 0 and 32, 1 and 33, … have the same remainder, 64 and 96, 65 and 97, … also have the same remainder. The miss rates of dm is 1, the miss of my random 2way design is 0.375 (due to some sequential pattern in the loop of tb). The LRU design has the lowest miss rate: 0.25.

(g) Experiences

我覺得 hw4 比 hw3 簡單蠻多的，在設計的時候比較容易想到要怎麼寫，不過在 debug 的時候還是花了很多時間在看 waveform，cache 裡面的值看得眼花撩亂= =。後來學到可以設置 subarray 的方式來看 cache 裡面的值，我覺得非常有用，可以很清楚的看到特定位置的值，不用在很長的 cache 裡面自己對位置。學會了這招之後就迅速地找到 bug 了。

(bonus)

我的設計使用了 random 的方式來增加 hit 的機率，在同一個 cache 裡面如果兩格位置都可以 read /write 的時候，根據 proc_wdata 和

proc_addr 的第 0 個 bit 是否相等來決定要使用位置 1 或是位置 2。

另外一種設計我使用的是 Least Recently Used (LRU)，將集合中距離上次使用最長時間的區塊優先取代掉，這樣在第二次寫入的時候，因為順序剛好是 0, 32, 1, 33, 2, 34, …, 64, 96, 65, 97, …，所以一開始的 0, 32 有兩個 miss，後面的 1, 33, 2, 34, 3, 35 則都是 hit，所以可以把 miss rate 壓在 0.25，是最佳的設計。