



La gestione delle memorie

Prof. Alberto Borghese
Dipartimento di Informatica
alberto.borghese@unimi.it

Università degli Studi di Milano

Riferimento Patterson: 5.2, 5.4 (blocking), 5.5, 5.10, 5.11



Sommario

Blocking

I codici di errore

Gli altri dispositivi di memoria



Gerarchia di memorie

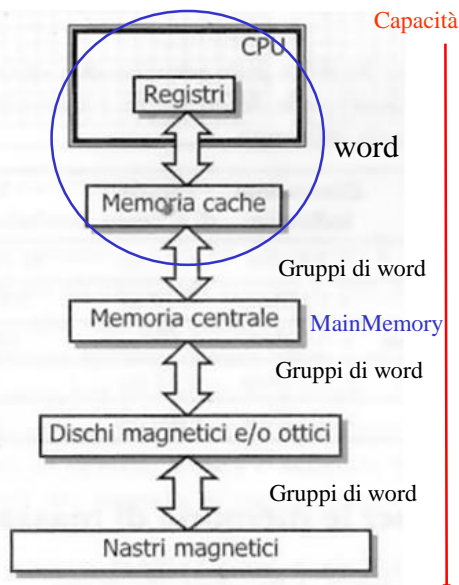


Livelli multipli di memorie con diverse dimensioni e velocità.

Nel livello superiore troviamo un sottoinsieme dei dati del livello inferiore.

Ciascun livello vede il livello inferiore e viceversa.

Cache (memoria nascosta)



Accesso efficiente alle cache: blocking



Blocked algorithms work at each iteration on part of the data or blocks of data.

Obiettivo: aumentare la coerenza temporale =>

- Massimizzare il riutilizzo dei dati presenti in una linea di cache prima di scaricarla.
- Minimizzare le operazioni di scaricamento e ricaricamento di una linea.
- Esempio. Operazioni su matrici vengono scomposte in operazioni su sotto-matrici o blocchi (**blocks**).

		j					
M[i,j]		0	1	2	3	4	5
0							
1							
2							
3							
4							
5							



Il contributo del SW: Blocking



Moltiplicazione di due matrici, $n \times n$ (*Double precision General Matrix Multiply* – DGEMM: $C = A * B$).

- Supponiamo una matrice M , $n \times n$, sia scritta per colonna come vettore, $M[i][j] = M[i+j*n]$.
- Le colonne vengono “impilate” una sopra l’altra in memoria.
- La riga i -esima della matrice $C[i][j]$ viene calcolata come:

```
for (int i=0; i<n; i++)
{ for (int j=0; j<n; j++)
  { double cij = 0; // cij = C[i][j]
    for (int k=0; k<n; k++)
      { cij = cij + A[i+k*n] * B[k+j*n]; // cij = cij + A[i][k]*B[k][j];
        C[i+j*n] = cij;
      }
  }
}
```

Prende gli elementi della riga i -esima di A ($i+k*n$) e li moltiplica per la j -esima colonna di B ($k+j*n$) e mette il risultato in $C[i][j]$.



Blocking e matrici



Vengono letti n elementi di A (la riga), e per ognuna delle colonne di B , altri n elementi, viene poi fatta la somma dei prodotti degli elementi.

Fino a che n è piccolo le 3 matrici stanno in memoria (dati su 1 Byte, $n = 32$, avremo $32 \times 32 \times 8 \times 3$ matrici = 24 Kbyte < 32 Kbyte di una cache L1 (e.g. Core i-7).

Non avremo miss di capacità. Non ci saranno miss di collisione per gli accessi sequenziali. Avremo solo le miss da cold-start, inevitabili.

Ma se la matrice cresce di dimensioni?

B contenuta interamente in memoria => A vengono caricate e scaricate le linee

B non è contenuta interamente in memoria => elementi sia di A che di B vengono caricati e scaricati.

Devo trovare una strategia ottimale di caricamento e scaricamento degli elementi di A e B.

Le operazioni vengono eseguite sequenzialmente su blocchi di matrici, contenuti in memoria per tutta la durata delle operazioni sul blocco.

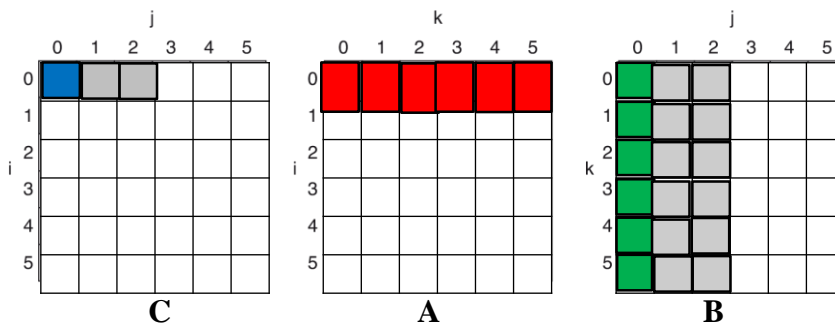


Moltiplicazione delle matrici



Supponiamo che il massimo numero di dati memorizzabili in cache sia 27

$$c(0,0) = a(0,0)*b(0,0)+a(0,1)*b(1,0)+a(0,2)*b(2,0) + a(0,3)*b(0,3)+a(0,4)*b(0,4)+a(0,5)*b(0,5)$$



Per ognuno dei 3 elementi di C, $\{c(0,1), c(0,2), c(0,3)\}$, devo caricare la riga i di A, $(A(i,*))$, e una colonna, j , di B diversa, $B(*,j)$, $0 < j < 2$. Carico una riga di A e carico 3 colonne di B.

Numero di elementi caricati in memoria: $3 + 6 + 3*6 = 27$. Per calcolare tutti gli elementi di C, devo ripetere questa operazione: $36 \text{ elementi} / 3 \text{ elementi/gruppo} = 12 \text{ gruppi}$.

Per calcolare tutti gli elementi di C, carico in totale 12 volte 27 elementi = 324 caricamenti



Blocking all'opera

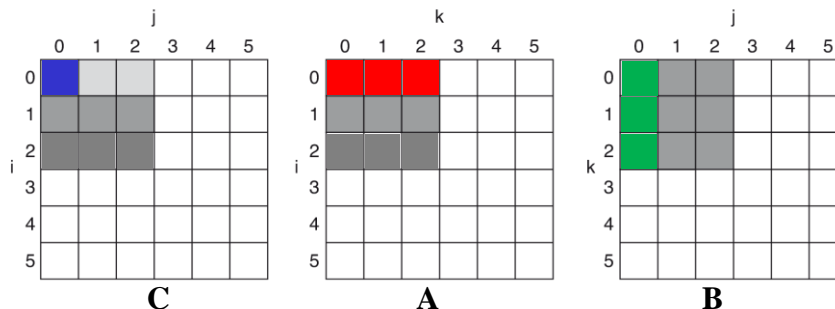


$$c(0,0) = a(0,0)*b(0,0)+a(0,1)*b(1,0)+a(0,2)*b(2,0)$$

..... $(0,1), (0,2), (1,0), (1,1), (1,2), (2,0), (2,1)$

$$c(2,2) = a(2,0)*b(0,2)+a(2,1)*b(1,2)+a(2,2)*b(2,2)$$

Lavoriamo su 3 sotto-matrici



Richiede il massimo numero di dati memorizzati in cache: $9*3 = 27$

Ciascun elemento del primo blocco di A, B, e C $\{c(0,0) - c(2,2)\}$ richiede 1 caricamento (Calcolo parziale di C).

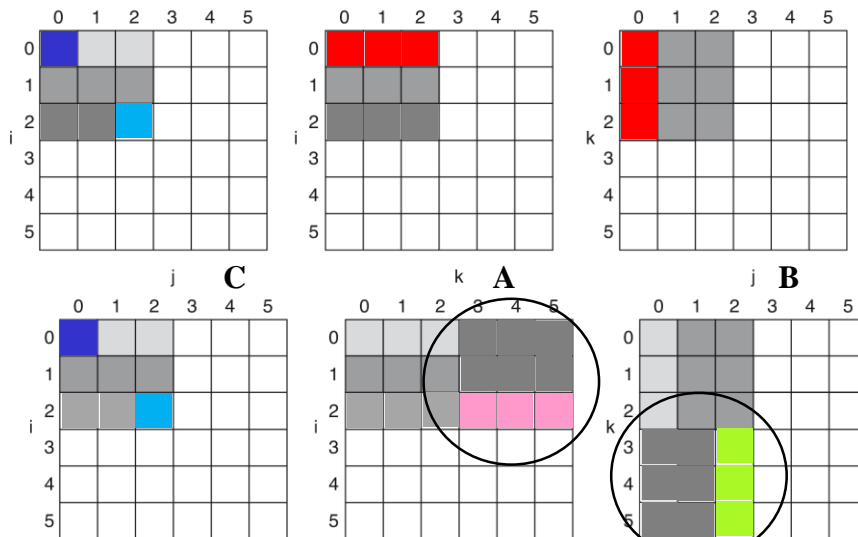


Blocking all'opera



$$\text{Da: } c(0,0) = a(0,0)*b(0,0)+a(0,1)*b(1,0)+a(0,2)*b(2,0) + a(0,3)*b(3,0)+a(0,4)*b(4,0)+a(0,5)*b(5,0)$$

$$\text{A: } c(2,2) = a(2,0)*b(0,2)+a(2,1)*b(1,2)+a(2,2)*b(2,2) + a(2,3)*b(3,2)+a(2,4)*b(4,2)+a(2,5)*b(5,2)$$



A.A. 2021-2022

Carico 4 blocchi di 3x3 elementi di A e B per ogni blocco 3x3 di C

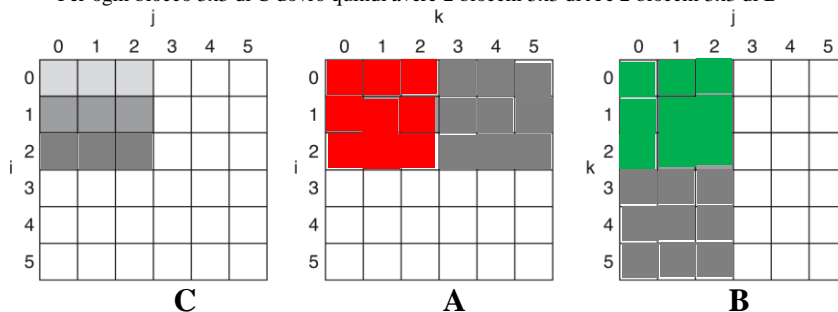
<http://borghese.di.unimi.it/>



Blocking all'opera - calcoli



Per ogni blocco 3x3 di C dovrò quindi avere 2 blocchi 3x3 di A e 2 blocchi 3x3 di B



Quanti caricamenti servono per calcolare il blocco 3x3 di C?

Caricamento 1: blocco di C, A e B = 27 elementi

Caricamento 2: blocco di A[3:5,3:5] e B[3:5,3:5] = $9 \times 2 = 18$ elementi

TOTALE: caricamento di 45 elementi

Ho 4 blocchi 3x3 in C quindi:

TOTALE COMPLESSIVO = $45 \times 4 = 180$ caricamenti

(324 caricamenti senza ottimizzazione mediante il blocking).

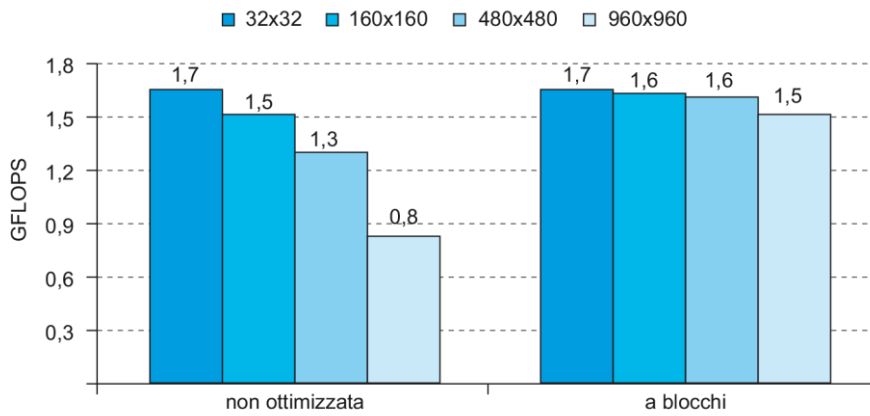
A.A. 2021-2022

10/59

<http://borghese.di.unimi.it/>



Aumento delle prestazioni



Sommario

Il blocking

I codici di errore

Gli altri dispositivi di memoria



Errori della memoria



Le parole di memoria sono costituite da gruppi di bit. Cosa succede se 1 bit commuta “spontaneamente” da 0 a 1?

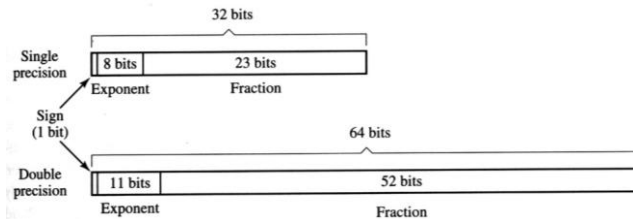
Nome campo	op	rs	rt	indirizzo
Dimensione	6-bit	5-bit	5-bit	16-bit
<code>lw \$t0, 32 (\$s3)</code>	100011	10011	01000	0000 0000 0010 0000

Nome campo	op	rs	rt	indirizzo
Dimensione	6-bit	5-bit	5-bit	16-bit
<code>sw \$t0, 32 (\$s3)</code>	101011	10011	01000	0000 0000 0010 0000

Se commuta il bit 29 di un’istruzione, una load viene tramutata in una store!
La commutazione di altri bit può fare leggere / scrivere il registro sbagliato, può fare accedere a un’indirizzo con un offset sbagliato.

Se commuta il bit 31 (63) il numero cambia segno. Altre commutazioni cambiano il numero FP in modo più o meno significativo.

A.A. 2021-2022



Come avere una memoria affidabile



Gerarchia

Parallelismo

+

Ridondanza

- Codice di correzione degli errori
- Dischi RAID

A.A. 2021-2022

14/59

<http://borghese.di.unimi.it/>



ECC – Error Correction Codes



- Errori dovuti a malfunzionamenti HW o SW.
 - Date le dimensioni delle memorie (**10¹¹ celle**) la probabilità d'errore non è più trascurabile.
 - Per applicazioni sensibili, è di fondamentale importanza gestirli.
- **Codici rivelatori d'errore**
 - Es: codice di parità su 1 bit.
 - Consente di individuare errori singoli in una parola.
 - Non consente di individuare su quale bit si è verificato l'errore.
- **Codici correttori d'errore (error-correcting codes – ECC)**
 - Consentono anche la correzione degli errori.
 - Richiedono più bit per ogni dato (più ridondanza)
 - Per la correzione di 1 errore per parole e l'individuazione di 2 errori, occorrono $\log_2(N)$ bit (codice di Hamming).



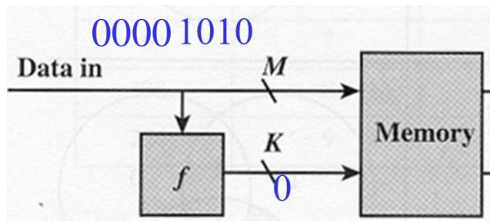
Codice rilevatore di errore: il codice di parità



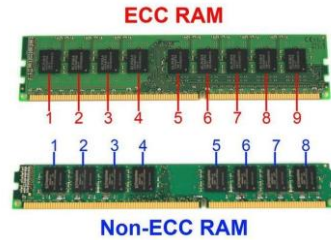
- **Es: Bit di parità (even):**
 - aggiungo un bit ad una sequenza in modo da avere un n. **pari (even)** di "1"
`0000 1010 0` ← **bit di parità (il numero di «1» sarà sempre pari)**
`0001 1010 1`
 - Un errore su uno dei bit porta ad un n. **dispari** di "1" e a un bit di parità a «1»
- Prestazioni del codice di parità su 1 bit
 - mi accorgo dell'errore
 - **rivelo errori singoli**
 - **COSTO: 1 bit aggiuntivo ogni 8** → $9/8 = +12,5\%$
 - ma non so dove sia l'errore.
 - **non correggo**



Scrittura in memoria



Chip che supportano la parità su 1 bit. MM e Cache.



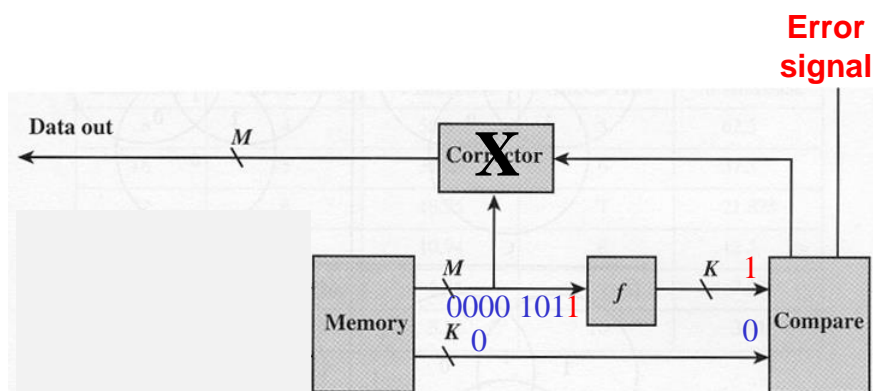
Invio il dato alla memoria e in parallelo invio anche il bit di parità -> 9 bit

f può essere ad esempio il codice di parità e *K* il bit di parità



Lettura dalla memoria

L'errore si identifica in lettura





Codici correttori d'errore



- **Es: Codice a ripetizione**
 - Ripeto ogni singolo bit della sequenza originale per altre 2 volte → triplico ogni bit
 $0 \ 00 \ 1 \ 11 \ 1 \ 11 \ 0 \ 00 \ 1 \ 11 \ 0 \ 00 \ 0 \ 00 \ 1 \ 11 \ \dots$
 - Un errore su un bit di ciascuna terna può essere corretto:
 $000 \rightarrow 010 \rightarrow 0\underline{00}$
 $111 \rightarrow 110 \rightarrow 1\underline{11}$
- Prestazioni del codice
 - **rivelo e correggo** errori singoli su **TUTTI** i bit.
 - **COSTO**: 2 bit aggiuntivi ogni 1 → $3/1 = +200\%$



Definizioni



- **Distanza di Hamming, d** (tra 2 sequenze di N bit)
 - il numero di cifre differenti, giustapponendole
 01001000
 $01000010 \rightarrow d = 2$
- **Distanza minima** di un codice, d_{MIN}
 il valor minimo di d tra tutte le coppie di sequenze valide di un codice
- **Capacità di rivelazione** di un codice: $t = d_{MIN} - 1$
- **Capacità di correzione** di un codice: $r = (d_{MIN} - 1) / 2$
- **Esempi:**
 - Codice a bit di **parità su 1 bit**: $d_{MIN} = 2 \rightarrow t = 1, r = 0$
 $0000 \ 1010 \ 0$ **Distanza minima: commuta 1 bit ->**
 $0000 \ 1011 \ 1$ commuta anche il bit di parità
 - Codice a **ripetizione (3,1)**: $d_{MIN} = 3 \rightarrow t = 2, r = 1$
 $000 \ 000 \ 111 \ 111$ **Distanza minima: commuta 1 bit ->**
 $000 \ 000 \ 111 \ 000$ commutano anche i 2 bit di ripetizione



Come calcolare il codice ECC Hamming



Filosofia: ciascun bit di parità è responsabile della parità di un certo gruppo di bit del dato.

- 1) Enumeriamo i bit partendo da sinistra: ($d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8$).
- 2) Segnare le posizioni potenze di 2 come posizione dei bit di parità (1, 2, 4, 8, 16...)
- 3) I bit dei dati vanno inseriti in tutti gli altri bit (3, 5, 6, 7, 9, 10, 11, 12, ..., 13, 14, 15, 17, ...)
- 4) La posizione del bit di parità determina i gruppi di bit dati di cui quel bit di parità è responsabile.

Esempio

Dato: 1 0 0 1 1 0 1 0 = 89 in base 10
d1 d2 d3 d4 d5 d6 d7 d8

Bit #	1	2	3	4	5	6	7	8	9	10	11	12
Semantica:	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Contenuto:	X	X	1	X	0	0	1	X	1	0	1	0



Calcolo del Bit p1



Bit # (dec)	1	2	3	4	5	6	7	8	9	10	11	12
Bit # (bin)	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
Semantica:	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Contenuto:	0	X	1	X	0	0	1	X	1	0	1	0

Considero i bit di dato la cui posizione contiene un 1 nel bit 2^0 (bit meno significativo): ~~X~~ 3, 5, 7, 9, 11: d1, d2, d4, d5, d7 = 1 0 1 1 1

Conto 4 «1» -> parità -> bit di parità pari = p1 = 0



Calcolo del Bit p2

Bit # (dec)	1	2	3	4	5	6	7	8	9	10	11	12
Bit # (bin)	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
Semantica:	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Contenuto:	0	1	1	X	0	0	1	X	1	0	1	0

Considero i bit di dato la cui posizione contiene un 1 nel bit 2¹: ~~X~~ 3, 6, 7, 10, 11: d1, d3, d4, d6, d7 = 1 0 1 0 1

Conto 3 «1» -> disparità -> bit di parità pari = p2 = 1



Calcolo del Bit p4

Bit # (dec)	1	2	3	4	5	6	7	8	9	10	11	12
Bit # (bin)	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100
Semantica:	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Contenuto:	0	1	1	1	0	0	1	X	1	0	1	0

Considero i bit di dato la cui posizione contiene un 1 nel bit 2²: ~~X~~ 5, 6, 7, 12: d2, d3, d4, d8 = 0 0 1 0.

Conto 1 «1» -> disparità -> bit di parità pari = p4 = 1



Calcolo del Bit p8

Bit # (dec) 1 2 3 4 5 6 7 8 9 10 11 12
 Bit # (bin) 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100

Semantica: p1 p2 d1 p4 d2 d3 d4 p8 d5 d6 d7 d8
 Contenuto: 0 1 1 1 0 0 1 0 1 0 1 0

Considero i bit di dato la cui posizione contiene un 1 nel bit 2³: ~~X~~ 9, 10, 11, 12: d5, d6, d7, d8 = 1 0 1 0.

Conto 2 «1» -> parità -> bit di parità pari = p8 = 0



Come calcolare il codice ECC Hamming

Bit position	1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X		X		X		X		X
	p2		X	X			X	X			X	X
	p4				X	X	X	X				X
	p8								X	X	X	X

Esempio

Dato: 1 0 0 1 1 0 1 0
 d1 d2 d3 d4 d5 d6 d7 d8

d1, d2, d4, d5, d7 -> p1
 d1, d3, d4, d6, d7 -> p2
 d2, d3, d4, d8 -> p4
 d5, d6, d7, d8 -> p8

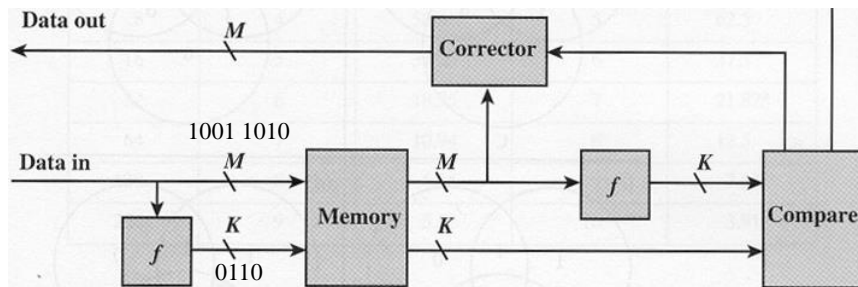
Bit # 1 2 3 4 5 6 7 8 9 10 11 12
 Semantica: p1 p2 d1 p4 d2 d3 d4 p8 d5 d6 d7 d8
 Contenuto: 0 1 1 1 0 0 1 0 1 0 1 0



Scrittura del dato in memoria

Dato: 1 0 0 1 1 0 1 0
d1 d2 d3 d4 d5 d6 d7 d8

Bit #	1	2	3	4	5	6	7	8	9	10	11	12
Semantica:	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Contenuto:	0	1	1	1	0	0	1	0	1	0	1	0

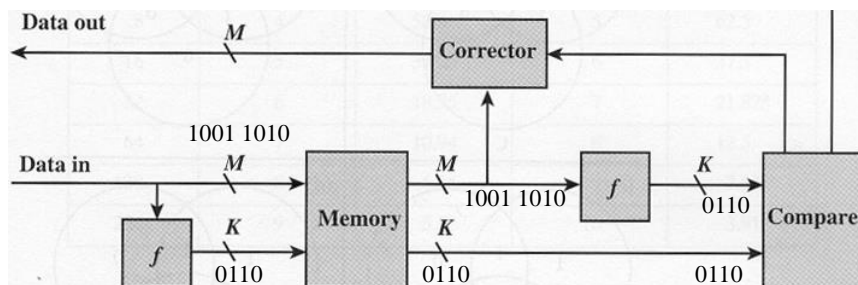


Lettura del dato dalla memoria

Dato: 1 0 0 1 1 0 1 0
d1 d2 d3 d4 d5 d6 d7 d8

Bit #	1	2	3	4	5	6	7	8	9	10	11	12
Semantica:	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Contenuto:	0	1	1	1	0	0	1	0	1	0	1	0

No error





Errore su un bit: lettura

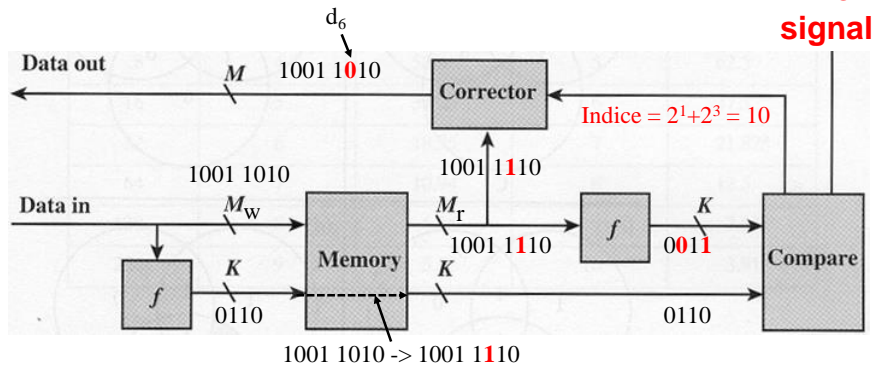


Dato: 1 0 0 1 1 **1** 1 0
d1 d2 d3 d4 d5 d6 d7 d8

Errore nel bit d6, posizione 10_{dec} -> modifica p2 e p8

Bit # (dec) 1 2 3 4 5 6 7 8 9 10 11 12
Bit # (bin) 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100

Semantica: p1 p2 d1 p4 d2 d3 d4 p8 d5 d6 d7 d8
Contenuto: 0 0 1 1 0 0 1 1 1 1 1 0



A.A. 2021-2022

29/59

<http://borghese.di.unimi.it/>



Errore su 2 bit: lettura

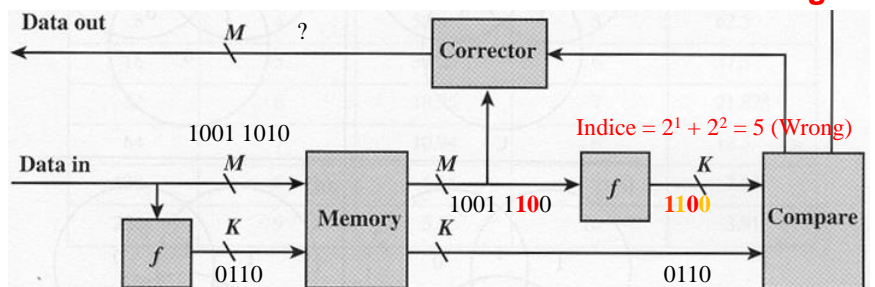


Dato: 1 0 0 1 1 **1 0** 0
d1 d2 d3 d4 d5 d6 d7 d8

Errore nel bit d6, posizione 10 -> modifica p2, p4 e p8
Errore nel bit d7, posizione 11 -> modifica p1, p2 e p8

Bit # (dec) 1 2 3 4 5 6 7 8 9 10 11 12
Bit # (bin) 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100

Semantica: p1 p2 d1 p4 d2 d3 d4 p8 d5 d6 d7 d8
Contenuto: 1 1 1 0 0 0 1 0 1 1 0 0



A.A. 2021

Rilevo che c'è stato errore, ma non sono in grado di correggere un doppio errore .di.unimi.it/



Errore su 1 bit di parità: lettura



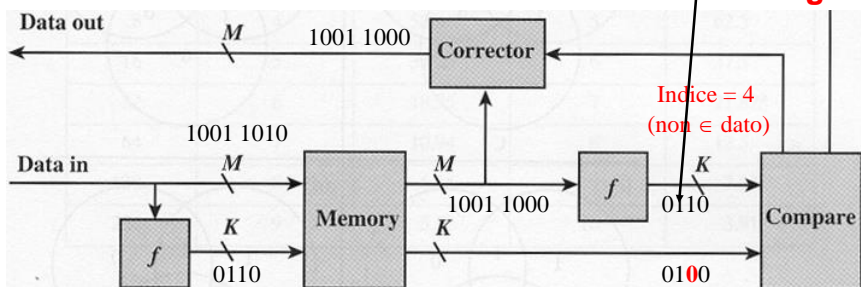
Dato: 1 0 0 1 1 0 1 0
d1 d2 d3 d4 d5 d6 d7 d8

Errore nel bit p4, posizione 4 -> nessuna modifica nei bit di parità associati alla parte dati

Bit # (dec) 1 2 3 4 5 6 7 8 9 10 11 12
Bit # (bin) 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100

Semantica: p1 p2 d1 p4 d2 d3 d4 p8 d5 d6 d7 d8
Contenuto: 0 1 1 0 0 0 1 0 1 0 1 0

Il bit 4 è un bit di parità, il dato non deve essere corretto



Come calcolare il codice ECC Hamming



- 1) Enumeriamo i bit partendo da sinistra: (d₁, d₂, d₃, d₄, d₅, d₆, d₇, d₈).
- 2) Segnare le posizioni potenze di 2 come posizione dei bit di parità (1, 2, 4, 8, 16...)
- 3) I bit dei dati vanno inseriti in tutti gli altri bit (3, 5, 6, 7, 9, 10, 11, 12, ..., 13, 14, 15, 17, ...)
- 4) La posizione del bit di parità determina di quali bit sarà responsabile (p₁ di tutti i bit che hanno indirizzo che termina con 1: i numeri dispari; p₂ di tutti i bit che hanno il secondo bit da destra = 1: 2,3 - 10,11; 6,7 - 110,111; 10,11 - 1010,1011,...).

Bit position	1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
Parity bit coverage	p1	X		X	X		X		X		X	
	p2		X	X		X	X			X	X	
	p4				X	X	X					X
	p8							X	X	X	X	X

Le quattro cifre p1p2p3p4 ci dicono dove c'è un errore eventualmente e quindi ci dà la possibilità di correggerlo.

Con un parola a 12 bit possiamo correggere errori su dati a 8 bit.



Dimensione di codici ECC

- Convieni applicare ECC a parole più lunghe possibile → aggiungo meno ridondanza → maggiore efficienza del codice
 - A costo di complessità maggiori di codifica/decodifica
 - La codifica avviene in parallelo sui diversi bit (indipendenza).

Data Bits	Single-Error Correction		Single-Error Correction/ Double-Error Detection	
	Check Bits	% Increase	Check Bits	% Increase
8	4	50	5	62.5
16	5	31.25	6	37.5
32	6	18.75	7	21.875
64	7	10.94	8	12.5
128	8	6.25	9	7.03
256	9	3.52	10	3.91

E' conveniente avere parole di memoria lunghe => compromesso



Sommario

Gestione della memoria

I codici di errore

Gli altri dispositivi di memoria



Gerarchia di memorie



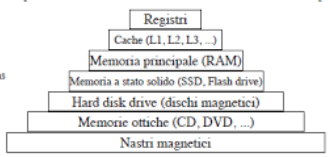
Livelli multipli di memorie con diverse dimensioni e velocità.

Nel livello superiore troviamo un sottoinsieme dei dati del livello inferiore.

Ciascun livello vede il livello inferiore e viceversa.

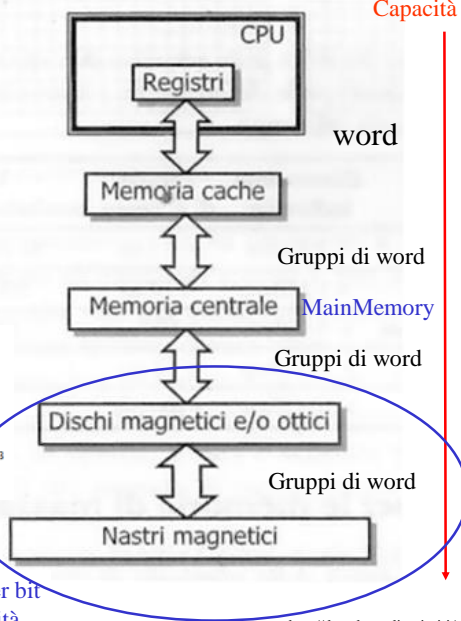
Tempo di accesso tipico

< 1 ns
1-2 ns
10-80 ns
0,05-0,5 ms
5-20 ms
200 ms
100 s



Capacità tipica

< 1 KB
64 KB - 64 MB
512 MB - 4 GB
4 GB - 256 GB
80 GB - 2TB
700 MB - 50 GB
20 GB /TB



Dischi



- Consentono di memorizzare dati in modo **non volatile**.
 - Dischi a stato solido
 - Dischi magnetici
 - Dischi ottici



Dischi a stato solido (Flash memory)



- EEPROM (memoria cancellabile elettronicamente: tempi di lettura e scrittura molto diversi)
- Mantiene l'informazione attraverso iniezione di carica nel gate di NAND (NOR) che sostituisce i condensatori delle DRAM.
- Used in portable devices and in hybrid systems



SanDisk Extreme Pro SSD - 2020

	240GB	480GB	960GB
Seq. Lettura (fino a)	550 MB/s	550 MB/s	550 MB/s
Seq. Scrittura (fino a)	520 MB/s	515 MB/s	515 MB/s
Rnd. Lettura (fino a)	100K IOPS	100K IOPS	100K IOPS
Rnd. Write (fino a)	90K IOPS	90K IOPS	90K IOPS

Characteristics	Kingston SecureDigital (SD) SD4/8 GB	Transend Type I CompactFlash TS16GCF133	RIDATA Solid State Disk 2.5 Inch SATA
Formatted data capacity (GB)	8	16	32
Bytes per sector	512	512	512
Data transfer rate (read/write MB/sec)	4	20/18	68/50
Power operating/standby (W)	0.66/0.15	0.66/0.15	2.1/—
Size: height x width x depth (inches)	0.94 x 1.26 x 0.08	1.43 x 1.68 x 0.13	0.35 x 2.75 x 4.00
Weight in grams (454 grams/pound)	2.5	11.4	52
Mean time between failures (hours)	> 1,000,000	> 1,000,000	> 4,000,000
GB/cu. in., GB/watt	84 GB/cu.in., 12 GB/W	51 GB/cu.in., 24 GB/W	8 GB/cu.in., 16 GB/W
Best price (2008)	~ \$30	~ \$70	~ \$300

IOPS = Input/Output per second

37/59

<http://borghese.di.unimi.it/>



Vantaggi e svantaggi



Pro

Latenza minore di 100-1000 volte dei dischi
Consuma meno ed è più resistente agli urti.

Contra

Wear out (usura) → **wear leveling** (uniformità delle scritture)

Flash introdotte come memorie di boot per rendere il boot più veloce e nei dispositivi mobili quali MP3, telefonini... Nei dispositivi portatili tende a sostituire le memorie a disco rigido.

Si va verso 1,000,000 scritture

Characteristics	NOR Flash Memory	NAND Flash Memory
Typical use	BIOS memory	USB key
Minimum access size (bytes)	512 bytes	2048 bytes
Read time (microseconds)	0.08	25
Write time (microseconds)	10.00	1500 to erase + 250
Read bandwidth (MBytes/second)	10	40
Write bandwidth (MBytes/second)	0.4	8
Wearout (writes per cell)	100,000	10,000 to 100,000
Best price/GB (2008)	\$65	\$4

A.A. 2021-2022

38/59

<http://borghese.di.unimi.it/>

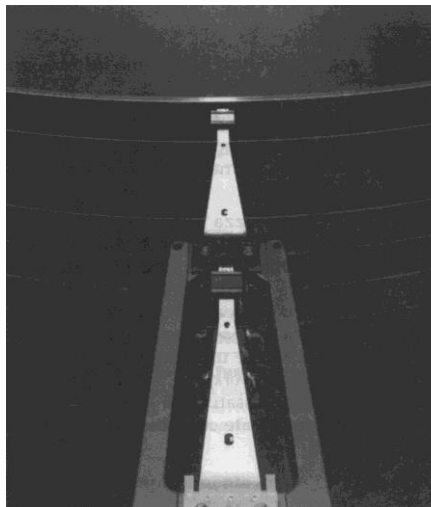


Hard disk magnetico



I dati sono letti/scritti mediante una testina.

- I dischi magnetici sono di due tipi principali:
 - hard disk
 - floppy disk (messi in commercio da Apple e Tandy nel 1978). In precedenza esistevano solamente le cassette magnetiche, a loro volta evoluzione dei nastri magnetici immessi sul mercato nel 1934 in Germania dalla IG Farben, ora Basf, per un magnetofono AEG.
- Costituiti da un insieme di piatti rotanti (da 1 fino a 25) ognuno con due facce, di diametro che va da 2,5cm a 10cm.
- La pila dei piatti viene fatta ruotare alla stessa velocità (5,400 – 15,000 rpm = revolutions per minute)
- Esiste una testina per ogni faccia.
- Le testine si muovono in modo solido.
- Su ciascun disco sono posizionate le **tracce** che contengono i dati, su circonferenze concentriche.
- L'insieme delle tracce di ugual posto su piatti diversi è chiamato **cilindro**.
- La quantità di dati che possono essere memorizzati per traccia dipende dalla qualità del disco.



Hard disk – lettura / scrittura



- Per leggere/scrivere informazioni sono necessari tre passi:
 - la testina deve essere posizionata sulla traccia corretta;
 - il settore corretto deve passare sotto la testina;
 - i dati devono essere letti o scritti.
- **Tempo di seek (ricerca):** tempo per muovere la testina sulla traccia corretta.
- **Tempo di rotazione:** tempo medio per raggiungere il settore da trasferire (tempo per 1/2 rotazione). Misurato in rpm (rounds per minute = giri al **minuto**).
- **Tempo di trasferimento:** tempo per trasferire l'informazione.
- A questi tempi va aggiunto il tempo per le operazioni del **controller**.



Hard disk magnetico - prestazioni

- Tempo medio di seek: da 8 a 20 ms (può diminuire di più del 75% se si usano delle ottimizzazioni).
- Tempo medio di rotazione: da 2.8 ms a 5.6 ms.
- Tempo medio di trasferimento: 2/15 MB per secondo e oltre con cache.
- Tempo di controllo (utilizzato dalla logica del controller).

Qual è il tempo di lettura/scrittura di un settore di 512Byte in un disco che ha velocità di rotazione di 7,200 rpm, tempo medio di seek di 12 ms, velocità di trasferimento di 10 Mbyte/s e tempo di controllo di 20 μ s?

Tempo totale: $12\text{ms} + (1/120) / 2 * 1000 \text{ms} + 0,5\text{kbyte} / 10\text{Mbyte/s} + 2\text{ms} = 12 + 4,2 + 0,05 + 0,02 = 16,27\text{ms}$

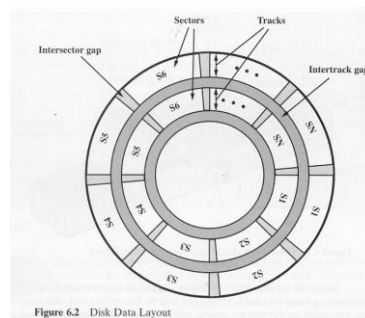
Per un tempo di seek medio pari al 25% del tempo nominale ($t_{\text{seek}} = 3\text{ms}$)

Tempo totale = $3\text{ms} + 4,2\text{ms} + 0,05 \text{ms} + 0,02 \text{ms} = 7,27\text{ms}$



Hard disk - struttura

- Ogni faccia è divisa in circonferenze concentriche chiamate **tracce** (10,000-50,000).
 - Ogni traccia è suddivisa in **settori** (64 - 500).
 - I settori sono suddivisi da **gap**
 - Il settore è la più piccola unità che può essere letta/scritta da/su disco (tipicamente blocco da 512byte, ma c'è la spinta a portarli a 4,096byte).



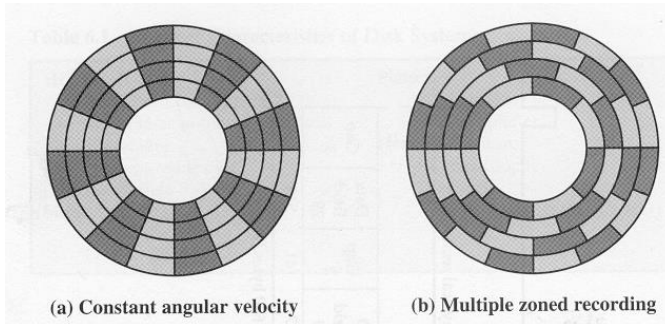


Memorizzazione dati su disco

La velocità di rotazione è costante.
Per ogni settore, il numero di bit per traccia è uguale.
Le tracce interne determinano la densità di bit.

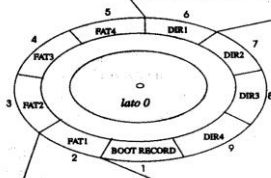
Solitamente, ogni traccia di un disco contiene la stessa quantità di bit \Rightarrow le tracce più esterne memorizzano informazione con densità minore.

Per aumentare l'impaccamento dell'informazione si utilizza la tecnologia del **multiple zone recording**.



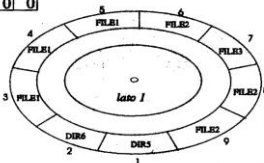
File name size date time FAT Attribute

FILE1	1340	04-11-94	13:15	3	A...
FILE2	1500	07-01-94	10:07	6	A...
FILE3	412	09-23-94	11:55	7	A...



Hard-disk: organizzazione delle informazioni

1	2	3	4	5	6	7	8	9
X	X	X	4	5	EOF	8	EOF	9
EOF	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0



La traccia 0 contiene il contenuto del disco.



Hard Disk - Seagate Ceetah 18XL



Dimensioni: 101.6 x 146.1 x 25.4mm.
Capacità: 18.2 Gbyte.

Forma: low-profile. Peso: 0.68kg.
Default Buffer (cache) Size 4,096 Kbytes Spindle Speed 10,000 RPM

Number of Discs (physical): 3
Total Cylinders: 14,384

Number of Heads (physical): 6
Bytes Per Sector: 512

Internal Transfer Rate (min-max): 284 Mbits/sec - 424 Mbits/sec

Formatted Int Transfer Rate (min-max) 26.6 MBytes/sec - 40.5 MBytes/sec
External (I/O) Transfer Rate (max): 200 MBytes/sec

Avg Formatted Transfer Rate: 35.5 MBytes/sec

Average Seek Time, Read-Write: 5.2-6 millisecc typical
Track-to-Track Seek, Read-Write: 0.6-0.8 millisecc typical
Average Latency: 2.99 msec

(→ milioni di cicli di clock)
(→ milioni di cicli di clock)

Typical Current (12VDC +/- 5%): 0.5 amps
Typical Current (5VDC +/- 5%): 0.8 amps
Idle Power (typ): 9.5 watts



Other disks



Characteristics	Seagate ST33000655SS	Seagate ST31000340NS	Seagate ST973451SS	Seagate ST9160821AS
Disk diameter (inches)	3.50	3.50	2.50	2.50
Formatted data capacity (GB)	147	1000	73	160
Number of disk surfaces (heads)	2	4	2	2
Rotation speed (RPM)	15,000	7200	15,000	5400
Internal disk cache size (MB)	16	32	16	8
External interface, bandwidth (MB/sec)	SAS, 375	SATA, 375	SAS, 375	SATA, 150
Sustained transfer rate (MB/sec)	73-125	105	79-112	44
Minimum seek (read/write) (ms)	0.2/0.4	0.8/1.0	0.2/0.4	1.5/2.0
Average seek (read/write) (ms)	3.5/4.0	8.5/9.5	2.9/3.3	12.5/13.0
Mean time to failure (MTTF) (hours)	1,400,000 @ 25°C	1,200,000 @ 25°C	1,600,000 @ 25°C	—
Annual failure rate (AFR) (percent)	0.62%	0.73%	0.55%	—
Contact start-stop cycles	—	50,000	—	>600,000
Warranty (years)	5	5	5	5
Nonrecoverable read errors per bits read	<1 sector per 10 ¹⁶	<1 sector per 10 ¹⁵	<1 sector per 10 ¹⁶	<1 sector per 10 ¹⁴
Temperature, shock (operating)	5°-55°C, 60 G	5°-55°C, 63 G	5°-55°C, 60 G	0°-60°C, 350 G
Size: dimensions (in.), weight (pounds)	1.0" x 4.0" x 5.8", 1.5 lbs	1.0" x 4.0" x 5.8", 1.4 lbs	0.6" x 2.8" x 3.9", 0.5 lbs	0.4" x 2.8" x 3.9", 0.2 lbs
Power: operating/idle/standby (watts)	15/11/—	11/8/1	8/5.8/—	1.9/0.6/0.2
GB/cu. in., GB/watt	6 GB/cu.in., 10 GB/W	43 GB/cu.in., 91 GB/W	11 GB/cu.in., 9 GB/W	37 GB/cu.in., 84 GB/W
Price in 2008, \$/GB	~ \$250, ~ \$1.70/GB	~ \$275, ~ \$0.30/GB	~ \$350, ~ \$5.00/GB	~ \$100, ~ \$0.60/GB



Errori nella memoria



Mean Time To Failure (MTTF): guasto medio a partire dall'accensione.

Per un disco arriva a 1.000.000 di ore = 114 anni.

Annual Failure Rate (AFR): percentuale di dischi che si possono guastare in un anno -> probabilità che un disco si guasti in 1 anno.

Se abbiamo un MTF = 1.000.000 => $(24 \times 365) / 1.000.000 = 0.876\%$

ore / failure

ore / anno

Failure / anno

Un grosso centro di calcolo può contenere 50.000 server, ciascuno contiene facilmente 2 dischi per un totale di 100.000 dischi.

Un AFR di 0,876% corrisponde per 100.000 dischi a una media di 876 guasti per anno, cioè più di 2 guasti al giorno!

Da qui la ragione di proteggere i dati!



Impatto dei guasti



Interruzione del servizio (impatto sul cliente)

Mean Time To Repair (MTTR) – Tempo medio per la riparazione.

Mean Time Between Failures (MTBF) – MTTF + MTTR – Metrica più utilizzata.

Availability – Disponibilità della memoria

MTTF

MTTF + MTTR

Availability misurata in “nines of availability”:

- 1 nove: 90% => 36.5 giorni di guasto / anno
- 2 nove: 99% => 3.65 giorni di guasto / anno
- 3 nove: 99,9% => 526 minuti di guasto / anno
- 4 nove: 99,99% => 52.6 minuti di guasto / anno**
- 5 nove: 99,999% => 5.26 minuti di guasto / anno**

100% Stand-by spears per avere MTTR = 0



Impatto dei guasti

Ridondanza
Gerarchia
Parallelismo
Speculazione

Interruzione del servizio (impatto sul cliente)

Mean Time To Repair (MTTR) – Tempo medio per la riparazione.

Mean Time Between Failures (MTBF) – $MTBF = MTTF + MTTR$ – Metrica più utilizzata.

Availability – Disponibilità della memoria

Availability misurata in “nines of availability”:

1 nove: 90% => 36.5 giorni di guasto / anno

2 nove: 99% => 3.65 giorni di guasto / anno

3 nove: 99,9% => 526 minuti di guasto / anno

4 nove: 99,99% => 52.6 minuti di guasto / anno

5 nove: 99,999% => 5.26 minuti di guasto / anno

MTTF

MTTF + MTTR

← **Good Internet Services**

Evitare i guasti: costruire i dispositivi per evitare che si guastino.

Tollerare i guasti: Stand-by spares → $MTTR = 0 \Rightarrow$ Availability del 100%

Predire i guasti: sostituzione prima del guasto (speculazione)



RAID

RAID è un acronimo che sta per Redundant Array of Independent Disks (originariamente, 1988, stava per Redundant Array of Inexpensive Disks).

Dischi grandi sono costituiti da insiemi di dischi piccolo (parallelismo nell'accesso, adatti ai multi-core) => aumenta la possibilità di guasti => RAID.

Ha queste caratteristiche:

- 1) RAID è un insieme, array, di dischi fisici visto dal sistema operativo come un drive logico singolo.
- 2) I dati vengono distribuiti attraverso i dispositivi fisici dell'array di dischi.
- 3) La capacità ridondante dei dischi viene utilizzata per memorizzare l'informazione di parità, che garantisce di potere recuperare i dati in casi di guasto (i guasti risultano più frequenti per la maggiore complessità dell'HW).



I RAID



Insieme di piccoli dischi letti in parallelo (cf. chip in parallelo di una MM).

RAID 0. Striping.

- Tecnica che distribuisce i dati sui diversi piccolo dischi in parallelo.
- I dati vengono letti come micro-blocchi unici, caricando dai diversi dischi.
- Non c'è ridondanza.
- Adatti all'elaborazione video, non adatti ai data-base, ecc....

RAID 1. Raddoppia il numero di dischi. Mirroring.

- Equivalente al codice di parità per ogni disco.
- Soluzione molto costosa in numero di dischi.
- Adopted by EMC, HP(Tandem), IBM

RAID 3. Raggruppa un certo numero di dischi a cui assegnare un disco di parità. Protection group.

- Equivalente al codice di parità (e.g. Hamming) per un insieme di parole di memoria.
- Compromesso tra numero di dischi (lunghezza della parola di memoria) e affidabilità.
- Occorre leggere più dischi per analizzare la parità (interleaving dei bit di parità e della parola di memoria).
- Popolare per grossi insiemi di dati: multi-media e calcolo scientifico.



I RAID



Stand-by spares.

- Cosa succede se si verifica un guasto fisico in una cella di memoria? Il blocco viene eliminato.
- Cosa succede se si verifica un guasto fisico a un disco? Il disco viene eliminato => Dischi aggiuntivi di riserva, sostituiscono il disco guasto e consentono il ricambio della parte con più calma

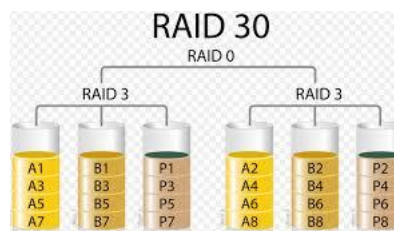
Evoluzione verso un controllo gerarchico.

- Accessi a dati ampi -> più dischi e controllo
- Accesso a dati stretti -> pochi dischi

RAID 10 (1+0). Striped mirrored.

RAID 01 (0+1). Mirrored striped.

RAID 30 (3+0). Striped on protection groups.

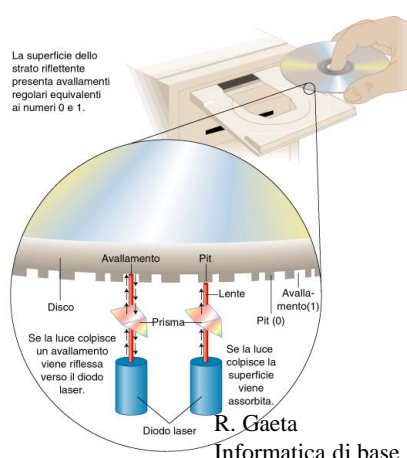




CD-ROM / DVD



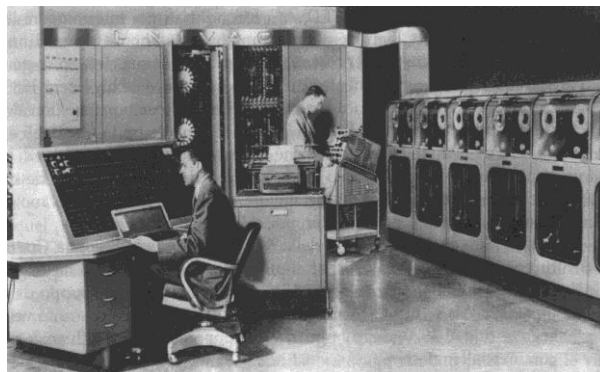
- I CD-ROM sono basati sulla tecnologia laser per la memorizzazione delle informazioni. Vennero lanciati sul mercato nel 1982 da Philips e Sony per la registrazione di suoni.
- Memorizzano l'informazione codificata mediante fori sulla superficie del disco.
- Un raggio laser colpisce la superficie del disco e viene o riflesso o produce scattering (assorbimento).
- Su CD-ROM è possibile immagazzinare informazione con una densità maggiore rispetto ai dischi magnetici.
- Un CD-ROM può memorizzare più di 650 MB di dati.
- Un DVD (Digital Video Disk) arriva a memorizzare 15.90 Gbyte (DVD-18). Esistono diversi dialetti e diversi formati HW: DVD-R, DVD+R.
- Un disco Blu-Ray (laser blu) arriva a 200 Gbyte ed è il dispositivo oggi più utilizzato.
- La scrittura è più problematica: ripristino della superficie mediante micro-fusion.



R. Gaeta
Informatica di base



Nastri



UNIVAC – I prima Main Memory



Nastri oggi



<https://www.ibm.com/it-infrastructure>



Data cartridge	Native data capacity
Ultrium 8 and WORM	12 TB (30 TB at a 2.5:1 compression ratio)
Ultrium 7 M8 Format (for LTO8 drives only)	9 TB (22.5 TB at a 2.5:1 compression ratio)
Ultrium 7 and WORM	6 TB (15 TB at a 2.5:1 compression ratio)
Ultrium 6 and WORM	2.5 TB (6.250 TB at a 2.5:1 compression ratio)
Ultrium 5 and WORM	1.50 TB (3 TB at a 2:1 compression ratio)
Ultrium 4 and WORM	800 GB (1.6 TB at a 2:1 compression ratio)
Ultrium 3 and WORM	400 GB (800 GB at a 2:1 compression ratio)
Ultrium 2	200 GB (400 GB at a 2:1 compression ratio)
Ultrium 1	100 GB (200 GB at a 2:1 compression ratio)

IBM Corporation, Hewlett-Packard (HP), and Quantum dal 1997

Tape System Library Manager -> up to 3,000 Petabyte of data.

Testine multiple – parità.



Specifiche tecniche



HP StoreEver LTO-6 Ultrium 6250 Tape Drive

Feature	Description
Native	2.5TB
Compressed (assumes 2:1 data compression for LTO-4 and 5; 2.5:1 compression for LTO-6)	6.25TB
Sustained transfer rate, native	160 MB/s
Sustained transfer rate, compressed (assumes 2:1 data compression for LTO-4 and 5; 2.5:1 compression for LTO-6)	400 MB/s
Burst transfer rate	600 MB/s with 6 Gb/s SAS
Data rate matching range	54 - 160 MB/s (native)
Data access time (from BOT)	50 s typical for LTO 6 media
Rewind time from EOT	98 s (2.5TB tape)
Rewind tape speed	9.0 m/s
Average load time	24 s
Average unload time	19 s

Comparable to SSD



Caratteristiche dei nastri e dei DVD



Stesse proprietà dei CD-ROM su quantità di dati molto maggiori.

- Memoria permanente
- Memoria removibile (to be stored in a safe place)
- Scalabile (è sufficiente aggiungere nastri, non è necessario aggiungere unità)
- Portatile
- Veloci (700 Mbit / s – senza seek time)
- Affidabili (codici di parità,)
- Basso impatto (basso consumo di corrente).



Caratteristiche della memoria



Posizione della memoria:

- Processore
- Interna (cache)
- Esterna (cache + Memoria Principale)
- Disco

Metodo di accesso:

- Sequenziale (e.g. Nastri).
- Diretto (posizionamento + attesa, e.g. Dischi).
- Random Access (circuito di lettura / scrittura HW, tempo indipendente dalla posizione e dalla storia, e.g. Cache e Memoria principale).
- Associativa (Random Access, il contenuto viene recuperato a partire da un sottoinsieme incompleto dello stesso).

Caratteristiche fisiche:

- Nelle memorie volatili (E.g. Cache), l'informazione sparisce quando si toglie l'alimentatore (memorie a semiconduttore).
- Nelle memorie non-volatili, l'informazione è duratura (un esempio di memoria volatile è la memoria magnetica dei dischi e dei nastri). Esistono memorie a semiconduttore non-volatili (ROM).



Sommario



Gestione della memoria

I codici di errore

Gli altri dispositivi di memoria