SET ASSOCIATIVE MAPPING

⇒ FLESSIBILE: C. CTRL PUÓ SCEGLIERE TRA K LINEE LA POSIZIONE IN C. DEL BLOCCO

⇒ VELOCITÀ, ECONOMICITÀ: SOLO K CONFRONTI PER ACCESSO ; ⇒ MOLTO MEGLIO DI ASS MAP.

⇒ TUTTE LE C. SONO SET-ASS. MAP.
$K \in [2, 16]$

ESEMPIO 1

CACHE: LINEE DA 16B
256 LINEE
8 VIE
MEMORIA: 32 bit

32 SET DA 8 LINEE

32 bit

23 5 4

DIM. DATO: 256 X 16B = 4096 B

DIM. TAG: 256 x 23b = 5888b
= 736B

DIM TOT. = 4096B + 736B = 4832B

ESEMPIO 2

MEMORIA: 64 kB
↳ A. BUS: 16 bit
CACHE: 16 LINEE
LINEE DA 32B
S.A.M a 4 VIE

⇒ 4 SET DA 4 LINEE

16 bit

9 2 5

ACCESSI:

352_{10}: 10 | 11 | 00000; T:2; I:3
T I OFF

590_{10}: 100 | 10 | 01110; T:4; I:2
T I OFF

AGGIORNAMENTO DI C.

- IN LETTURA:
  [- C CTRL COPIA IL BLOCCO IN C.]
  - CPU LEGGE DA C.

- IN SCRITTURA

  [- C. CTRL COPIA IL BLOCCO IN C.]

  - CPU SCRIVE IN C.

  ⇒ C. E MEM. HANNO UN DATO ≠

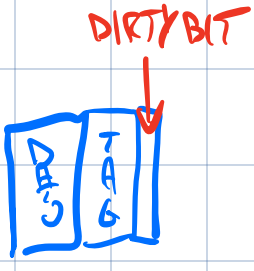
  ⇒ INCOERENZA

- DUE TECNICHE

1. WRITE BACK

- PRIMA DI RIMPIAZZARE IL BLOCCO IN UNA LINEA, AGGIORNA LA C. SE IL BLOCCO È STATO MODIFICATO

⇒ 1 DIRTY BIT: MESSO A 1 SE IL BLOCCO IN UNA LINEA È STATO MODIFICATO



⇒ INTUITIVA

⇒ UNA MISS PUÓ RICHIEDERE 2 ACCESSI IN M.

2. WRITE THROUGH

- QUANDO LA CPU SCRIVE, LO FA SIA SU C. CHE SULLA MEM.

⇒ C. NON USATA IN SCRITTURA

⇒ STUPIDO? NO

⇒ LETTURE PIÙ DI SCRITTURE

⇒ QUANDO C. CTRL RIMPIAZZA IL BLOCCO IN UNA LINEA NON DEVE NIENTE DI AGGIUNTIVO

ENTRAMBE TECNICHE USATE