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	001	010	011	100	101	110	111
	0	1	2	3	4	5	6	7

TAG

LEVEL N-1 MAIN MEMORY

TAG								
00	0	1	2	3	4	5	6	7
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

$8 \mid 213$ (5)

26 R 5

BLOCK

BYTE OFFSET

- 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
- TAG BITS – SET TO 11

← IN CACHE BLOCK 2

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	

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)

CACHE: 0 1 [2] 3 4 5 6 7

BLOCK 2 IN CACHE IS AN EXACT COPY OF BLOCK [26] IN RAM (OR LEVEL N-1)

AFTER THE WRITE (TO CACHE)

CACHE:

0 1 [2] 3 4 5 6 7

BLOCK 2 IN CACHE IS NOW MODIFIED
IT IS DIFFERENT TO BLOCK [26] IN RAM

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

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