

①

Cache

Dirty Bit - D

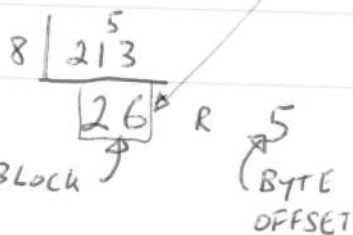
INDEX	VALID BIT	D	TAG	DATA
000	0	0	00	8 BYTES PER CACHE LINE
001	0	0	00	
010	0	0	00	
011	0	0	00	
100	0	0	00	
101	0	0	00	
110	0	0	00	
111	0	0	00	

LEVEL N	CACHE	INDEX	000	100	010	110	100	010	110	---
			0	1	2	3	4	5	6	7

LEVEL N-1	TAG									
	MAIN	00	0	1	2	3	4	5	6	7
	MEMORY	01	8	9	10	11	12	13	14	15
		10	16	17	18	19	20	21	22	23
		11	24	25	26	27	28	29	30	31

ADDRESS 213 :

TAG INDEX BYTE OFFSET  
11 : 010 : 101



2

- FOR ADDRESS 213, BLOCK 26 WILL BE COPIED FROM, MAIN MEMORY (LEVEL N-1) TO CACHE (LEVEL N)
- BLOCK 26 (RAM) IS COPIED TO BLOCK 2 IN CACHE (INDEX : 010)

• VALID BIT	-	SET TO 1	← IN CACHE BLOCK 2
• TAG <del>BIT</del> <sup>BITS</sup>	-	SET TO 11	

INDEX	VALID BIT	DIRTY BIT	TAG	DATA
000	0	0	00	
001	0	0	00	
010	1	0	11	COPY OF BYTES: 208 209 210 211 ... 215
011	0	0	00	
100	0	0	00	
101	0	0	00	
110	0	0	00	
111	0	0	00	

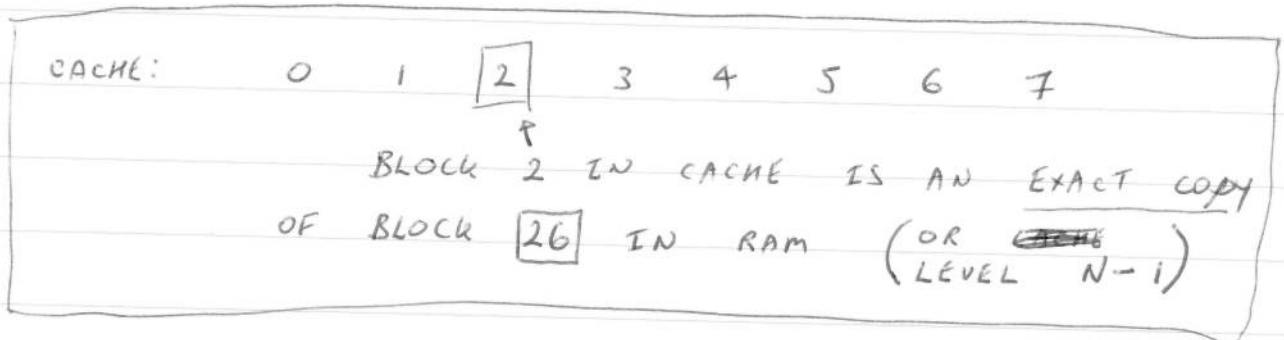
(3)

What happens when a byte anywhere in block 2 in cache is now written to?

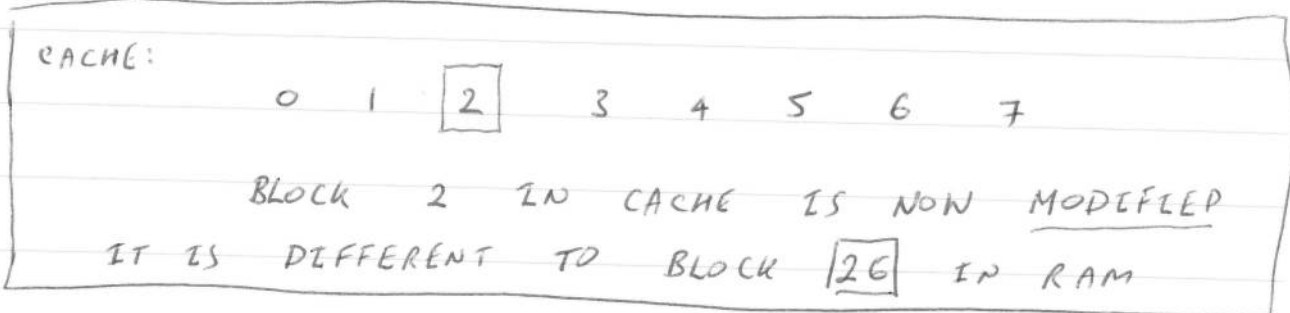
(So far we have only considered reads)

E.g. what happens if we write to byte 209.

BEFORE THE WRITE: (TO CACHE)



AFTER THE WRITE (TO CACHE)



WE CAN SAY THAT ~~IT IS~~  
BLOCK 2 IS DIRTY. TO INDICATE THIS  
THE DIRTY BIT IS SET TO 1.

(4)

When is the DATA WRITTEN BACK TO RAM?

## WRITING POLICIES

- WRITE-THROUGH CACHE
- WRITE-BACK CACHE

### WRITE-THROUGH:

"EVERY TIME THE PROCESSOR WRITES TO A CACHED MEMORY LOCATION, BOTH THE CACHE AND THE UNDERLYING MEMORY LOCATION ARE UPDATED"

### WRITE BACK CACHE

~~"WITH THIS METHOD, EVERY TIME THE PROCESSOR WRITES TO A CACHED MEMORY LOCATION, BOTH THE CACHE AND THE UNDERLYING MEMORY LOCATION ARE UPDATED"~~

### WRITE BACK CACHE

"WHEN A WRITE IS MADE TO A BLOCK THAT IS CURRENTLY CACHED, THE NEW DATA IS ONLY WRITTEN TO THE CACHE (NOT TO THE RAM). LATER WHEN A BLOCK IS EVICTED FROM THE CACHE, (E.G. BECAUSE OF A CONFLICT), THE EVICTED BLOCK WILL BE WRITTEN TO RAM IF THE DIRTY BIT IS SET"