La tecnologia delle memorie

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione
borghese@dsi.unimi.it

Università degli Studi di Milano

Sommario

Gestione delle memorie cache.

SRAM.

DRAM.

Correzione degli errori.

Tassonomia del funzionamento

HIT Successo nel tentativo di accesso ad un dato: è presente al livello superiore della gerarchia.

MISS Fallimento del tentativo di accesso al livello superiore della gerarchia => il dato o l'indirizzo devono essere cercati al livello inferiore.

HIT_RATE Percentuale dei tentativi di accesso ai livelli superiori della gerarchia che hanno avuto successo.

HIT_RATE = Numero_successi / Numero_accessi_memoria

MISS_RATE Percentuale dei tentativi di accesso ai livelli superiori della gerarchia che sono falliti

MISS_RATE = Numero_fall. / Numero_accessi_memoria

HIT_RATE + MISS_RATE = 1

Criteri di progettazione

Cache primaria: massimizzo Hit rate.

Cache secondaria: minimizzo Miss penalty.

Come funziona la scrittura?

Write-through. Scrittura in cache e contemporaneamente in RAM.

Write_buffer per liberare la CPU (DEC 3100)

Sincronizzazione tra contenuto della Memoria Principale (che può essere letto anche da I/O e da altri processori) e Cache.

Svantaggio: traffico intenso sul bus per trasferimenti di dati in memoria.

Write-back. Scrittura ritardata. Scrivo quando devo scaricare il blocco di cache.

Utilizzo un bit di flag: UPDATE, che viene settato quando altero il contenuto del blocco.

Vantaggiosa con cache n-associative.

Alla Memoria Principale trasferisco il blocco.

Cache coherence

Mantenimento dell'informazione di cache coerente tra varie cache (sistemi multi-processori).

Bus watching with write through.
Il controller della cache monitora il bus indirizzi + segnale di controllo write della memoria.
Invalida il contenuto di un blocco se il suo corrispondente in memoria viene scritto.
Quando funziona? Quando tutti i dispositivi utilizzano un meccanismo write-through.

Hardware trasparency.
Circuito addizionale attivato ad ogni scrittura della Memoria Principale.
Copia la parola aggiornata in tutte le cache che contengono quella parola.

Noncachable memory.
Viene definita un'area di memoria condivisa, che non deve passare per la cache.

Gestione dei fallimenti di una cache

Hit – è quello che vorremmo ottenere, il funzionamento della CPU non viene alterato.

Miss – in lettura devo aspettare che il dato sia pronto in cache -> stallo.

Passi da eseguire in caso di Miss:
1) Ricaricare l'indirizzo dell'istruzione (PC-> PC-4)
2) Leggere il blocco di memoria dalla memoria principale.
3) Trasferire il blocco in cache, aggiornare i campi validita' e tag.
4) Riavviare la fase di fetch dell'istruzione.

NB Il programma non può continuare!!

In scrittura?

Tempi di trasferimento

Tempi di accesso:
1 ciclo di clock per inviare l'indirizzo.
15 cicli di clock per ciascuna attivazione della Memoria (lettura di parola).
1 ciclo di clock per trasferire una parola al livello superiore (cache).

Blocco di cache di 4 parole, blocco di memoria RAM di 1 parola

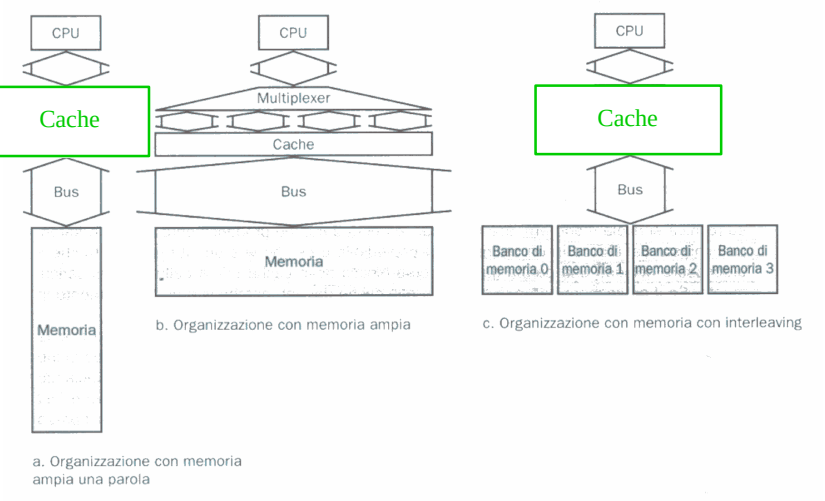
⇓

Miss_Penalty = 1 + 15 * 4 *(parole)* + 1 * 4 *(parole)* = *65 cicli_clock*

#byte / ciclo_clock = 4(*parole*) * 4(*byte/parola*) / 65(*cicli_clock*) ≅ 0,25(*byte / ciclo_clock*)

Obbiettivi:
•Diminuire la penalità di fallimento (miss_penalty).
•Diminuire il tasso di fallimento (miss_rate).

Riduzione del miss penalty



Interleaving (interallacciamento). Banchi che possono essere letti in parallelo.

Valutazione della riduzione della "miss_penalty"

Architettura standard: penalità di miss è di 65 cicli_clock.

Maggiore ampiezza della memoria:

- Organizzazione della Memoria Principale per blocchi.
- Bus più ampio (bus dati largo un blocco, 4 parole).
- Per blocchi di Memoria di 4 parole, blocchi di cache di 4 parole:

  Miss_penalty = 1 + 15 * 1 + 1 * 1 = 17 cicli_clock.

  #byte / ciclo_clock = 4(*parole*) * 4(*byte/parola*) / 16(*cicli_clock*) = 0,94(*byte / ciclo_clock*)

Interleaving:

- Organizzazione della Memoria Principale per banchi con accesso indipendente alla memoria (interleaving).
- Bus standard (trasferimento di 1 parola alla volta).
- Per blocchi di Memoria di 1 parola, blocchi di cache di 4 parole:

  Miss_penalty = 1 + 15 * 1 + 1*4 = 20 cicli_clock.

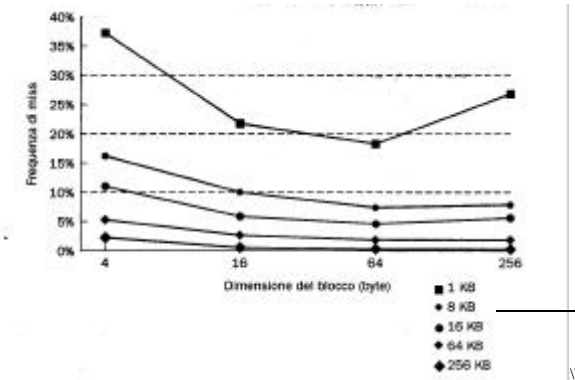
  #byte / ciclo_clock = 4(*parole*) * 4(*byte/parola*) / 20(*cicli_clock*) = 0,80(*byte / ciclo_clock*)

Riduzione del Miss rate

La parola di cache (blocco) è un multiplo della parola della macchina.

Vantaggi: per la località spaziale, diminuisco la frequenza di miss.
*Svantaggi:*per le dimensioni del blocco rispoetto alla dimensione totale della cache aumenta la penalità di miss: competizione per le poche linee di cache.
La località diminuisce all'aumentare della dimensione della linea.

La lunghezza della linea di cache dipende dalla parola del processore. Oggi si va verso 64-128byte.



Cache gerarchiche

Cache primaria nel processore.
Cache secondaria con un bus dedicato per il trasferimento al processore.

Problemi: complessità nel circuito che deve assicurare la cache coherence.

Split-cache: Cache dati e cache istruzioni.

Vantaggi. Possibilità di analizzare le istruzioni in coda (contenute nella cache istruzioni) mentre si eseguono altre istruzioni (che lavorano su dati contenuti nella cache dati), senza dovere competere per l'accesso alla cache. Efficiente per le architetture superscalari.

Svantaggi. Minore hit rate, perchè non si sfrutta al meglio la memoria cache. Si potrebbe riempire un'unica cache maggiormente con dati od istruzioni a seconda del frammento di codice correntemente in esecuzione.

Le dimensioni di alcune cache

Processor	Type	Year of Introduction	L1 cache[a]	L2 cache	L3 cache
IBM 360/85	Mainframe	1968	16 to32 KB	—	—
PDP-11/70	Minicomputer	1975	1KB	—	—
VAX 11/780	Minicompterer	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]	PC/server	2001	16 KB/16 KB	96 KB	4 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	—

Principi sulla Memoria Virtuale

Due motivazioni:

•Estensione della memoria fisica. Maggiore quantità di memoria.

•Gestione del multi-tasking. Negli anni ’90 *overlay* definito nel linker, ora trasparente tramite il gestore della memoria virtuale.

Ogni programma ha il suo spazio di indirizzamento.
Mappatura della spazio di indirizzamento nella memoria fisica
(*memory mapping* tramite la *page table*).

Memoria virtuale (estensione su disco) è concettualmente analoga alla cache.

Blocco di memoria → Pagina.
Miss →Page Fault.

Sommario

Gestione delle memorie cache.

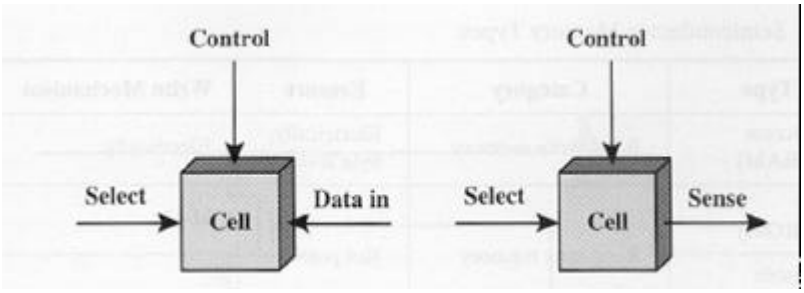
SRAM.

DRAM.

Correzione degli errori.

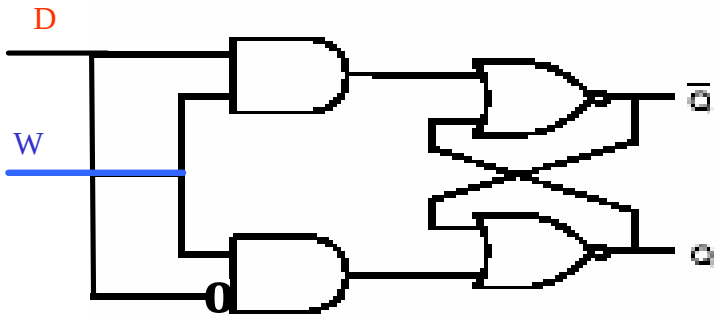
Cella di memoria

La memoria è suddivisa in celle, ciascuna delle quali assume un valore binario stabile.
Si può scrivere il valore 0/1 in una cella.
Si può leggere il valore di ciascuna cella.

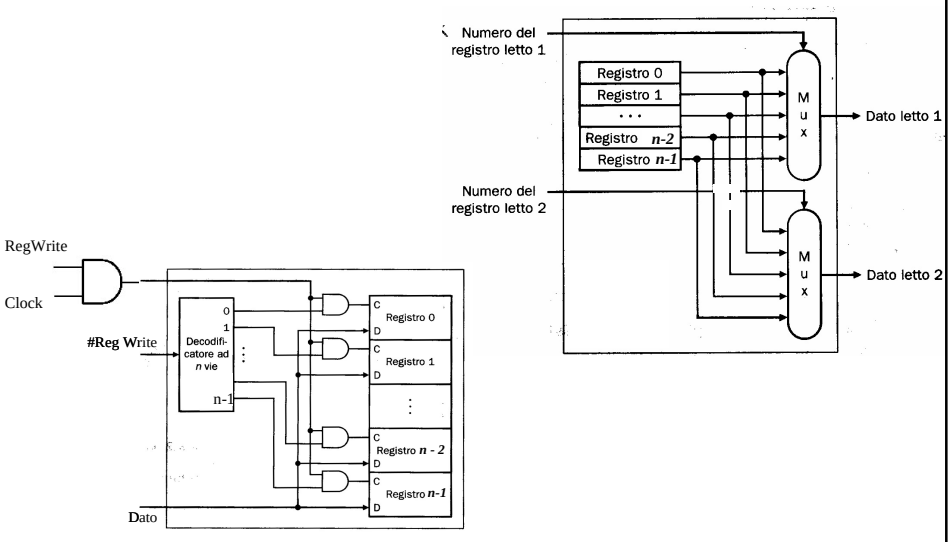


Quale struttura di memoria abbiamo già incontrato?

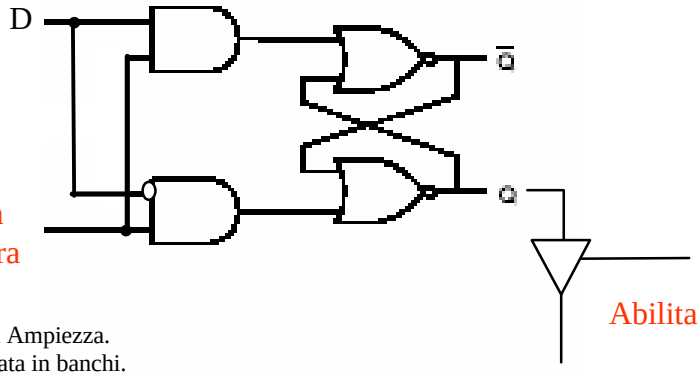
Cella SRAM



Register file

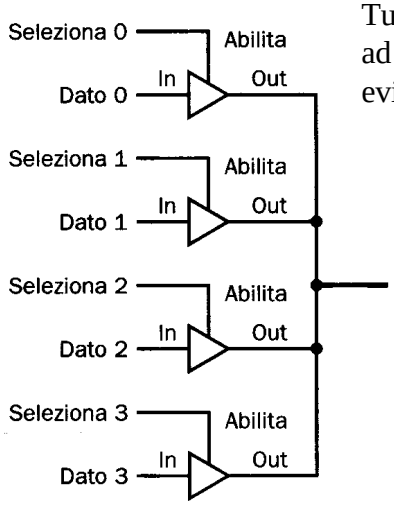


Memorie SRAM



- Altezza x Ampiezza.
- Organizzata in banchi.
- Tempo di lettura è il tempo di abilitazione del buffer di uscita.
- Seleziona il chip, seleziona la cella di memoria: attivo per operazioni di lettura e scrittura.
- La temporizzazione della scrittura deve rispettare: tempo di set-up + tempo di hold.
- Non si può utilizzare la tecnica del register file. Memorie di 64K x richiederebbero un MUX a 64K vie! Si utilizza invece la tecnologia three-state, aggiungendo un buffer three-state.

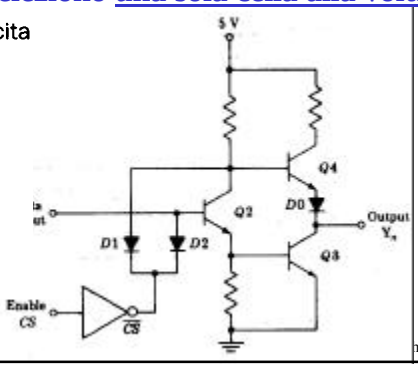
Memoria three-state



Tutte le uscite delle celle sono collegate ad un'uscita comune => E' necesario evitare conflitti fra le uscite.

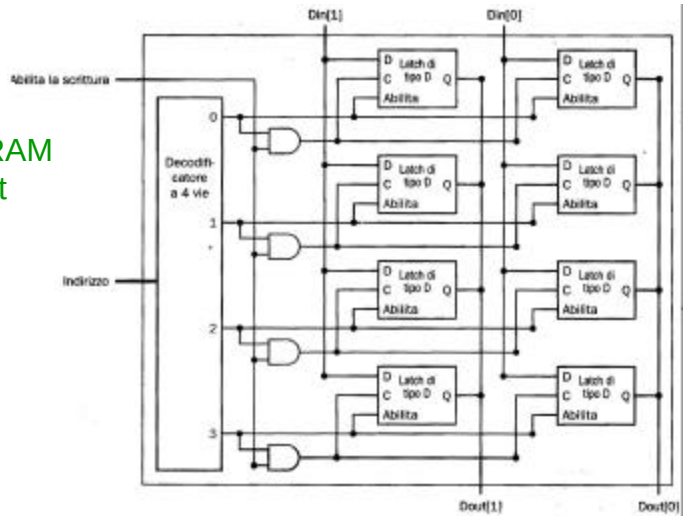
Uscite "isolate" con porte *three-state*

Seleziono una sola cella alla volta



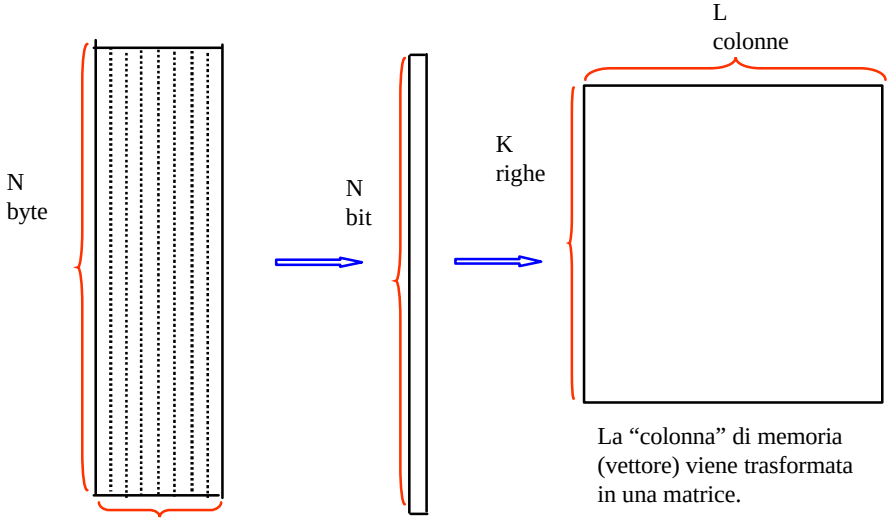
Esempio di SRAM

Esempio: SRAM
4 celle x 2 bit



Problemi con il crescere del numero di linee. Esempio: SRAM 16K x 8.
Decodificatore a 14 ($\log_2$16K) bit e 16K uscite per 16K linee di abilitazione e di selezione (ingresso C).

Struttura di una SRAM



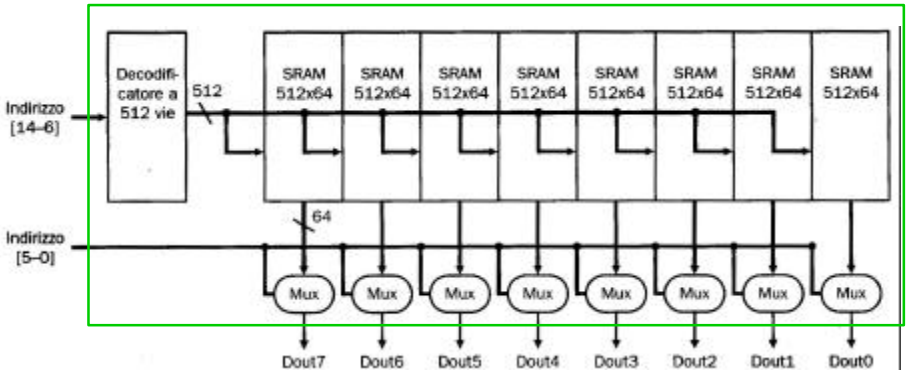
La "colonna" di memoria (vettore) viene trasformata in una matrice.

Ampiezza: 8 bit = 8 banchi con ampiezza 1 bit.

$N = K * L$

Indirizzamento SRAM a BANCHI

Esempio: SRAM 32K x 8. Trasformo 32K linee in una matrice: 512 linee x 64 colonne (bit)



Il decodificatore sarà a 9 bit ($\log_2 512$) per selezionare una delle 512 linee. Ciascuna linea fornisce 64bit. Ne seleziono uno con il Mux.

Nell'approccio non a banchi avrei avuto bisogno di un decodificatore a 15 bit ($\log_2 32K$).

Sommario

Gestione delle memorie cache.

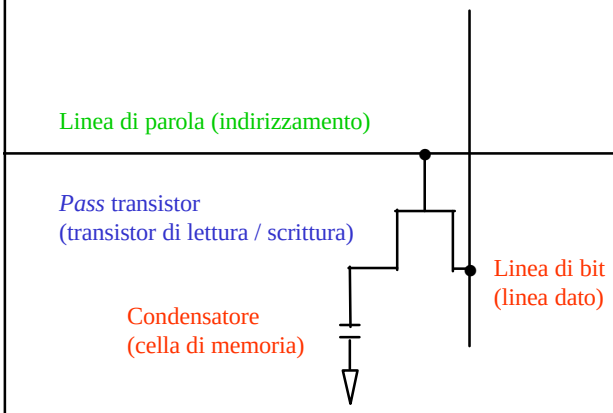
SRAM.

DRAM.

Correzione degli errori.

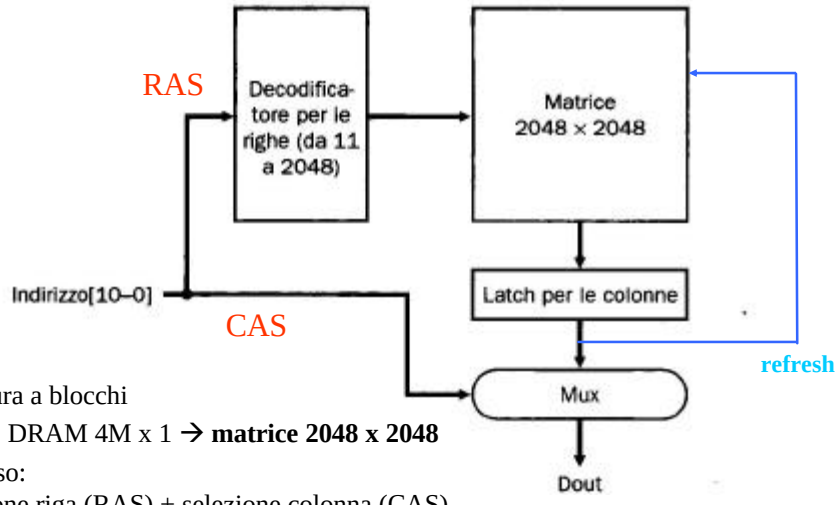
Memorie DRAM

Dynamic RAM. Condensatore che viene caricato.



1 *Pass* transistor + 1 condensatore.
La lettura scarica la memoria che deve essere ricaricata: refresh gestito autonomamente dal controllore della memoria.

Struttura a matrice di una DRAM



- Struttura a blocchi
  - es. DRAM 4M x 1 → **matrice 2048 x 2048**
- Accesso:
  selezione riga (RAS) + selezione colonna (CAS)

Efficiente per il refresh (refresh di riga) – (35-70ms, tempo di carica dei condensatori).
Utilizzo per la Memoria Principale.

Evoluzione delle SRAM e DRAM

Synchronous version.

Trasferimento ***a burst*** o a pagina: trasferimento consecutivo di parole ad indirizzi consecutivi (e.g. RAM EDO).

SDRAM

La fase di indirizzamento e di recupero dei dati vengono separate in modo da ridurre al minimo l'impatto della latenza.

Tra l'indirizzamento ed il recupero dei dati, il processore puo' eseguire altri compiti (NB il processore puo' essere la CPU o il controllore della memoria, o altro: il dispositivo che controlla la memoria).

DDR-SDRAM. Riescono a trasferire 2 bit per ciclo di clock. Frequenza doppia rispetto alla frequenza del clock del bus.

Alcuni dati: aprile 2004

http://www.samsung.com/Products/Semiconductor/

SRAM

Low power, 8M x 16, tempi di accesso: 70ns. Disponibilita' Page e Burst.
Sincrone, 1M x 36, 2M x 18, tempi di accesso: 2.6ns. Controllo di parità.
Sincrone, high speed, 1M x 18 o 512K x 36, tempi di accesso: 1.6ns.
Asincrone: 8M x 16, tempi di accesso: 10ns.

DRAM (DDR)

256M x 4, rate 266Mb/s (133Mhz). Tempo di refresh 30-40ms. (DDR).

SDRAM (DDR)

128M x 8, rate 266Mb/s (133Mhz).
16M x 16, rate 400Mb/s (200Mhz). 3 clock di latenza, 2-4-8 larghezza del burst.

Sommario

Gestione delle memorie cache.

SRAM.

DRAM.

Correzione degli errori.

ECC

- Errori dovuti a malfunzionamenti HW o SW.
  - Date le dimensioni delle memorie (**10^{10} 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.
  - Consente di individuare errori singoli in una parola.
  - Non consente di individuare su quale bit si e' 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 8bit /128 bit.

Codici rivelatori d'errore

- **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**
    - **0001 1010 1**
  - Un errore su uno dei bit porta ad un n. dispari di "1"

- Prestazioni del codice
  - mi accorgo dell'errore, ma non so dov'è
  - **rivelo ma non correggo errori singoli**
  - **COSTO: 1 bit aggiuntivo ogni 8 → 9/8 = +12,5%**

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 ...

  - Un errore su un bit di ciascuna terna può essere corretto:

    000 → 010 → 000

    111 → 110 → 111

- Prestazioni del codice
  - mi accorgo dell'errore, ma non so dov'è
  - **rivelo e correggo errori singoli**
  - **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 → d = 2

- **Distanza minima** di un codice, d_{MIN}
  - il valor minimo di **d** tra tutte le coppie di sequenze 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**: $d_{MIN} = 2 \rightarrow$ **t=1, r=0**
  - Codice a **ripetizione (3,1)**: $d_{MIN} = 3 \rightarrow$ **t=2, r=1**

Applicazioni nelle memorie

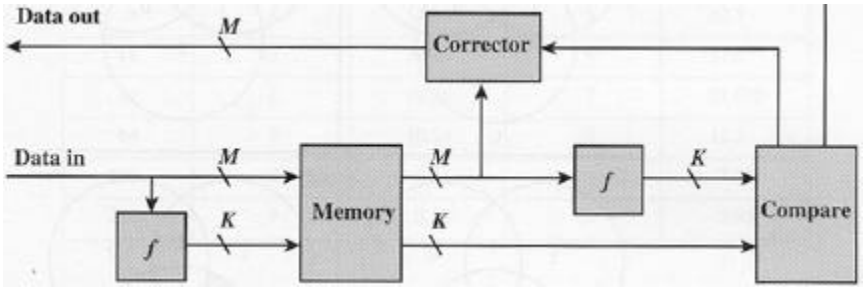
- RAM con controllo di parità
  - Aggiungo un bit di parità ad ogni byte
  - Es: RAM 1 M x 9 bit (8+1)

- RAM con codice correttore di errori (ECC)
  - si usa nelle memorie cache
  - codici ECC evoluti (alta efficienza)
    - Hamming, CCITT-32, Reed Solomon, …

Correzione degli Errori

- **OUT** possibili:
  - No errors detected
    - I dati letti possono essere inviati in uscita così come sono.
  - 1 errore è stato individuato e corretto
    - I bit del dato, più il codice associato vengono inviati al correttore, il quale provvede a correggere il dato.
  - 1 errore individuato, ma impossibile da correggere
    - segnala l'errore.

out



Dimensione di codici ECC

- Conviene applicare ECC a parole più lunghe possibile → aggiungo meno ridondanza → maggiore efficienza del codice
  - A costo di complessità maggiori di codifica/decodifica

	Single-Error Correction		Single-Error Correction/ Double-Error Detection	
Data Bits	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

Sommario

Gestione delle memorie cache.

SRAM.

DRAM.

Correzione degli errori.