

Architettura degli Elaboratori

a.a. 2023 – 2024

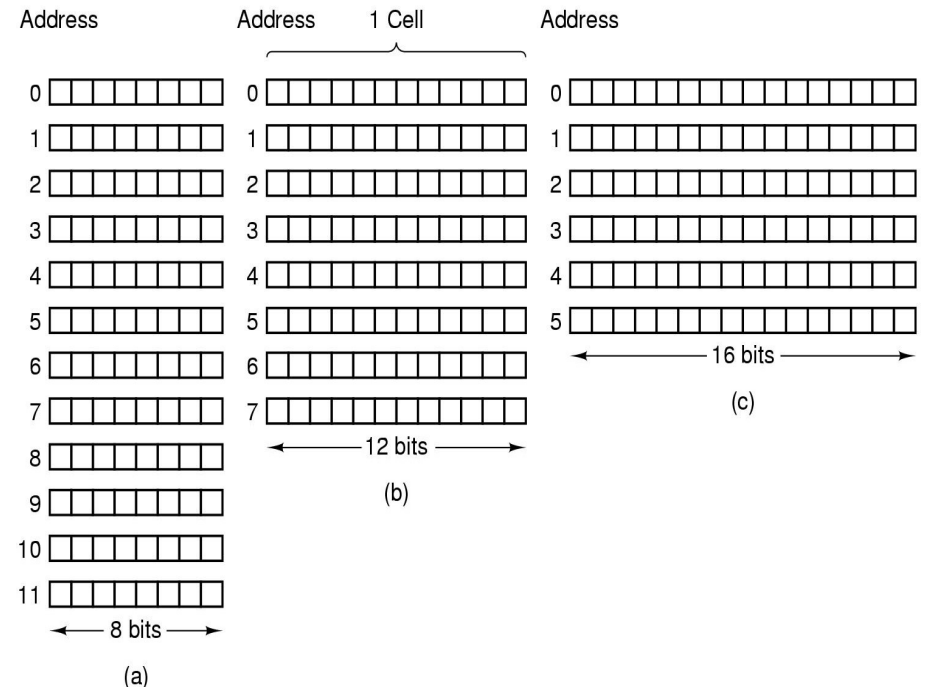
Memoria: Organizzazione e gerarchia

“Many hundred large volumes have been published upon this controversy: but the books of the Big-endians have been long forbidden, and the whole party rendered incapable by law of holding employments”

Jonathan Swift *“Gulliver’s Travels”*, Part 1, Chap. 4

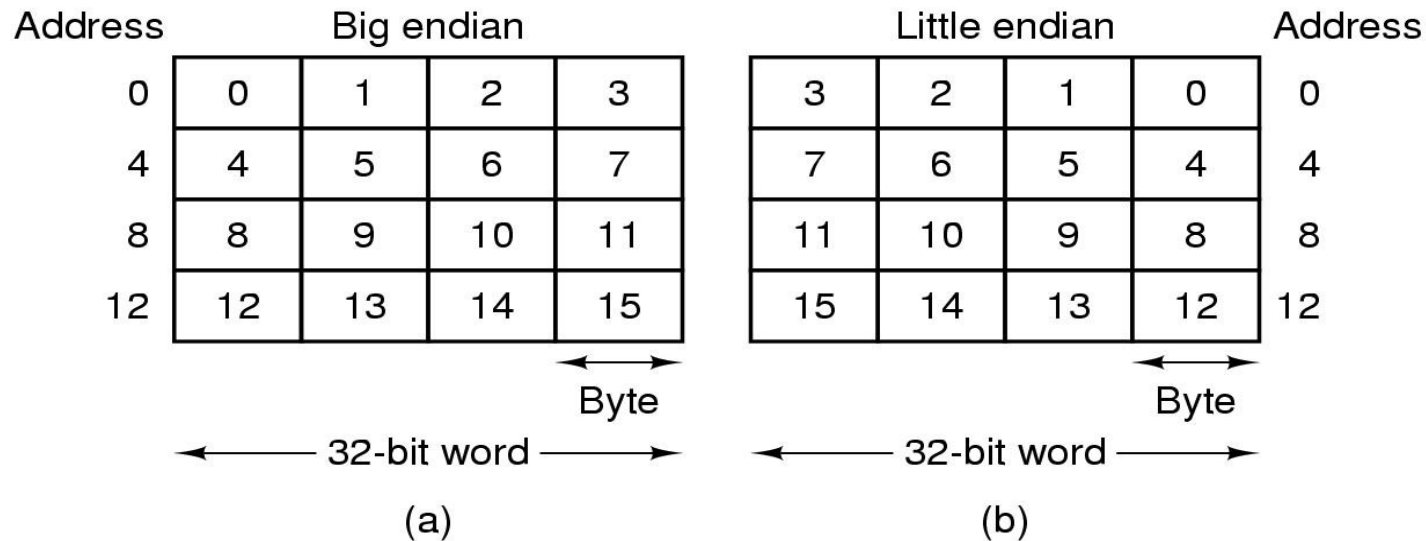
La memoria principale

- Store o storage
- Memoria come insieme di celle (o locazioni)
- Tutte le celle hanno stessa dimensione
- Ogni cella è identificata da un indirizzo
- Se un calcolatore ha n celle allora gli indirizzi vanno da **0** a **$n-1$** *indipendentemente* dalla dimensione della cella
- Il numero di bit nell'indirizzo determina il numero massimo di celle indirizzabili



- La cella è la minima unità indirizzabile
- Recentemente dimensione standard: 8 bit (byte), 4 o 8 byte definiscono una *parola* (unità di manipolazione per le istruzioni)

Ordinamento dei byte



- (a) **big endian**: la numerazione va da sinistra verso destra: *big end first* (SPARC, mainframe IBM)
- (b) **little endian**: la numerazione va da destra verso sinistra *little end first* (Intel, RISC-V)

Ordinamento dei byte: little endian

	7	0
0	\$1D	
1	\$2C	
2	\$3B	
3	\$4A	
4	'C'	
5	'I'	
6	'A'	
7	'O'	
	byte	

	15	8	7		0	0
1	\$2C		\$1D		0	
3	\$4A		\$3B		2	
5	'I'		'C'		4	
7	'O'		'A'		6	
	Word					

	31		16	15		0
3	\$4A	\$3B	\$2C	\$1D		0
7	'O	'A	'I	'C		4
	Long-word					

Organizzazione little endian: numero \$4A3B2C1D E STRINGA "CIAO"

Ordinamento dei byte: big endian

	7	0
0	\$1D	
1	\$2C	
2	\$3B	
3	\$4A	
4	'C'	
5	'I'	
6	'A'	
7	'O'	
	byte	

	15	8	7	0	0
0	\$1D	\$2C		1	
2	\$3B	\$4A		3	
4	'C'	'I'		5	
6	'A'	'O'		7	
	Word				

	31	16	15	0	
0	\$1D	\$2C	\$3B	\$4A	3
4	'C'	'I'	'A'	'O'	7
	Long-word				

Organizzazione big endian: numero \$1D2C3B4A E STRINGA "CIAO"

Ordinamento dei byte

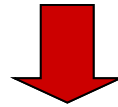
Big endian					Little endian					Transfer from big endian to little endian					Transfer and swap				
0	J	I	M			M	I	J		0		M	I	J	J	I	M		0
4	S	M	I	T	T	I	M	S		4	T	I	M	S	S	M	I	T	4
8	H	0	0	0	0	0	0	H		8	0	0	0	H	H	0	0	0	8
12	0	0	0	21	0	0	0	21		12	21	0	0	0	0	0	0	21	12
16	0	0	1	4	0	0	1	4		16	4	1	0	0	0	0	1	4	16
(a)					(b)					(c)					(d)				

- Mancanza di uno *standard*
- Problemi per la trasmissione byte per byte di informazioni di tipo alfanumerico

CPU e memoria

CPU più veloci delle memorie: quando la CPU effettua una richiesta alla memoria, essa non ottiene la parola desiderata se non dopo molti cicli di CPU.

Piccola quantità di memoria veloce (costosa) o grande quantità di memoria più lenta?

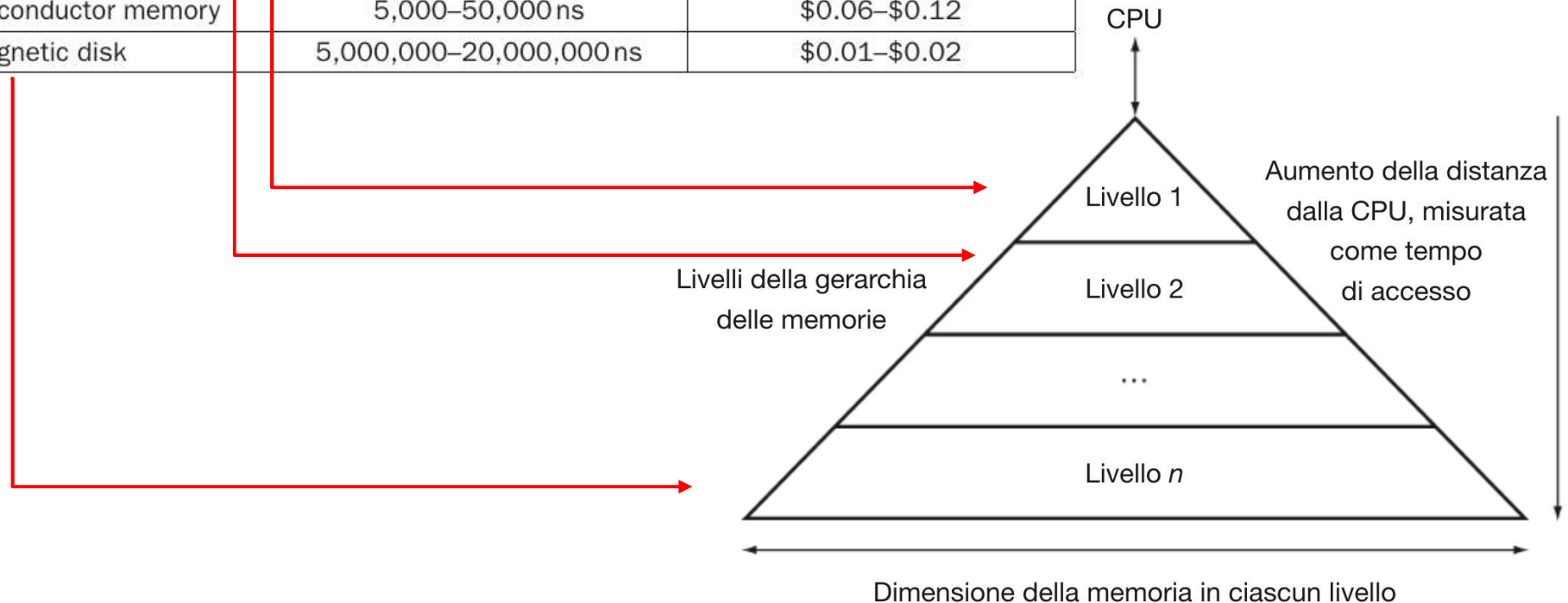


Tecniche che permettono di combinare le due soluzioni per ottenere a un prezzo ragionevole sia la velocità sia la notevole capacità

Memoria piccola e veloce: **cache**

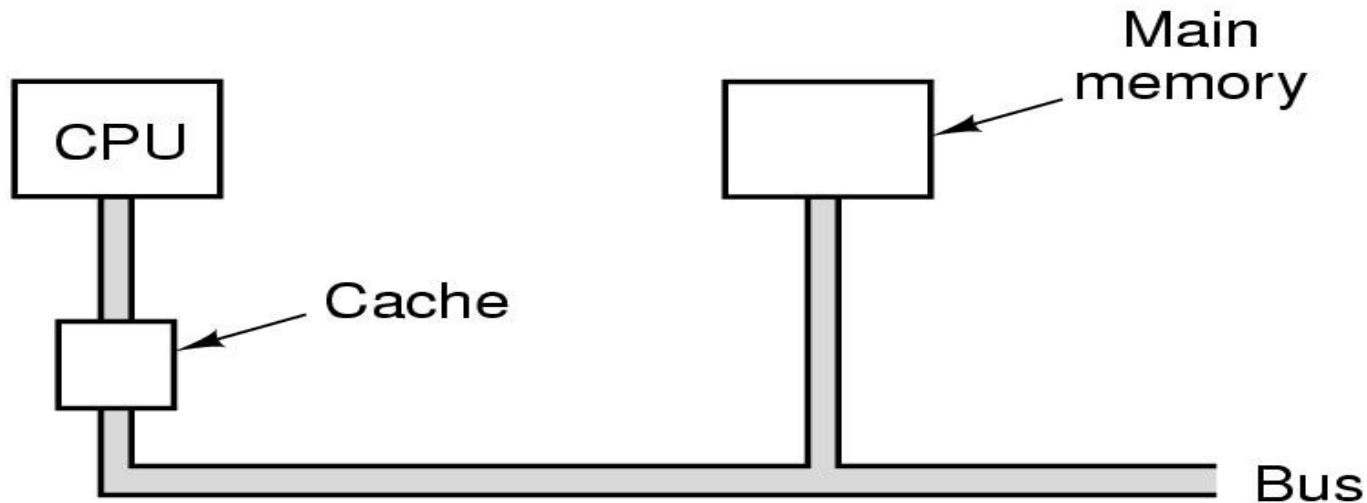
Gerarchia della memoria

Memory technology	Typical access time	\$ per GiB in 2020
SRAM semiconductor memory	0.5–2.5 ns	\$500–\$1000
DRAM semiconductor memory	50–70 ns	\$3–\$6
Flash semiconductor memory	5,000–50,000 ns	\$0.06–\$0.12
Magnetic disk	5,000,000–20,000,000 ns	\$0.01–\$0.02



- Gerarchia della memoria secondo la capacità di memorizzazione
- Questa gerarchia rispetta i costi di memorizzazione (più in basso i più economici, più in alto i più costosi)
- Rispetta anche i tempi di accesso all'informazione

Memoria cache



- L'uso della memoria di cache è basata sul ***principio di località*** del codice: i programmi non accedono alla memoria a caso
- Le parole di memoria più usate vengono tenute in una cache

Località Temporale

Principio secondo cui se si accede a una determinata locazione di memoria, è molto probabile che vi si acceda di nuovo dopo poco tempo.

fib:

```
beq a0,zero,fine
addi t0,zero,1
beq a0,t0,fine
```

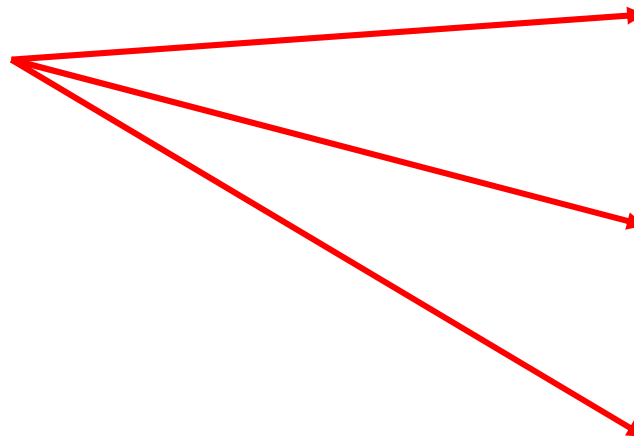
```
# general case n > 1
addi sp,sp,-16
sd ra,0(sp)
sd a0,8(sp)
```

```
addi a0,a0,-1
jal fib
ld t0,8(sp)
sd a0,8(sp)
addi a0,t0,-2
jal fib
```

```
ld t0,8(sp)
add a0,a0,t0
ld ra,0(sp)
addi sp,sp,16
```

fine:

```
ret
```



Località Spaziale

principio secondo cui se si accede a una determinate locazione di memoria, è molto probabile che si acceda alle locazioni vicine ad essa dopo poco tempo.

```
loop:
    beq t1, a1, endloop      # while(i < size)
    lw t2, 0(a0)             # array[i]

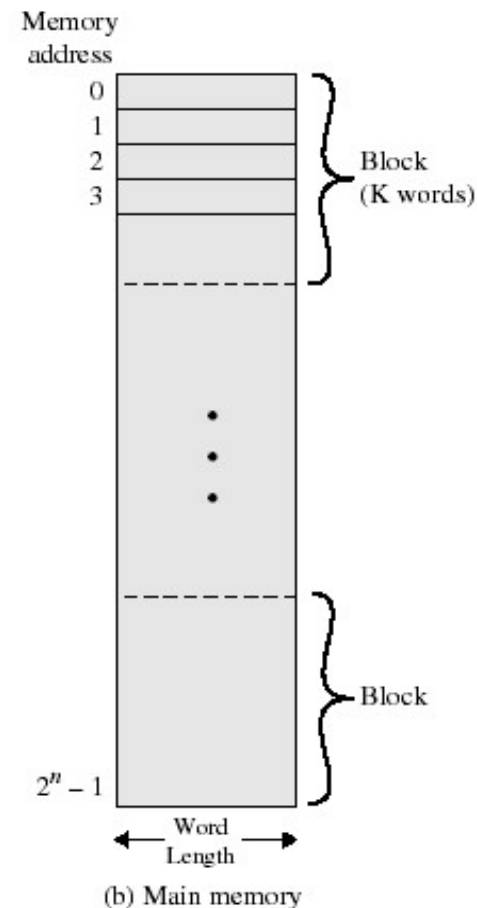
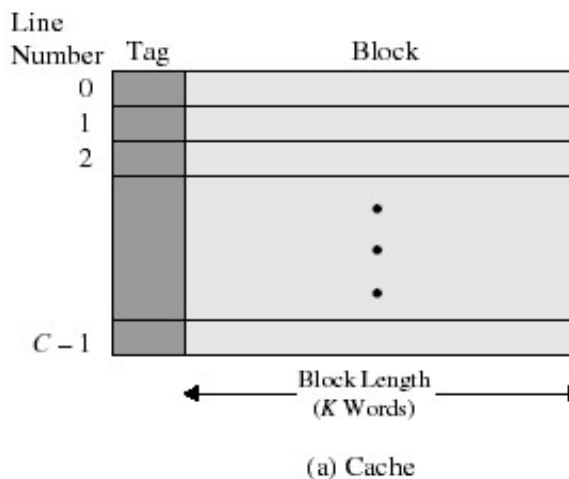
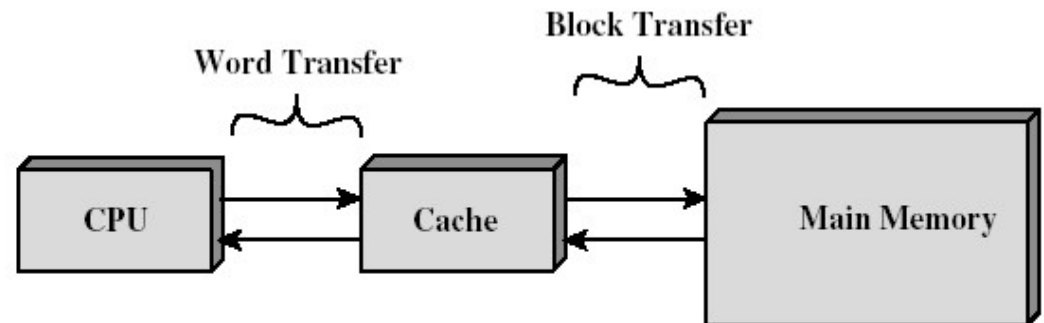
    bge t2, a2, carryon      # if(array[i] < min_val)
    mv t0, t1                # min_idx = i;
    mv a2, t2                # min_val = array[i];

    carryon:
    addi a0, a0, 4           # sposto il puntatore alla prossima word
    addi t1, t1, 1           # ++i;
    j loop

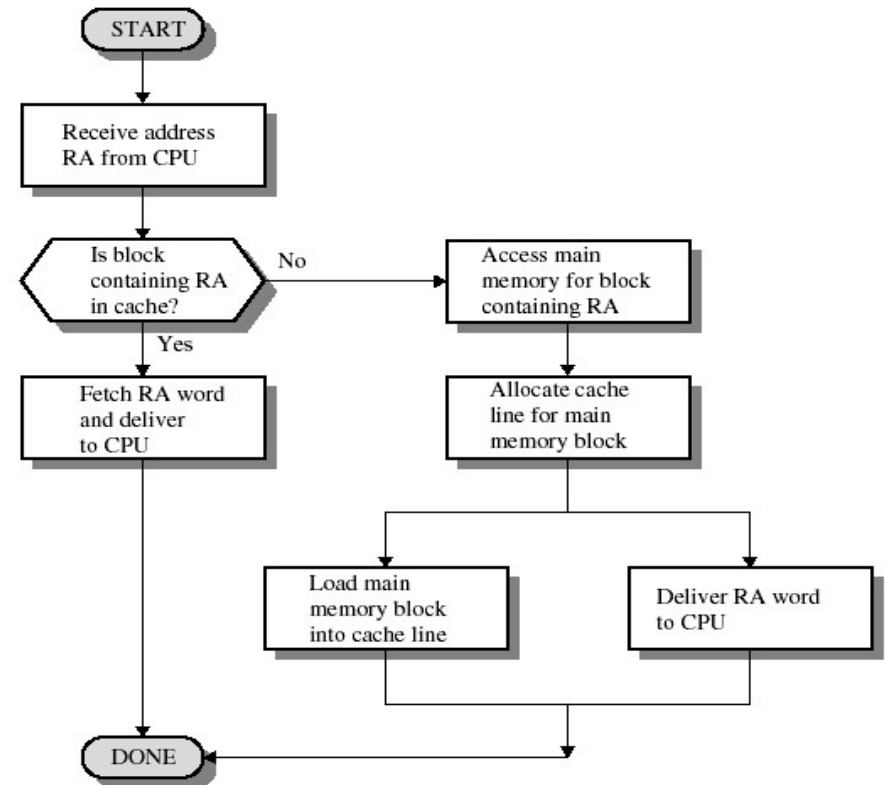
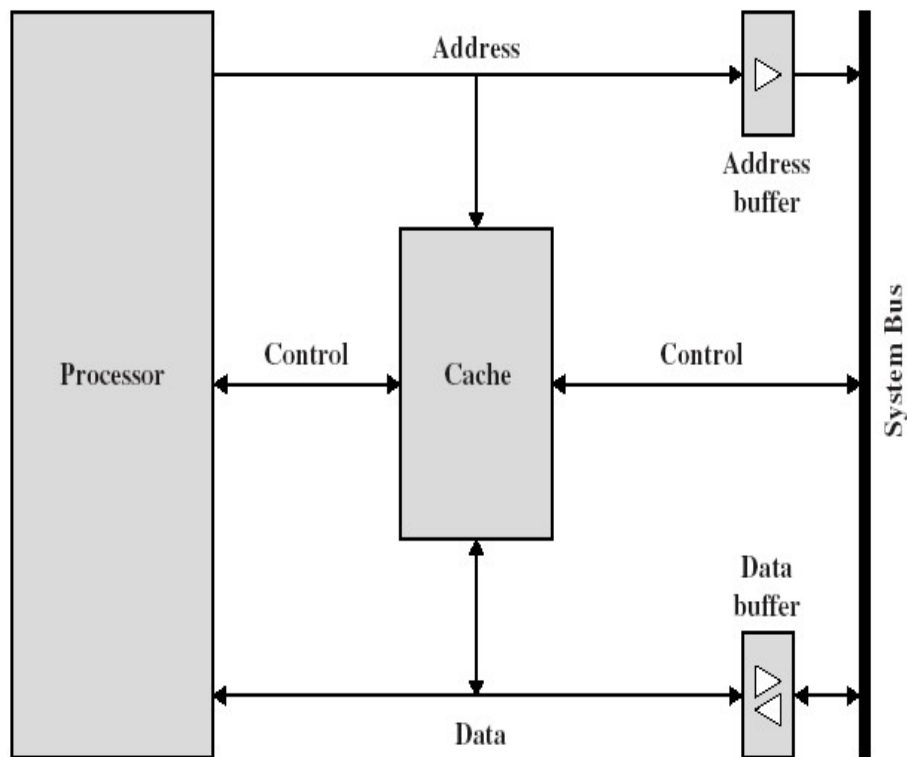
endloop:
    mv a0, t0
    jr ra
```

Memoria cache

- La memoria principale consiste di 2^n byte indirizzabili
- La memoria può essere considerata anche organizzata in **blocchi di k byte** ($M = 2^n/k$ blocchi)
- La cache consiste di C linee di k byte (blocchi)
- $C \ll M$
- Si usano blocchi per il principio di località



Memoria cache



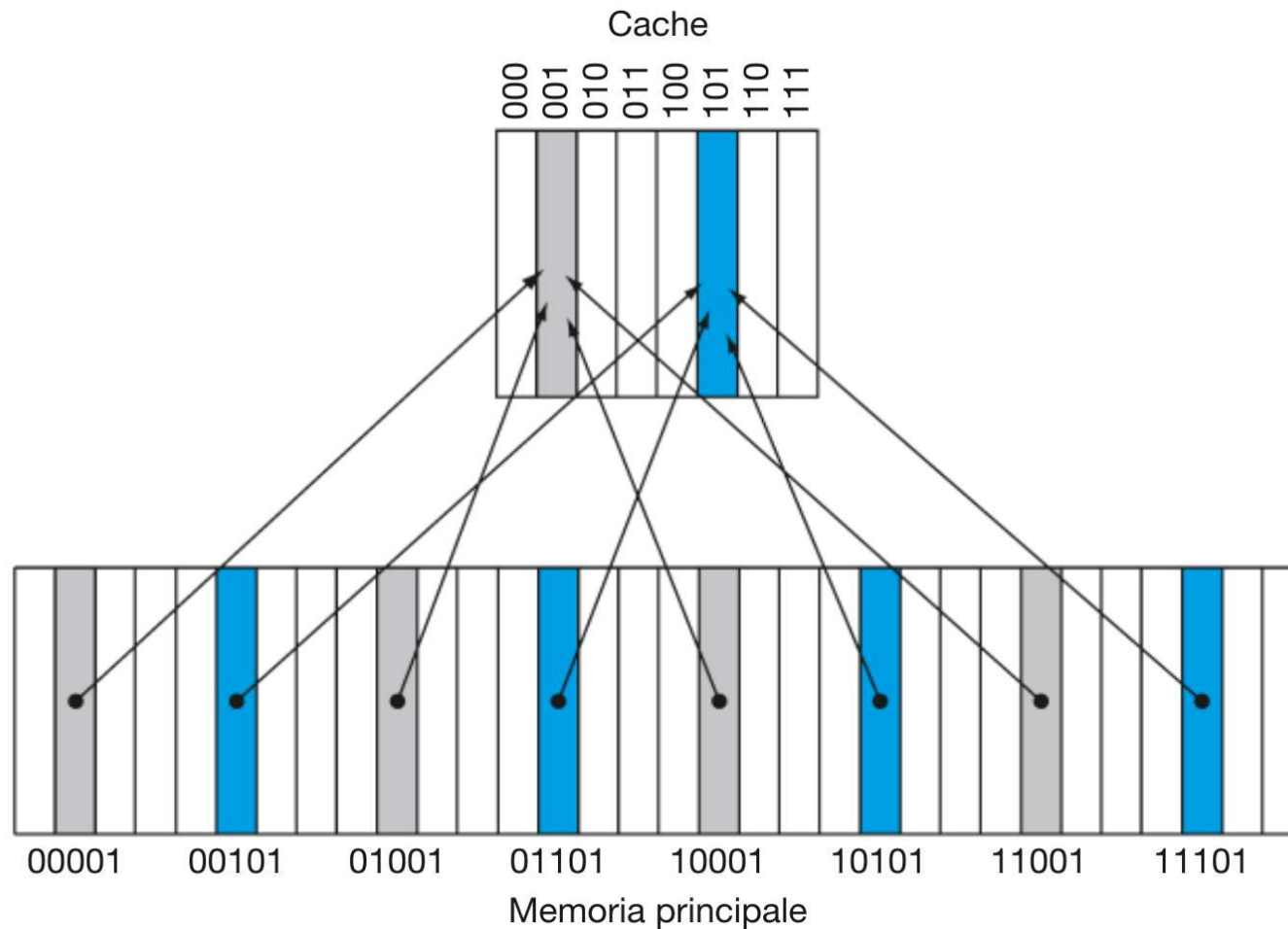
- Se la parola è contenuta nella cache questa è inviata direttamente alla CPU (**hit / hit rate / tempo di hit**)
- Altrimenti il blocco contenente la parola cercata è caricata nella cache e inviata nello stesso tempo alla CPU (**miss della cache / miss rate / penalità di miss**)

Memoria cache

Problemi relativi alla progettazione della memoria cache:

- **Dimensione**: maggiore dimensione implica migliori prestazioni, ma maggiore costo
- **Dimensione della linea di cache** (in una cache di 14 KB è meglio avere 1024 linee di 16 byte o 2048 di 8 byte?)
- **Come tener traccia** di quali parole di memoria sono memorizzate nella cache in un dato momento?
- Istruzioni e dati in una sola cache “**unificata**” o è preferibile avere cache “**specializzate**” per dati e istruzioni (architettura Harvard), in modo da parallelizzare l’accesso alle istruzioni e ai dati?
- **Numero delle cache**: oggi anche tre livelli di cache

Mappatura diretta



- Una cache in cui ad ogni locazione della memoria principale corrisponde una (e una sola) locazione della cache

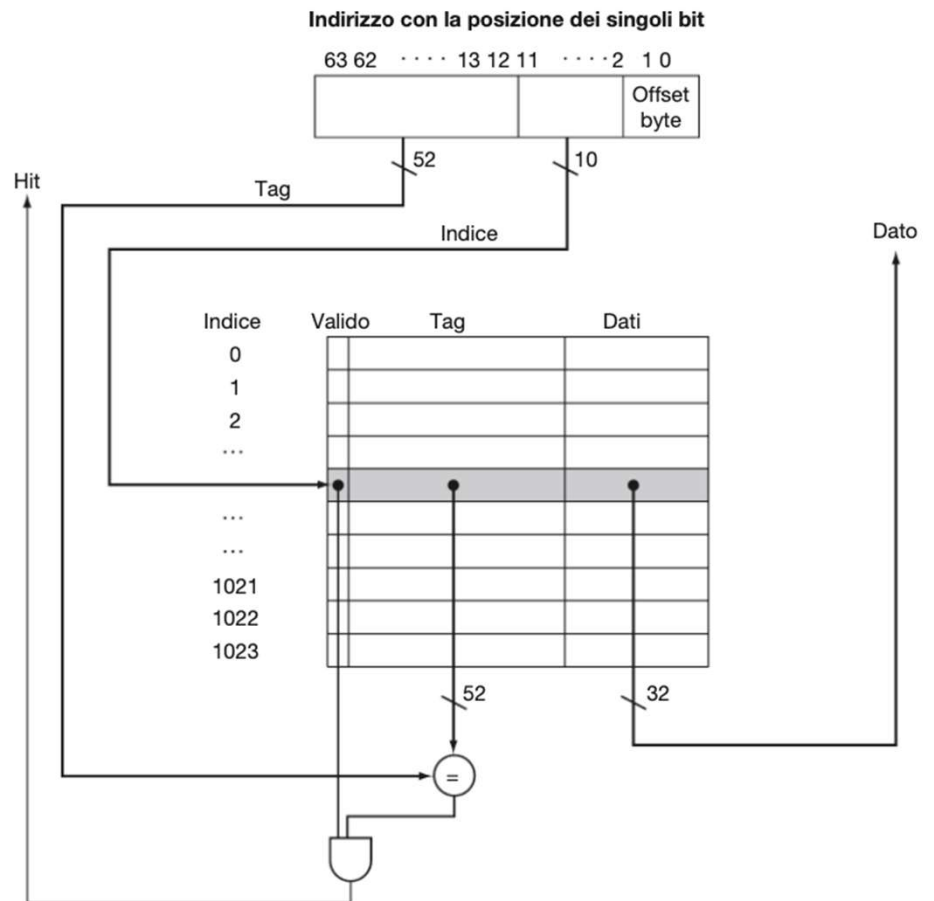
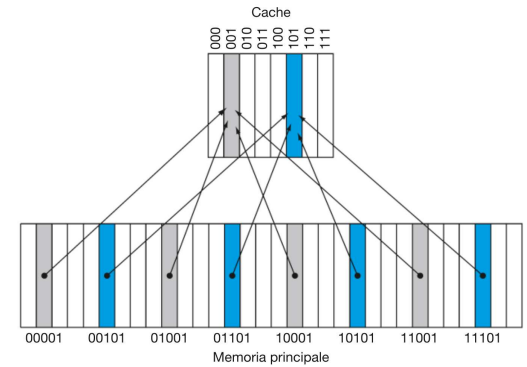
Nella figura: 1 blocco = 1 parola, in pratica blocchi da k byte

(Indirizzo del blocco) modulo (numero di blocchi nella cache)

Mappatura diretta

× Come si capisce se

- ✓ In una linea è presente un dato?
 - ◆ Bit di validità
- ✓ Nella cache è presente il dato ricercato?
 - ◆ tag



(Indirizzo del blocco) modulo (numero di **blocchi** nella cache)

Mappatura diretta

- × Un esempio
 - ✓ Cache a mappatura diretta di otto blocchi
 - ✓ I tre bit meno significativi indicano il numero della linea
 - ✓ Elenco di nove richieste

Indirizzo decimale del dato nella memoria principale	Indirizzo binario del dato nella memoria principale	Hit o miss nell'accesso alla cache	Blocco della cache corrispondente (dove trovare o scrivere il dato)
22	10110 _{due}	Miss (5.9b)	(10 110 _{due} mod 8) = 110 _{due}
26	11010 _{due}	Miss (5.9c)	(11 010 _{due} mod 8) = 010 _{due}
22	10110 _{due}	Hit	(10 110 _{due} mod 8) = 110 _{due}
26	11010 _{due}	Hit	(11 010 _{due} mod 8) = 010 _{due}
16	10000 _{due}	Miss (5.9d)	(10 000 _{due} mod 8) = 000 _{due}
3	00011 _{due}	Miss (5.9e)	(00 011 _{due} mod 8) = 011 _{due}
16	10000 _{due}	Hit	(10 000 _{due} mod 8) = 000 _{due}
18	10010 _{due}	Miss (5.9f)	(10 010 _{due} mod 8) = 010 _{due}
16	10000 _{due}	Hit	(10 000 _{due} mod 8) = 000 _{due}

Mappatura diretta

× Un esempio

Indice	V	Tag	Dati
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	N		
111	N		

a. Lo stato iniziale della cache dopo l'accensione del calcolatore

Indice	V	Tag	Dati
000	N		
001	N		
010	N		
011	N		
100	N		
101	N		
110	S	10 _{due}	Memoria (10110 _{due})
111	N		

b. Dopo avere gestito una miss all'indirizzo (10110_{due})

Indice	V	Tag	Dati
000	N		
001	N		
010	S	11 _{due}	Memoria (11010 _{due})
011	N		
100	N		
101	N		
110	S	10 _{due}	Memoria (10110 _{due})
111	N		

c. Dopo avere gestito una miss all'indirizzo 11010_{due}

Indice	V	Tag	Dati
000	S	10 _{due}	Memoria (10000 _{due})
001	N		
010	S	11 _{due}	Memoria (11010 _{due})
011	N		
100	N		
101	N		
110	S	10 _{due}	Memoria (10110 _{due})
111	N		

d. Dopo avere gestito una miss all'indirizzo 10000_{due}

Mappatura diretta

× Un esempio

Indice	V	Tag	Dati
000	S	10 _{due}	Memoria (10000 _{due})
001	N		
010	S	11 _{due}	Memoria (11010 _{due})
011	S	00 _{due}	Memoria (00011 _{due})
100	N		
101	N		
110	S	10 _{due}	Memoria (10110 _{due})
111	N		

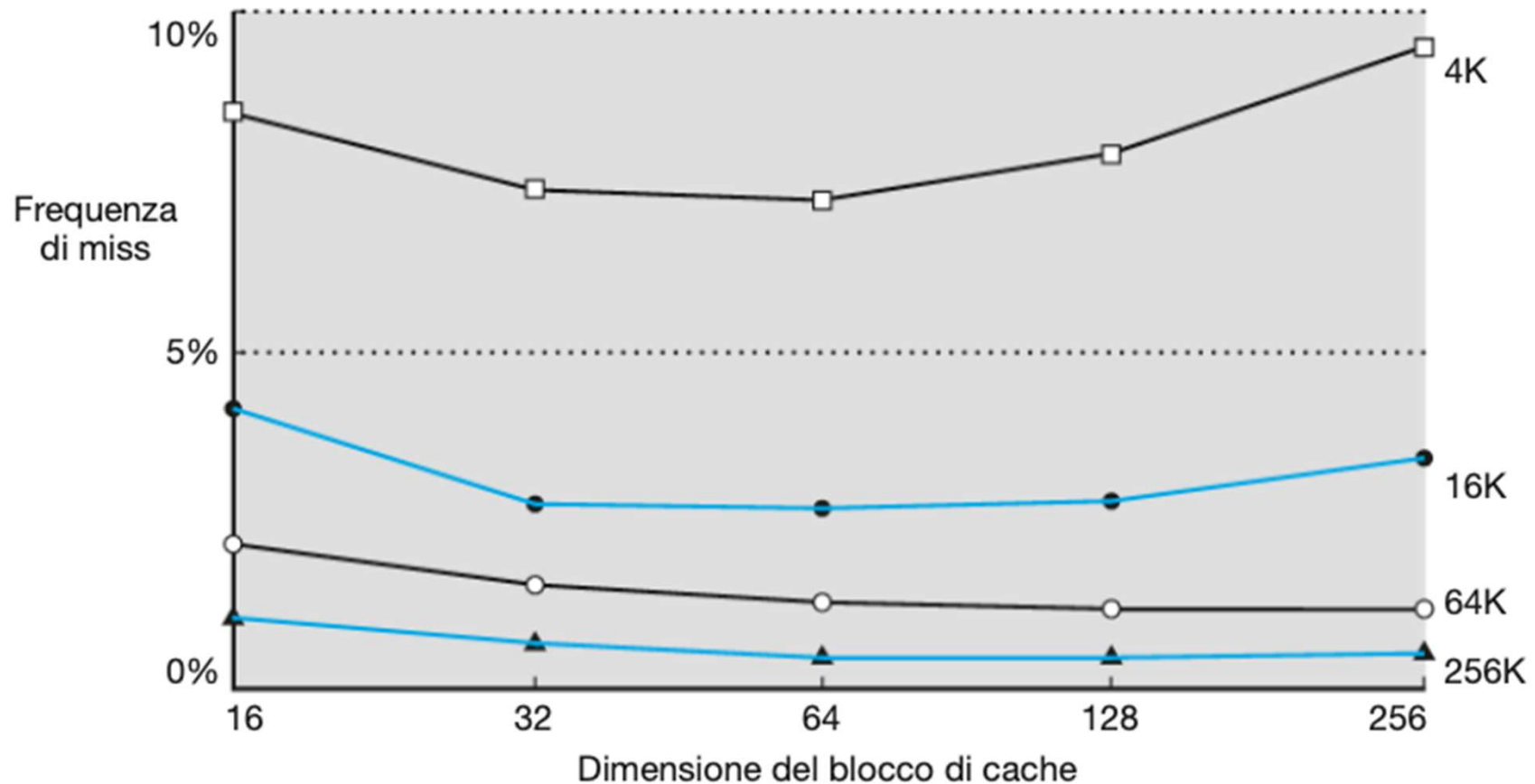
e. Dopo avere gestito una miss all'indirizzo 00011_{due}

Indice	V	Tag	Dati
000	S	10 _{due}	Memoria (10000 _{due})
001	N		
010	S	10 _{due}	Memoria (10010 _{due})
011	S	00 _{due}	Memoria (00011 _{due})
100	N		
101	N		
110	S	10 _{due}	Memoria (10110 _{due})
111	N		

f. Dopo avere gestito una miss all'indirizzo 10010_{due}

Prestazioni di una cache

- × Aumentare la dimensione del blocco non è sempre una buona idea



Completamente associativa

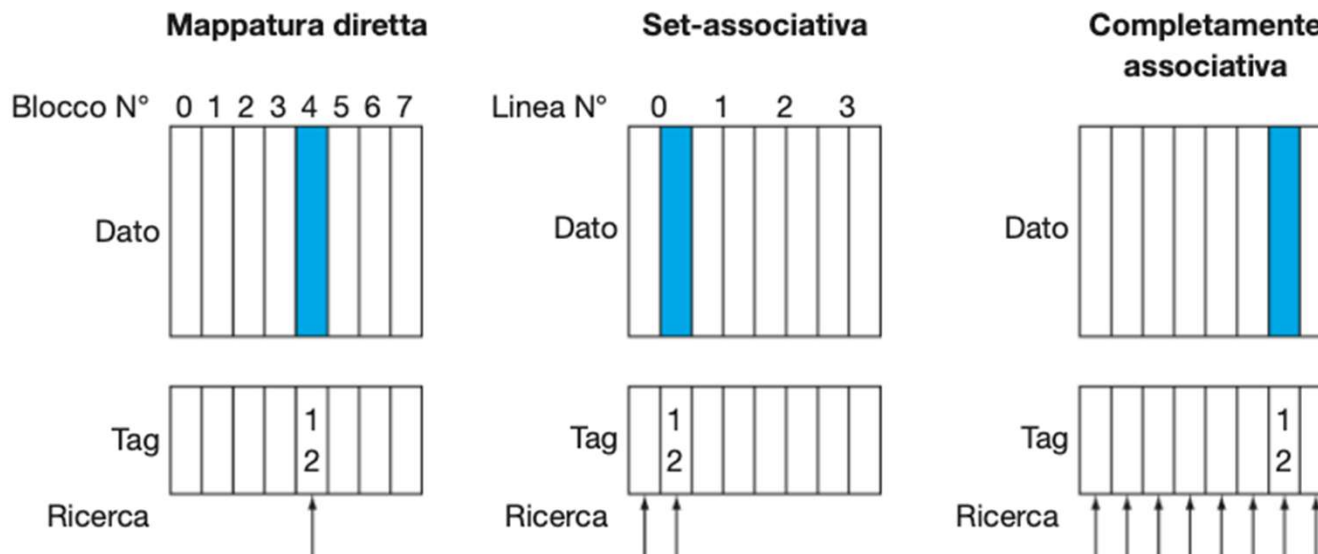
- × Un blocco della memoria principale può essere scritto in una qualsiasi locazione della cache
- × La ricerca va effettuata su tutti gli elementi della cache
 - ✓ Svolta in parallelo attraverso un comparatore (aumentando i costi)
- × Adatta solo per un numero ridotto di blocchi



Cache set-associativa

- × Schema intermedio
- × Ogni linea può contenere più blocchi (numero di vie)
- × Individuata la linea, il blocco viene ricercato all'interno degli slot della linea

(Numero del blocco) modulo (Numero delle **linee** della cache)



Cache set-associativa

Cache set-associativa a una via

(a mappatura diretta)

Blocco	Tag	Dato
0		
1		
2		
3		
4		
5		
6		
7		

Cache set-associativa a due vie

Linea	Tag	Dato	Tag	Dato
0				
1				
2				
3				

Cache set-associativa a quattro vie

Linea	Tag	Dato	Tag	Dato	Tag	Dato	Tag	Dato
0								
1								

Tag: l'informazione per capire se il blocco contiene la parola richiesta

Utilizzo meno recente (LRU):
il blocco sostituito è quello che è rimasto inutilizzato per più tempo

Esempio di prestazione:

Associatività	Frequenza di miss
1	10,3%
2	8,6%
4	8,3%
8	8,1%

- Possiede un numero prestabilito di locazioni alternative (almeno due) nelle quali può essere scritto o letto ogni blocco della memoria principale

(Numero del blocco) modulo (Numero delle linee della cache)

Gestione delle miss

- × La gestione di miss e hit necessita di una modifica all'unità di controllo del processore *
- × Miss della cache (istruzioni)
 1. inviare il valore originale del program counter alla memoria;
 2. comandare alla memoria principale di eseguire un'operazione di lettura e attendere che la memoria completi la lettura;
 3. scrivere la parola che proviene dalla memoria nella posizione opportuna del blocco della cache, aggiornare il campo tag corrispondente scrivendovi i bit più significativi dell'indirizzo presi direttamente dalla ALU e impostare a 1 il bit di validità;
 4. far ripartire l'esecuzione dell'istruzione dall'inizio, ripetendo la fase di fetch, che questa volta troverà l'istruzione all'interno della cache.
- × Miss della cache (dati): sostanzialmente identico

Gestione della scrittura

× Write-through

- ✓ Ad ogni scrittura in cache si modifica il valore anche in memoria
- ✓ Se il dato da scrivere non è presente in cache, viene prima caricato, poi modificato in cache e in memoria
- ✓ Prestazioni non buone (uso di un buffer di scrittura)

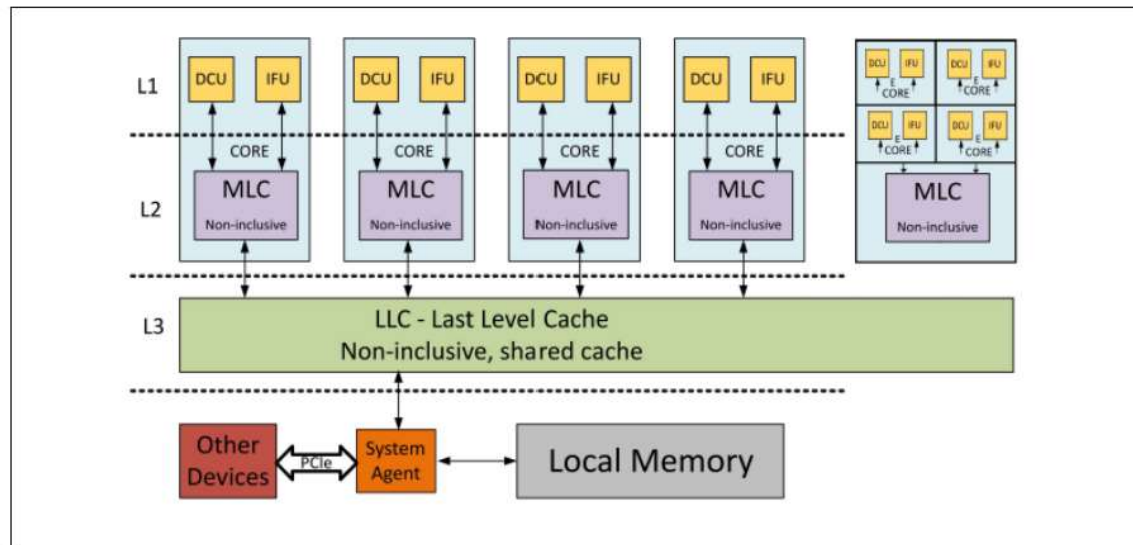
× Write-back

- ✓ Il dato viene scritto solo in cache
- ✓ La scrittura al livello inferiore avviene solo quando il blocco nella cache deve essere rimpiazzato
- ✓ Più complesso da implementare

Memoria cache

12th Generation Intel® Core™ Processors (2022)

Hybrid Cache



NOTES

1. L1 Data cache (DCU) - 48KB (P-core) - 32KB (E-Core)
2. L1 Instruction cache (IFU) - 32KB (P-Core) - 64KB (E-Core)
3. MLC - Mid Level Cache - 1.25MB (P-Core) - 2MB (shared by 4 E-Cores)

P Cores 1st level cache is divided into a data cache (DFU) and an instruction cache (IFU). The processor 1st level cache size is 48KB for data and 32KB for instructions. The 1st level cache is an 12-way associative cache.

E Cores 1st level cache is divided into a data cache (DFU) and an instruction cache (IFU). The processor 1st level cache size is 64KB for data and 32KB for instructions. The 1st level cache is an 8-way associative cache.

The 2nd level cache holds both data and instructions. It is also referred to as mid-level cache or MLC. The P processor 2nd level cache size is 1.25MB and is a 10-way non-inclusive associative cache., 4 E Cores processors share 2MB 2nd level cache and is a 16-way non-inclusive, associative cache.

SiFive Performance P550

Memory System And Caches

The P550 memory system has been tuned for high performance workloads. The instruction memory system consists of a dedicated 32KB 4-way set-associative L1 instruction cache with a line size of 64 bytes. The data memory subsystem has a 4-way set-associative 32KB write-back L1 data cache that supports 64-byte cache lines.

The private L2 cache is a 256KB 8-way set-associative cache with a line size of 64 bytes.

A flexible Level 3 cache can be configured to be either 1MB, 2MB or 4MB, with a multi-cluster option of 8MB.

Memoria cache

Table 4.3 Cache Sizes of Some Processors

Processor	Type	Year of Introduction	L1 cache ^a	L2 cache	L3 cache
IBM 360/85	Mainframe	1968	16 to 32 KB	—	—
PDP-11/70	Minicomputer	1975	1 KB	—	—
VAX 11/780	Minicomputer	1978	16 KB	—	—
IBM 3033	Mainframe	1978	64 KB	—	—
IBM 3090	Mainframe	1985	128 to 256 KB	—	—
Intel 80486	PC	1989	8 KB	—	—
Pentium	PC	1993	8 KB/8 KB	256 to 512 KB	—
PowerPC 601	PC	1993	32 KB	—	—
PowerPC 620	PC	1996	32 KB/32 KB	—	—
PowerPC G4	PC/server	1999	32 KB/32 KB	256 KB to 1 MB	2 MB
IBM S/390 G4	Mainframe	1997	32 KB	256 KB	2 MB
IBM S/390 G6	Mainframe	1999	256 KB	8 MB	—
Pentium 4	PC/server	2000	8 KB/8 KB	256 KB	—
IBM SP	High-end server/ supercomputer	2000	64 KB/32 KB	8 MB	—
CRAY MTA ^b	Supercomputer	2000	8 KB	2 MB	—
Itanium	PC/server	2001	16 KB/16 KB	96 KB	4 MB
SGI Origin 2001	High-end server	2001	32 KB/32 KB	4 MB	—

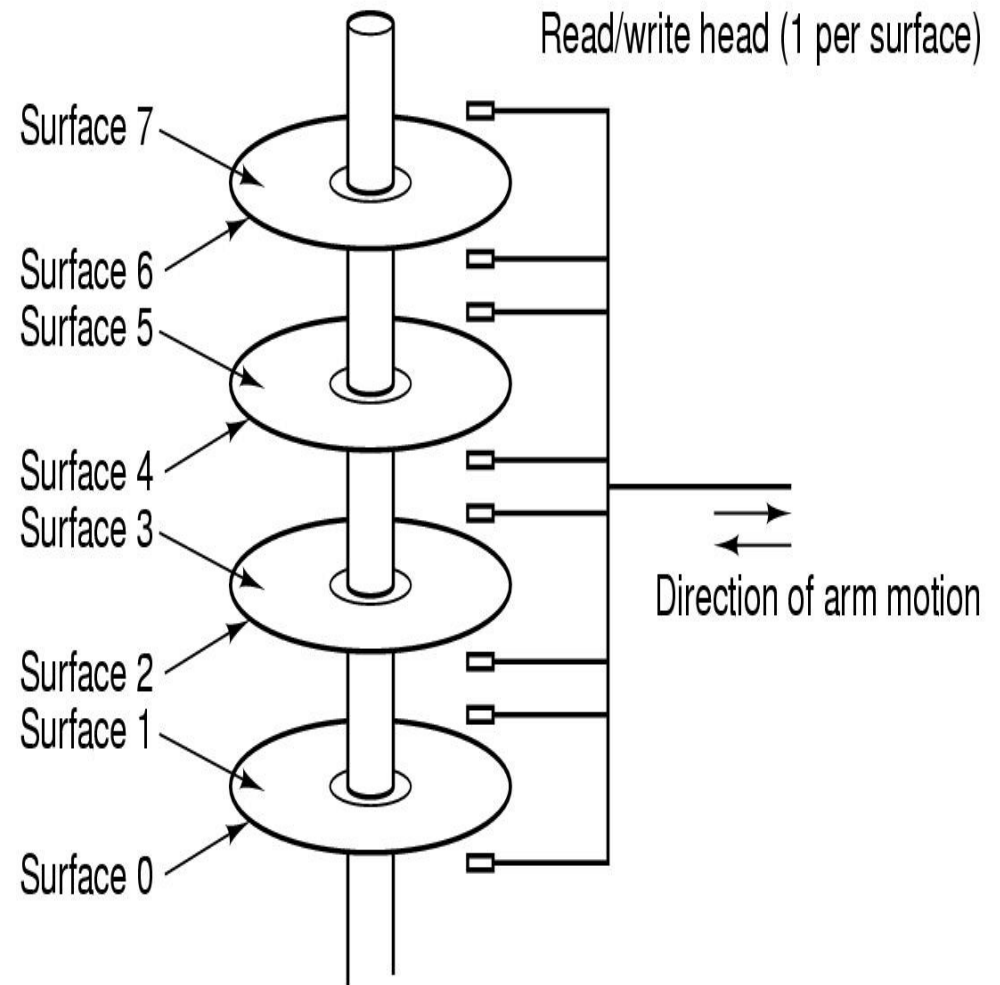
^a Two values separated by a slash refer to instruction and data caches

^b Both caches are instruction only; no data caches

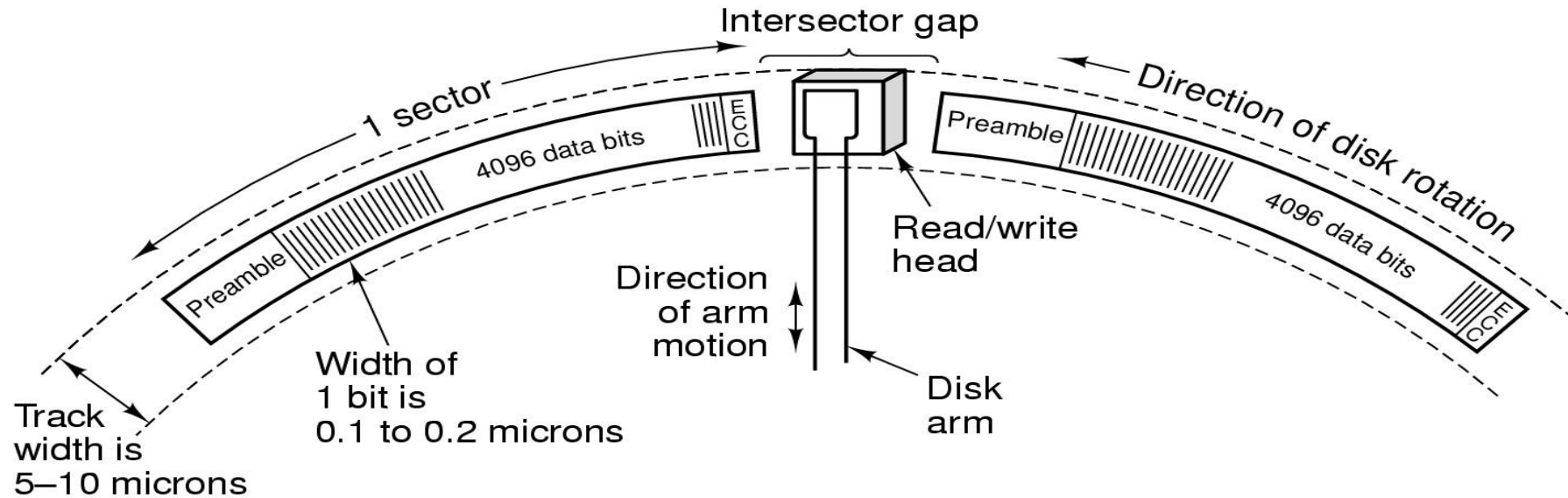
Intel Core i7	PC	2008	32KB/32KB	256KB	4-16MB
Intel Core i9	PC	2019	???	???	16MB (Max)

Memoria secondaria: Dischi magnetici

- Alluminio con rivestimento magnetizzabile
- Accesso dell'ordine dei millisecondi (nanosecondi per i registri!)
- Sigillati in fabbrica
- Controllore del disco
- Organizzazione a cilindri (tracce alla stessa distanza dal centro)
- Tempo di seek
- Latenza di rotazione
- Tempo di trasferimento

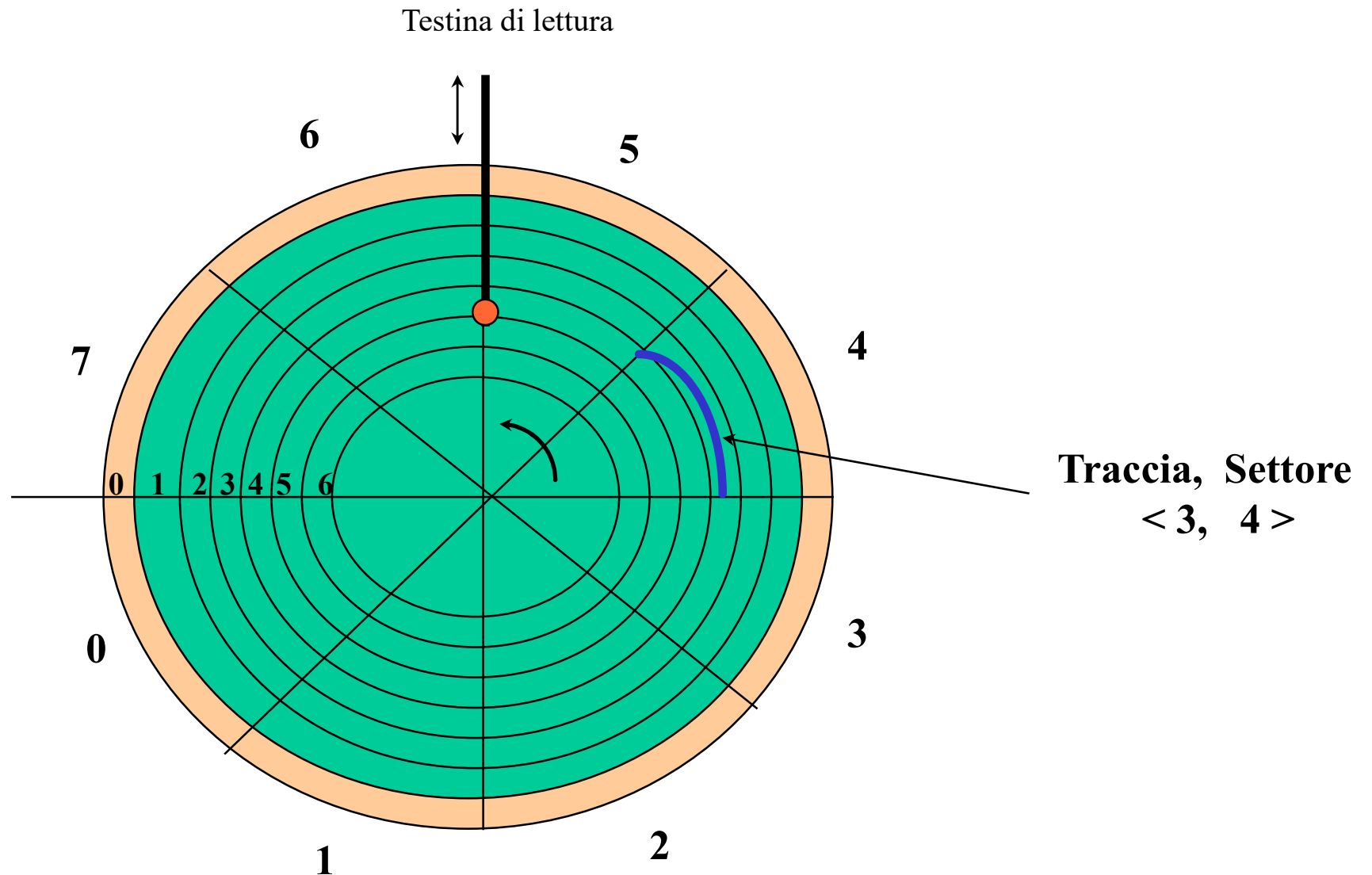


Dischi magnetici



- Ogni traccia è organizzata in settori
- Preambolo per la sincronizzazione della testina
- Insieme dei dati
- Codice correzione degli errori (Hamming o Reed-Salomon)
- Gap tra settori

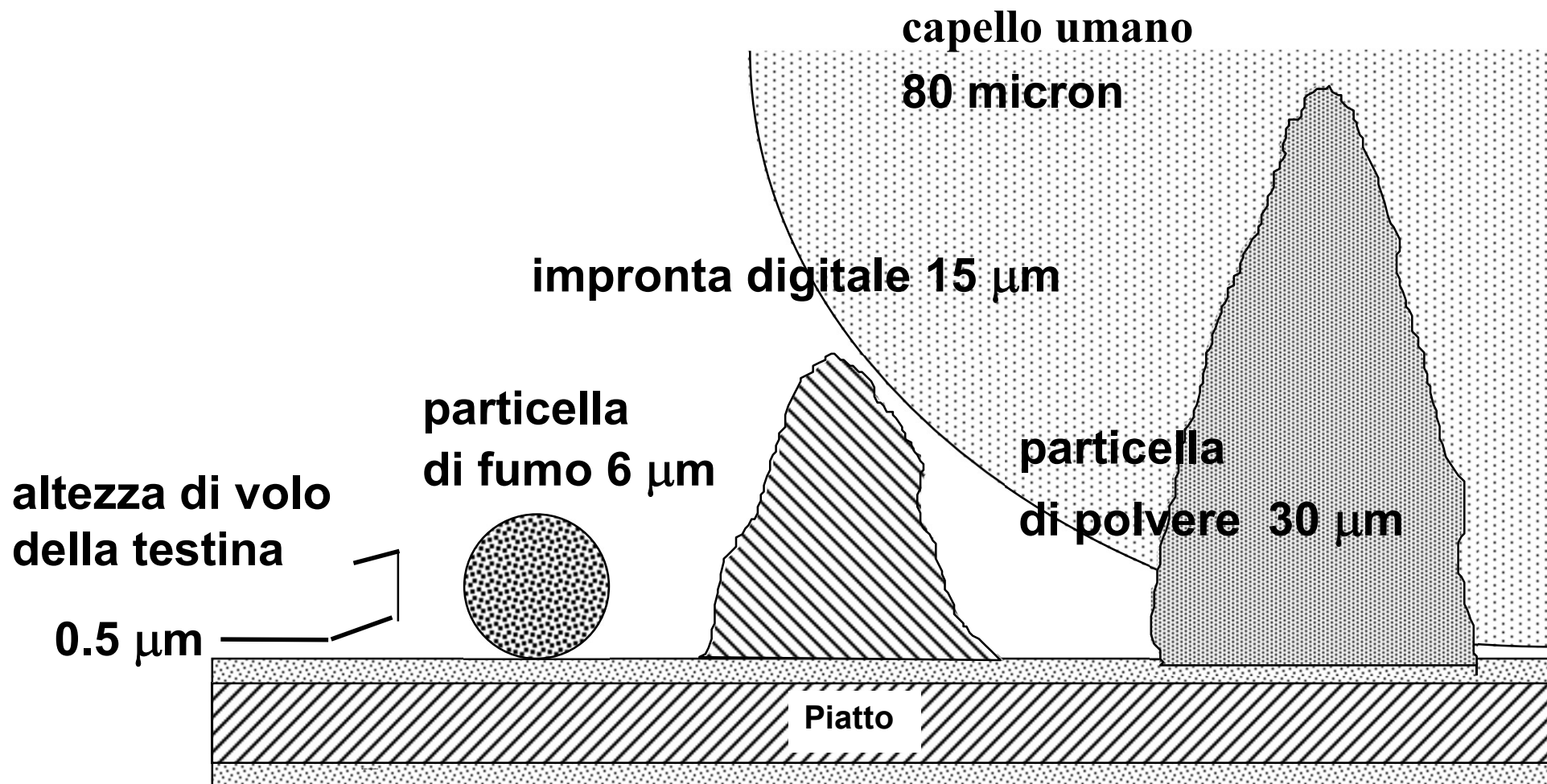
Dischi magnetici



Dischi magnetici

- Per effettuare un'operazione di lettura (scrittura) su un blocco è necessario che la testina si posizioni in corrispondenza dell'indirizzo desiderato
- Il tempo di accesso alle informazioni sul disco è dato dalla somma di tre tempi dovuti a:
 - spostamento della testina in senso radiale fino a raggiungere la traccia desiderata (**seek time**);
 - attesa che il settore desiderato si trovi a passare sotto la testina; tale tempo dipende dalla velocità di rotazione del disco (**latency time**);
 - **tempo di lettura** vero e proprio dell'informazione

I dischi magnetici – hard disk



Dischi magnetici

- Dischi **Winchester**
- **Floppy** disk
- **IDE** (Integrated Drive Electronics, 504 MB, 4 MB/s) anni '80 e **EIDE** (Extended IDE, 128GB, 16,67 MB/s): controllore integrato nel drive
- **ATA-3** (AT Attachement, 33 MB/s), **ATAPI-4** (ATA Packet Interface, 66 MB/s), **ATAPI-6** (100 MB/s): aumento della velocità di trasferimento e dello spazio degli indirizzi
- **ATAPI-7** (*serial* ATA, 150 MB/s): anziché aumentare la velocità di trasferimento aumentando la dimensione del connettore questo standard utilizza un trasferimento seriale.

Dischi magnetici

- . **SCSI** (Small Computer System Interface), 1986, e SCSI-2, 1994: interfaccia che garantisce alta velocità di trasferimento, di fatto è costituita da un bus cui sono attaccati il controllore e fino a 7 dispositivi (hard disk, CD_ROM, scanner...)
- . **RAID** (Redundant Array of Inexpensive/Independent Disk): di fatto è costituita da un insieme di dischi, organizzati a livelli e da un controllore. Un RAID si comporta come un unico disco rispetto al SO, ma il controllore può effettuare operazioni in parallelo, migliorando quindi prestazioni e affidabilità.

Dischi magnetici

- **Diametro:** tra 3 e 9 cm (per i portatili anche < 3 cm)
- **Settore:** in genere almeno 512 byte con alcuni byte di preambolo, di codice per correzione di errori e spazio tra settori.
 - Capacità dei dischi, a volte, include questi byte di supporto (capacità lorda). La capacità netta (solo dati) è, più o meno, un 15% in meno a causa della formattazione (scrittura preamboli, codici e spazi tra settori).
- **Dimensione (larghezza su traccia) di 1 bit:** tra 0.1 e 0.2 micron (milionesimi di metro)
- **Numero tracce:** intorno a 50000 per cm (larghezza di una traccia 200nm)

Dischi magnetici

- **Numero piatti:** tra 1 e 12 (tra 2 e 24 superfici)
- **Capacità lorda per piatto:** fino a 1TB
- **Tempi medio di seek:** tra 5 e 10 ms
- **Velocità di rotazione:** 5400, 7200, 10800, 15000 RPM
(Round Per Minute)