


La tecnologia delle memorie

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

Università degli Studi di Milano

Riferimento Patterson: 2.11 - 5.2, 5.4, 5.8, C8, C9.


Sommario

SRAM.

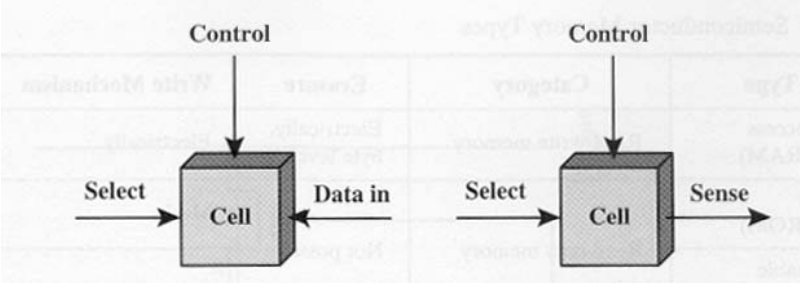
DRAM.

Gestione delle memorie cache.

Codici di errore

Cella di memoria

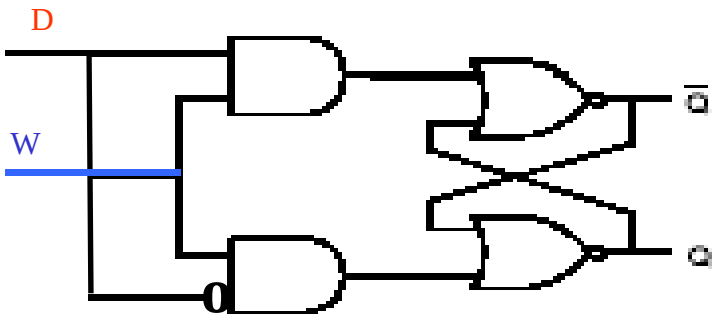
La memoria è suddivisa in celle, ciascuna delle quali assume un valore binario stabile.
Si può scrivere il valore 0/1 in una cella.
Si può leggere il valore di ciascuna cella.



Quale struttura di memoria abbiamo già incontrato?

Control (lettura – abilitazione; scrittura)
Select (cf. dataport)
Data in & Sense (Data in & Data out).

Cella SRAM



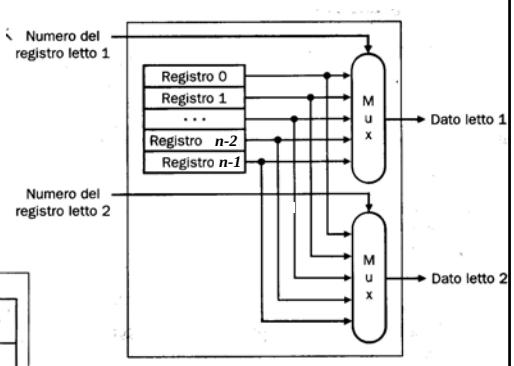
Selezione (porta di lettura e porta di scrittura)
Lettura - sempre disponibile in uscita
Scrittura – segnale esplicito (in AND con il clock in caso di cella sincrona).

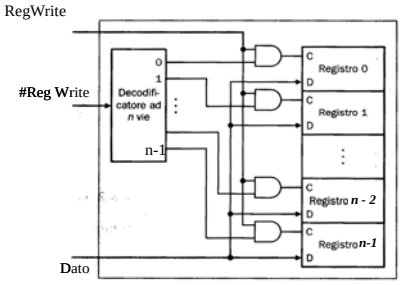
Register file

Il tempo di lettura dipende dal cammino critico dei Mux.

Il tempo di scrittura dipende dal cammino critico del Decoder.

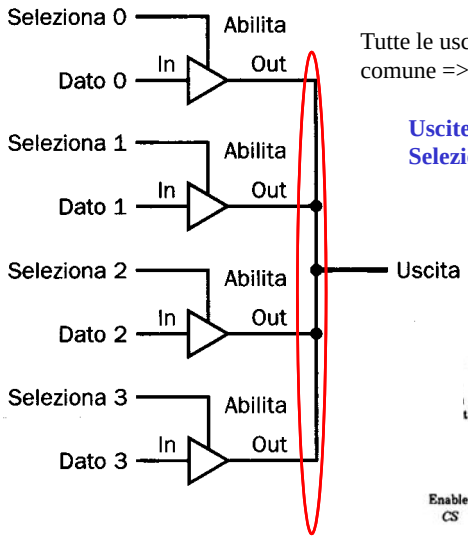
Numero_registro = selettore.





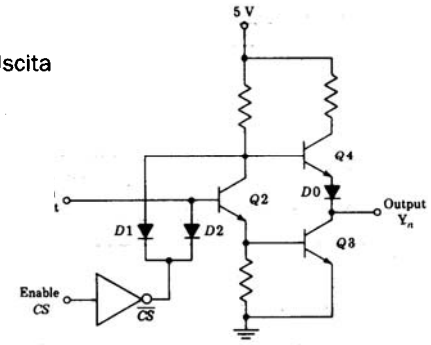
Selezione – #registro
Lettura .- sempre disponibile in uscita (dopo tempo di commutazione del MUX)
Scrittura – segnale esplicito (in AND con il clock in caso di cella sincrona).

Memoria three-state

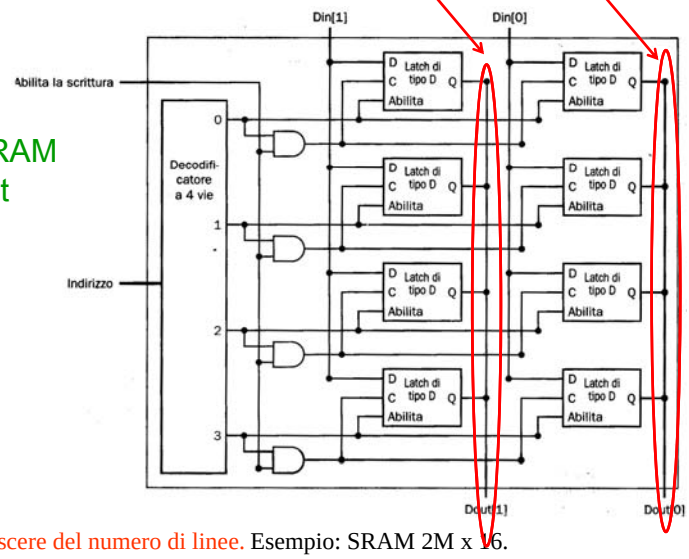


Tutte le uscite delle celle sono collegate ad un'uscita comune => E' necesario evitare conflitti fra le uscite.

Uscite "isolate" con porte *three-state*
Seleziono una sola cella alla volta



Esempio di SRAM

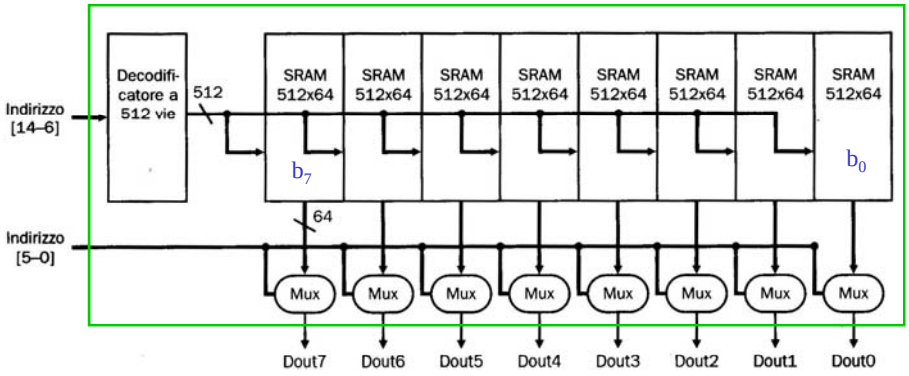


Esempio: SRAM
4 celle x 2 bit

Problemi con il crescere del numero di linee. Esempio: SRAM 2M x 16.
Decodificatore a 21 ($\log_2$2M) bit e 2M uscite per 2M linee di abilitazione e di selezione (ingresso C) dei bistabili. Utilizzo del decodificatore anche in lettura.

Indirizzamento SRAM a matrice

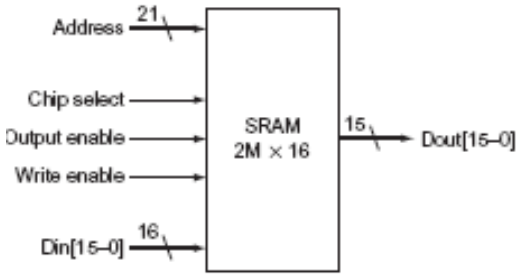
Esempio: SRAM 32K x 8 bit. Trasformo 32K linee in una matrice: 512 linee x 64 (bit)



Il decodificatore sarà a 9 bit ($\log_2$512) per selezionare una delle 512 linee (cf. cache). Ciascuna linea fornisce 64 bit. Ne seleziono uno con il Mux (controllato dai 6 bit meno significativi).

Nell'approccio non a matrice avrei avuto bisogno di un decodificatore a 15 bit ($\log_2$32K). Qual è il vantaggio?
Quale gruppo di 64 b_j leggo dalle SRAM?

Chip di SRAM



Tempo di accesso:
da Address a Dout.

Selezione – indirizzo + Chip select.
Scrittura – Write enable.
Lettura – Output enable.

E' una memoria veloce, ma costosa.

Sommario

SRAM.

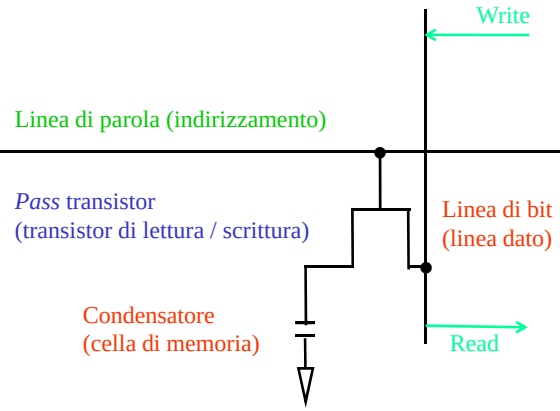
DRAM.

Gestione delle memorie cache.

Codici di errore

Memorie DRAM

Dynamic RAM. Condensatore che viene caricato. ∀bit (1 condensatore contro 4-6 SRAM).

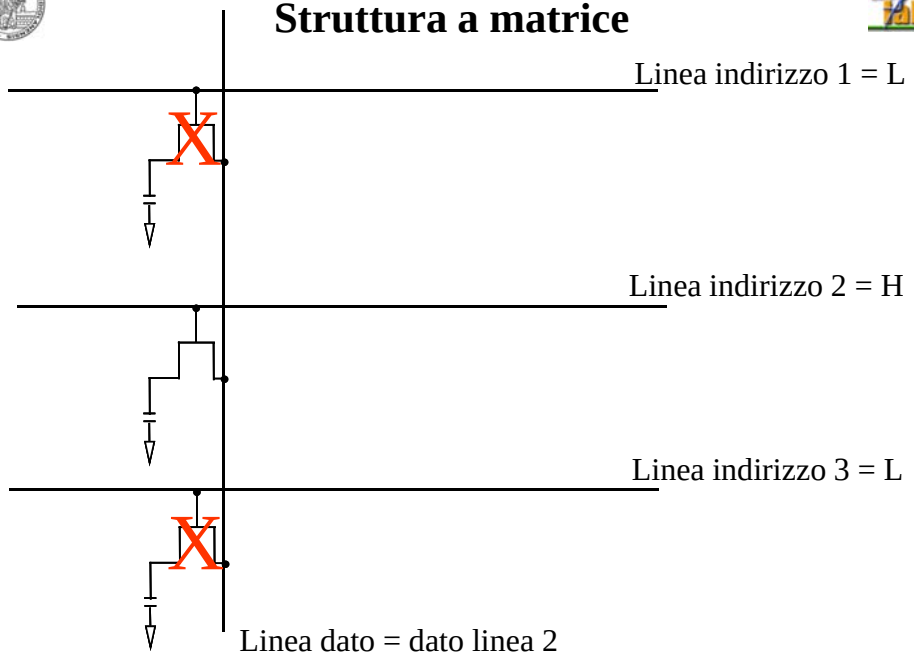


Scrittura: carica / scarica

Lettura: la linea di bit è portata ad una tensione intermedia, e poi tirata verso low o high a seconda che il condensatore sia carico / scarico: amplificazione della carica.

1 *Pass* transistor + 1 condensatore.
La lettura scarica la memoria che deve essere ricaricata.

Struttura a matrice



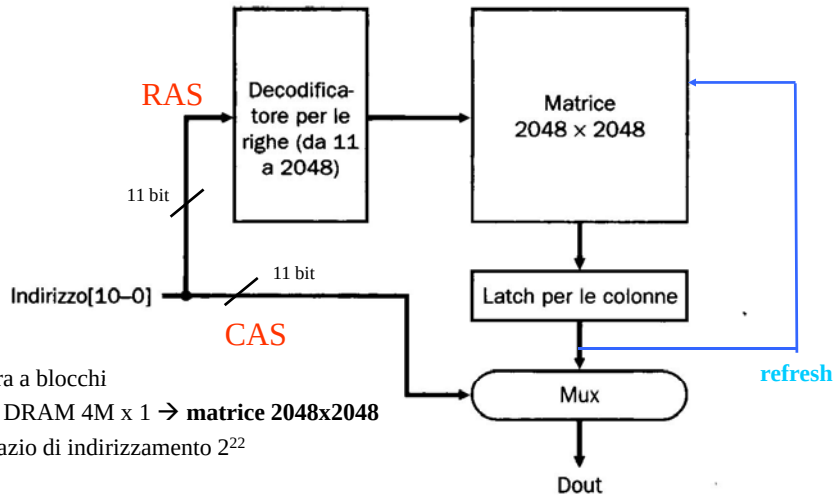
I problemi delle DRAM

I condensatori …. si scaricano (qualche millisecondo)

refresh gestito autonomamente dal controllore della memoria mediante ciclo lettura/scrittura

Cosa leggo/scrivo? Quale/i bit?

Struttura a 2 livelli (matrice) di una DRAM



- Struttura a blocchi
  - es. DRAM 4M x 1 → **matrice 2048x2048**
  - Spazio di indirizzamento 2^{22}

•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 (Synchronous DRAM)

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.

Specifiche delle DRAM (Patterson and Hennessy, 2008)

Year introduced	Chip size	$ per GB	Total access time to a new row/column	Column access time to existing row
1980	64 Kbit	$1,500,000	250 ns	150 ns
1983	256 Kbit	$500,000	185 ns	100 ns
1985	1 Mbit	$200,000	135 ns	40 ns
1989	4 Mbit	$50,000	110 ns	40 ns
1992	16 Mbit	$15,000	90 ns	30 ns
1996	64 Mbit	$10,000	60 ns	12 ns
1998	128 Mbit	$4,000	60 ns	10 ns
2000	256 Mbit	$1,000	55 ns	7 ns
2004	512 Mbit	$250	50 ns	5 ns
2007	1 Gbit	$50	40 ns	1.25 ns

Organizzazione della memoria

Organizzazione a matrice (cf. cache)

Organizzazione gerarchica.

Sommario

SRAM.

DRAM.

Gestione delle memorie cache.

Codici di errore

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 (fase Mem):

1) Ricaricare la pipeline (indirizzo dell'istruzione (PC-> PC-4), decodifica, esecuzione, memoria…)
2) Leggere il blocco di memoria dalla memoria principale.
3) Trasferire il blocco in cache, aggiornare i campi validita' e tag.
4) Avviare la fase di fetch, decodifica, esecuzione, memoria dell'istruzione con i dati corretti. .

NB Il programma non può continuare!!

In scrittura?

Come funziona la scrittura?

Quando c'è register spilling il dato viene scritto in cache e si verifica un **disallineamento** della memoria principale e della cache.

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 quando devo scrivere da CPU a cache (è equivalente al register spilling).

Contenuto della memoria principale e della cache può non essere allineato.


Coerenza

Coerenza: determina se il dato letto è congruente con quello contenuto nella memoria principale.
Consistenza: quando un dato può essere letto dopo una scrittura?

Esempio su una memoria Write-through

Time step	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				0
1	CPU A reads X	0		0
2	CPU B reads X	0	0	0
3	CPU A stores 1 into X	1	0	1

Cosa succederebbe se la cache fosse “Write-back”?

Need for a serialization of the writes (cf. Commit unit in the CPU)


Bus snooping

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 indirizzo in memoria viene scritto.

Quando funziona? Quando tutti i dispositivi utilizzano un meccanismo write-through.

Write invalidate protocol
Il controller della memoria monitora il bus indirizzi (**snooping**). Quando si verifica una write, cerca se il dato nell’indirizzo di scrittura è presente nelle altre cache.
Invalida il blocco nelle cache in cui il dato sra stato copiato.
Non c’è un’informazione di “blocco aggiornato” centralizzata, ma è distribuita sulle varie cache.

Cache coherence

Altri meccanismi:

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.

NB Blocchi di grandi dimensioni possono provocare il **false sharing**. I compilatori (ed i programmatori) sono incaricati di risolvere questo problema.

Protezione della memoria

Data race sulla memoria principale ➔ Lock di una cella di memoria e unlock.

L'accesso ad una cella di memoria viene riservato ad una certa procedura (e ad un certo processore)

```
try:    add $t0,$zero,$s4 ;copy exchange value
        ll $t1,0($s1) ;load linked
        sc $t0,0($s1) ;store conditional
        beq $t0,$zero,try ;branch store fails
        add $s4,$zero,$t1 ;put load value in $s4
```

Si controlla che la load abbia avuto successo controllando il contenuto di $t0 (sc). Se la procedura non è riuscita a leggere perchè c'era un blocco sulla cella di memoria, riprova.

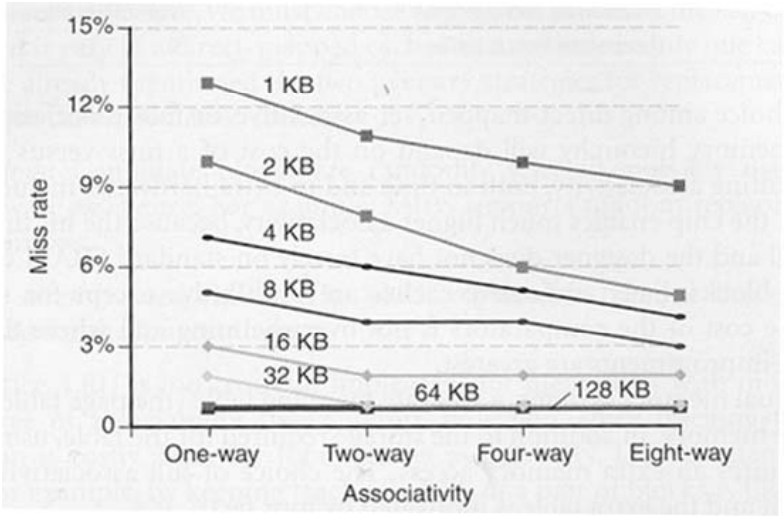
Criteri di progettazione

Cache primaria: massimizzo Hit rate.

Cache secondaria: minimizzo Miss penalty (massimizzo transfer rate).

Miss rate funzione del grado di associatività

Aumenta la associatività → Aumenta il tempo di trasferimento alla CPU.

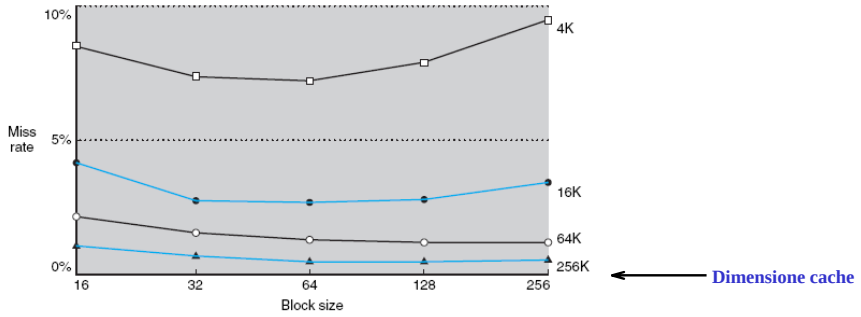


Miss rate in funzione della lunghezza del blocco

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 rispetto alla dimensione totale della cache aumenta la penalità di miss: competizione per le poche linee di cache. La penalità di miss è influenzata dalla lunghezza della linea 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.



Split cache

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.

Il register spilling e le miss sono inevitabili ➔ Come fare vedere al processore una memoria sufficientemente veloce?

Criteri di progettazione

Cache primaria: massimizzo Hit rate.

Cache secondaria: minimizzo Miss penalty (massimizzo transfer rate).

Miss penalty: esempio

Tempi di accesso:

1 ciclo di clock per inviare l'indirizzo.

15 cicli di clock per ciascuna attivazione della Memoria (lettura di parola, dettata dal cammino critico in lettura).

1 ciclo di clock per trasferire una parola al livello superiore (cache).

Blocco di cache di 4 parole, blocco di memoria (cache secondaria) di 1 parola

⇓

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

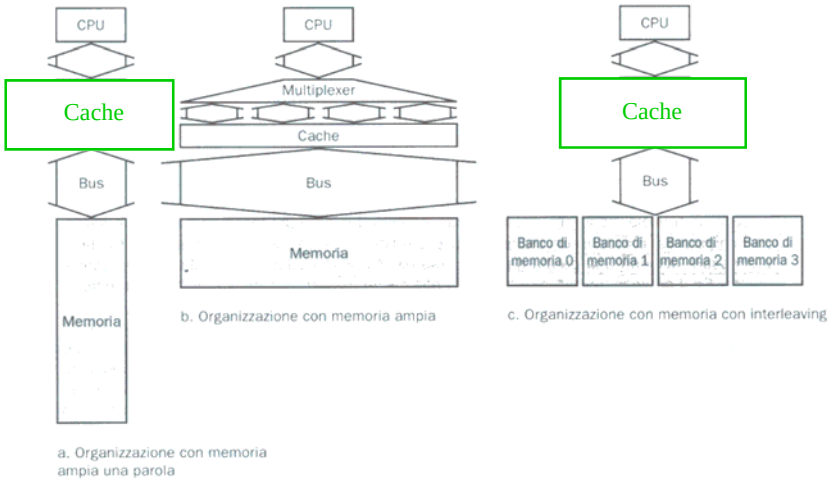
#byte / ciclo_clock = 4(*parole*) * 4(*byte/parola*) / 65(*cicli_clock*) ≅ 0,25(*byte / ciclo_clock*)

Obbiettivi:

•Diminuire la penalità di fallimento (miss_penalty).

•Diminuire il tasso di fallimento (miss_rate).

Riduzione del miss penalty



Interleaving (interallacciamento). Banchi che potrebbero essere trasferiti in parallelo alla cache.

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*)

Sommario

SRAM.

DRAM.

Gestione delle memorie cache.

Codici di errore

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
  - **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**: $d_{MIN} = 2$ → t=1, r=0
  - Codice a **ripetizione (3,1)**: $d_{MIN} = 3$ → t=2, r=1

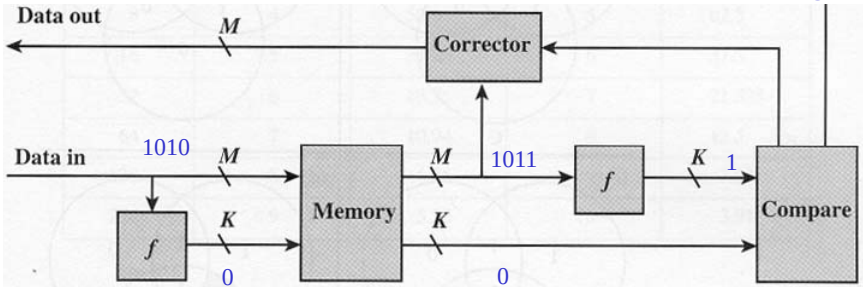

Applicazioni nelle memorie

- 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.



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

E' conveniente avere parole di memoria lunghe.

Sommario

Gestione delle memorie cache.

SRAM.

DRAM.

Correzione degli errori.