


La struttura delle memorie cache

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

Università degli Studi di Milano

Riferimento Patterson: 5.2, 5.4, B.9


Sommario

Ottimizzazioni software

SRAM

DRAM

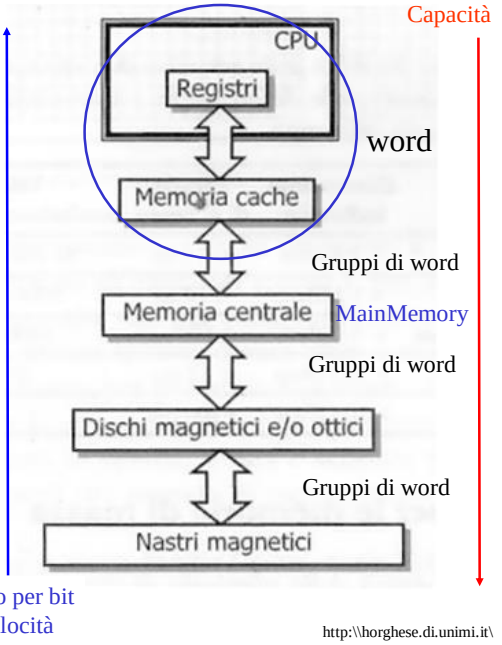
Gerarchia di memorie

Livelli multipli di memorie con diverse dimensioni e velocità.

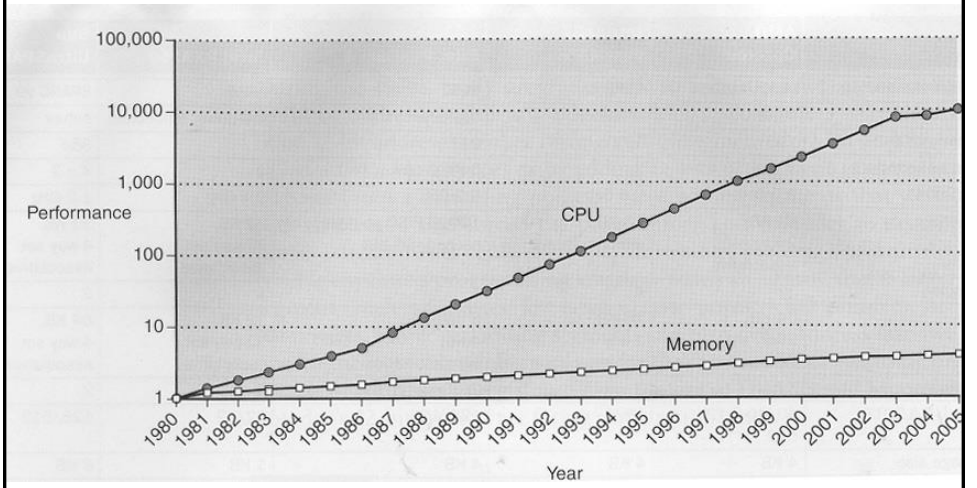
Nel livello superiore troviamo un sottoinsieme dei dati del livello inferiore.

Ciascun livello vede il livello inferiore e viceversa.

Cache (memoria nascosta)



Prestazioni processore vs memoria principale



Prestazioni misurate in tempo per completare un'operazione sulla memoria
Utilizzo diretto della memoria principale da parte della CPU provocherebbe tantissimi stalli

Criteri di progettazione

Parametri di definizione della struttura di una cache:
- Capacità
- Grado di associatività

Cache primaria: massimizzo Hit rate (minimizzo Miss rate)

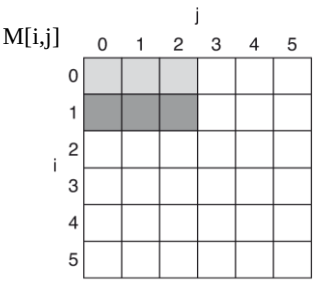
Cache secondaria: minimizzo Miss penalty (massimizzo transfer rate)

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 matici vengono scomposte in operazioni su sotto-matrici o blocchi (**blocks**).



Il contributo del SW: Blocking

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

- Supponiamo una matrice M n x 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 (n = 32, avremo 32 x 32 x 8 x 3 = 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.

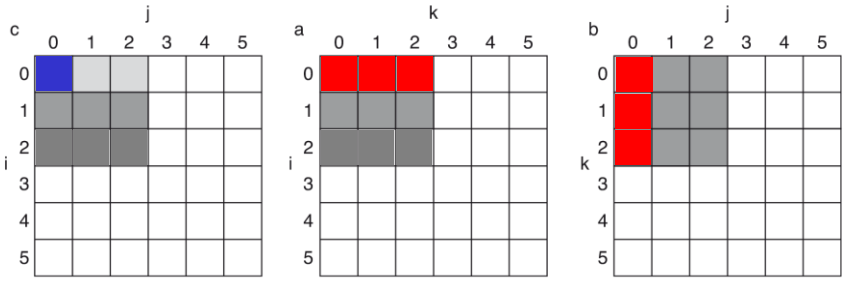
Le operazioni vengono eseguite sequenzialmente su blocchi di matrici, contenuti in memoria per tutta la durata delle operazioni sul blocco.

Blocking all'opera

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

....

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



9 DGEMM senza scaricare dati

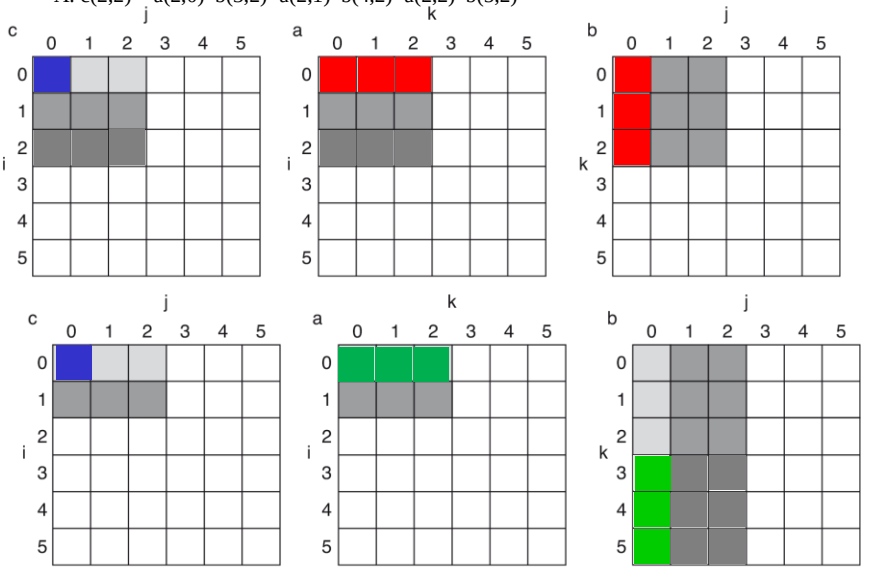
Blocking all'opera

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

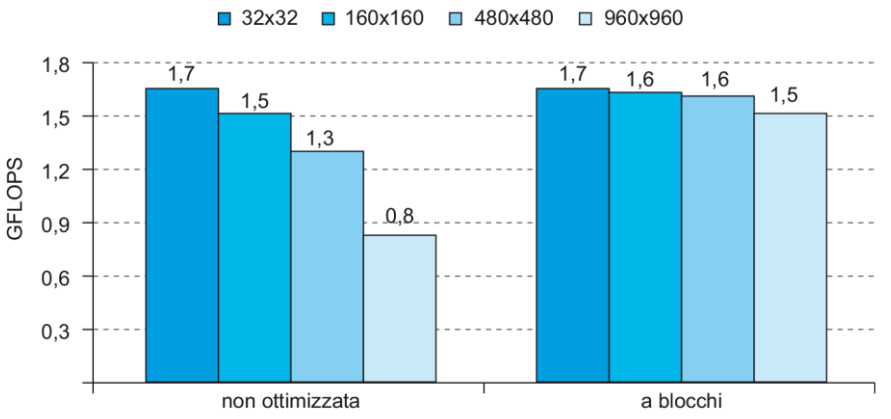
....

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

9 DGEMM in cui
Sostituisco solo B



Aumento delle prestazioni



Sommario

Ottimizzazioni software

SRAM

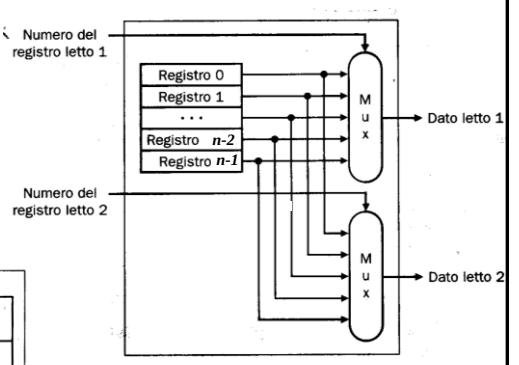
DRAM

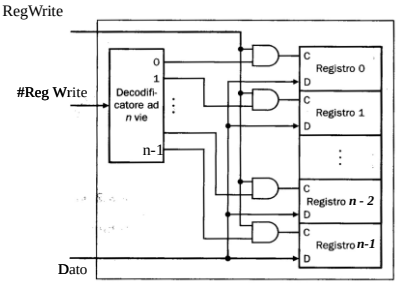
Register file

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).

SRAM oggi

Static RAM. Bistabili (4-6 **transistor** per cella, ottimizzazione della cella rispetto al latch, cf. Register File).

Tempo di accesso uguale per ogni dato.

Informazione stabile (non ci sono disturbi da parte di quello che succede intorno)

Non c'è bisogno di rinfrescare il contenuto della memoria (refresh)

Sono memorie volatile (dipendono dall'alimentazione)

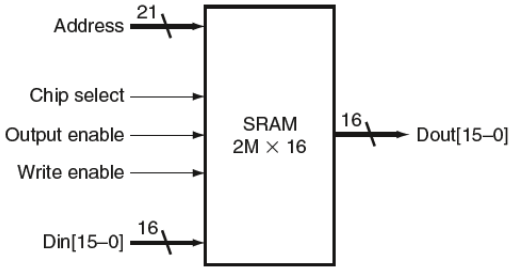
Con la tecnologia CMOS, consumano energia solo quando commutano.. Poca energia viene consumata per mantenere il dato (**standby**).

1 porta di lettura / scrittura (o leggo o scrivo o non faccio nulla: 2 segnali)

Cache -> inserita nella CPU -> Produzione delle SRAM è principalmente da parte dei produttori di CPU.

Memorie veloci per dispositive embedded.

Un chip di SRAM



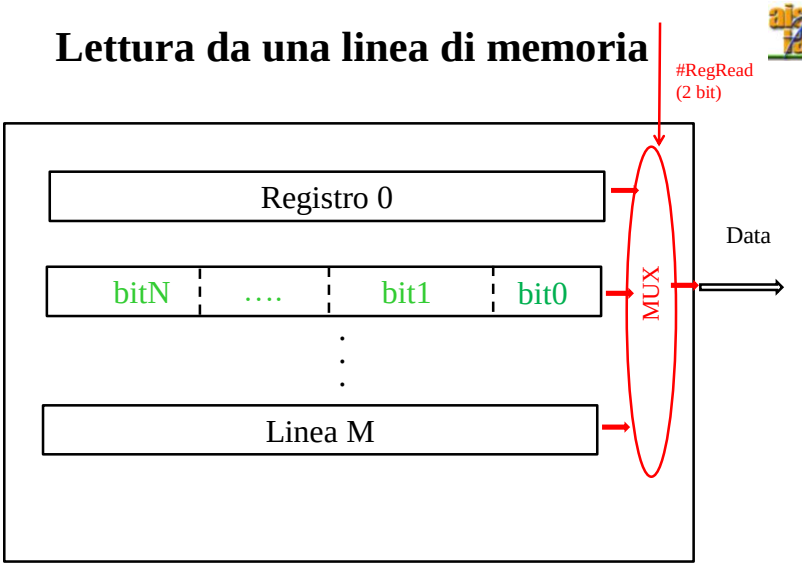
Altezza (2 Mbit) x Ampiezza (16 bit) – Oggi 1-4 bit

Altezza (2 Mbit) ➔ 21 bit di indirizzamento
Ampiezza (16 bit) ➔ Bus dati su 16 bit.

C**hip select -> Abilitazione del chip in uscita.**

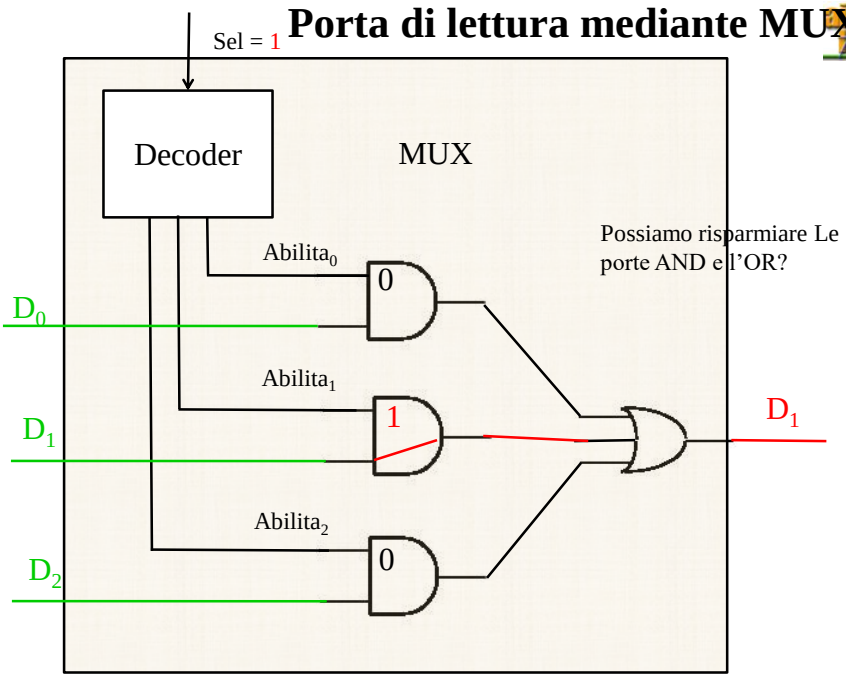
Write enable, Output enable (per risparmiare energia).

Lettura da una linea di memoria



Seleziono uno dei registri = porto in uscita l'uscita di tutti i bit del registro selezionato.
Il mux è pratico per 32 registri, ma non per migliaia di linee come nelle cache L1-L3.

Porta di lettura mediante MUX

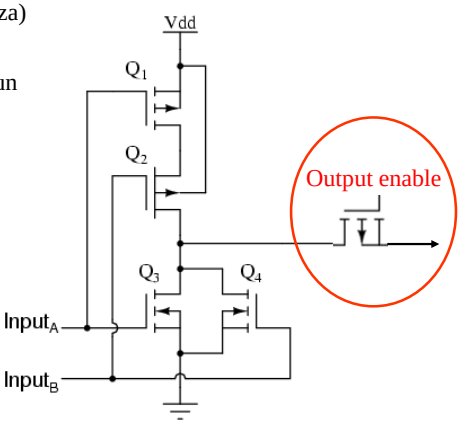


Possiamo risparmiare Le porte AND e l'OR?

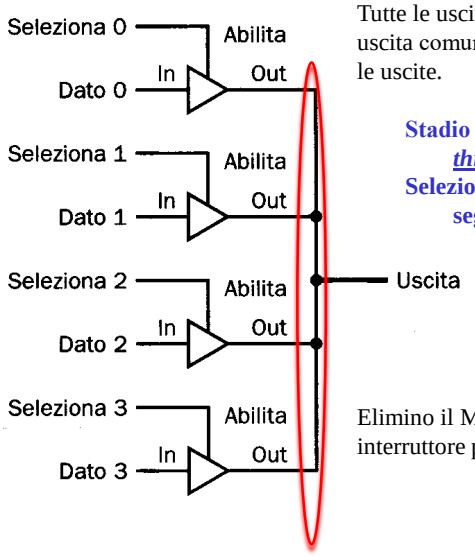
Porte three-state

- **Stato 0** – spento – connesso a massa (0V)
- **Stato 1** – acceso – connesso all'alimentazione (+Vcc)
- **Stato disabilitato** – disconnesso – non connesso a nulla, tensione floating (alta impedenza)

I circuiti a valle vedono l'equivalente di un "filo tagliato".



Memoria three-state (soluzione HW)

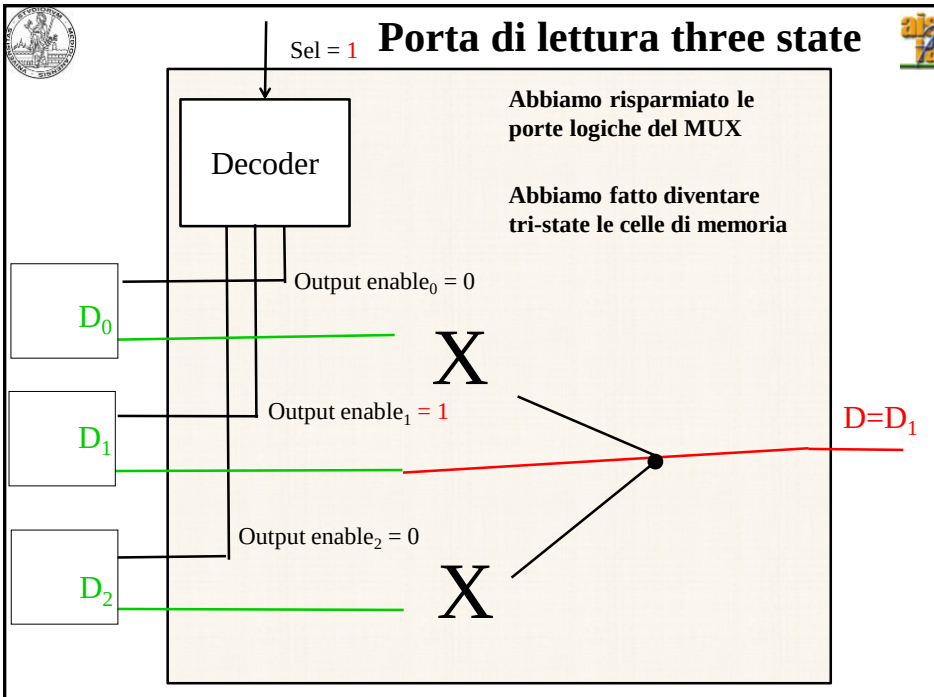


Tutte le uscite delle celle sono collegate ad un'unica uscita comune => E' necessario evitare conflitti fra le uscite.

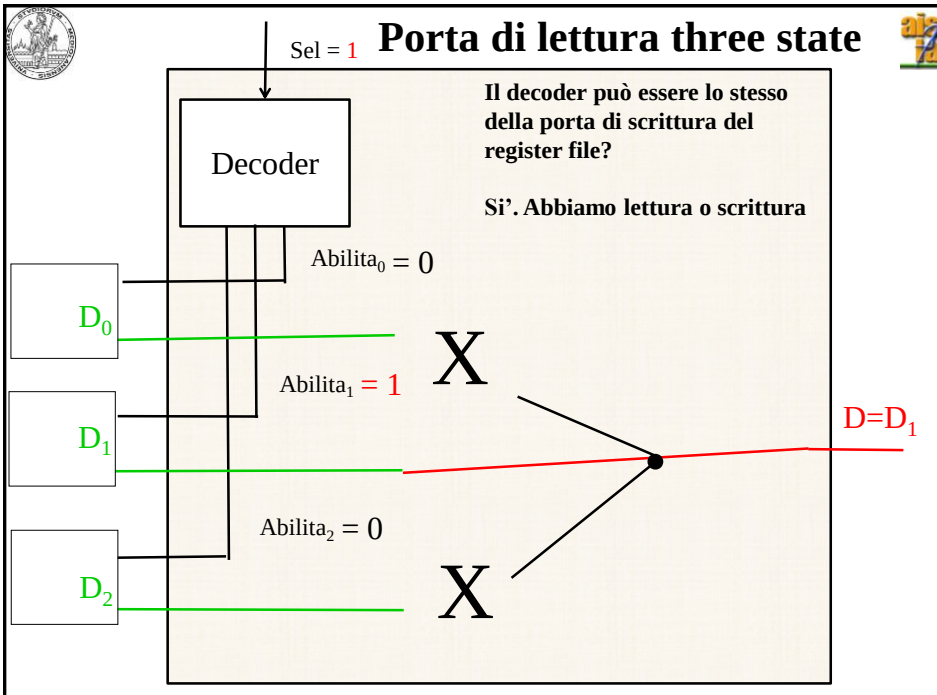
Stadio di uscita "isolata" con porte di uscita *three-state*
Seleziono una sola cella alla volta mediante segnale di abilitazione.

Elimino il MUX -> sostituito da un transistor-interruttore per bit, che lavorano in parallelo.

Porta di lettura three state



Porta di lettura three state



Esempio di SRAM in scrittura

Risparmio il Mux di uscita

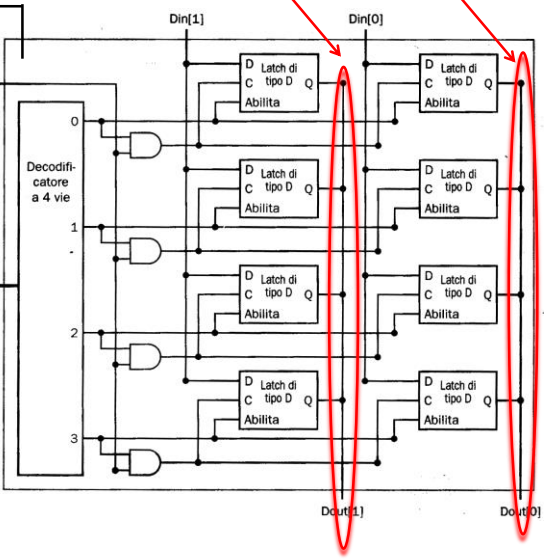
Chip enable

Write enable

Esempio: SRAM
4 elementi di memoria
x 2 bit ciascuno

Indirizzo -> Output enable
Abilito sempre un'uscita!

Chip select?



Il decodificatore è una schiera di AND che lavorano in parallelo: meno complesso e più veloce di un Mux. **Not enough!**

SRAM a matrice

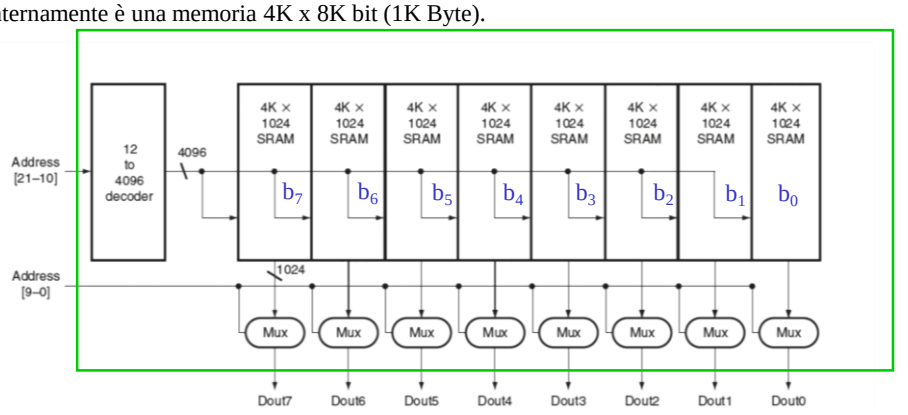
Una SRAM 4M x 8 avrebbe bisogno di un decoder di 22 -> 4M linee, con 4M porte AND ciascuna con 22 bit in ingresso.

C'e' un limite elettrico al numero di linee che si possono collegare assieme.

E' più conveniente costruire una matrice e separare la lettura delle linee dalla lettura delle colonne (estrazione di una linea «lunga» dalla memoria – cf. Memoria cache).

SRAM a matrice

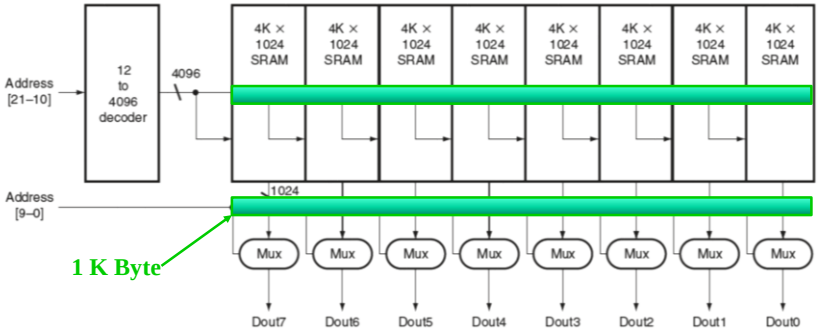
SRAM 4M x 8. Ad ogni bit assegno un chip (banco) 4K x 1024 (1024 bit per linea) => internamente è una memoria 4K x 8K bit (1K Byte).



Il decodificatore sarà a 12 bit ($\log_2 4096$) per selezionare una delle 4096 linee (ciascuna di 1024 bit) per ogni banco di memoria. Ciascuna linea fornisce 1024 bit. Ne seleziono uno con il Mux (controllato dai 10 bit meno significativi). Questo in parallelo su tutti i banchi.

Nell'approccio non a matrice avrei avuto bisogno di un decodificatore a 22 bit ($\log_2 4M$).

Modalità burst

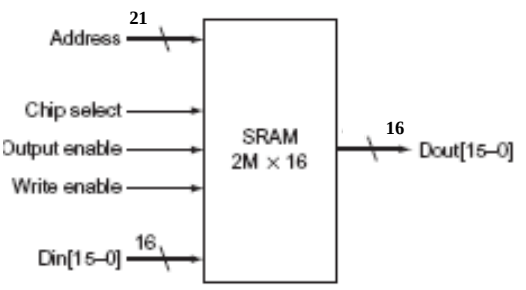


Leggo prima una intera linea e poi attraverso i mux di uscita leggo la parola (1 Byte)
Lettura gerarchica in 2 passi.

Syncronous Static RAM (SSRAM) -> trasferisco **burst**.
Burst: indirizzo iniziale + lunghezza (e.g. indirizzo del primo elemento della linea + 1K di lunghezza al Massimo = tutte le parole della linea selezionata).

- Non viene richiesto di specificare un nuovo indirizzo per ogni parola -> viene generato automaticamente.
- E' una tecnica potente per trasferire blocchi di memoria.

Chip di SRAM (2M x 16)

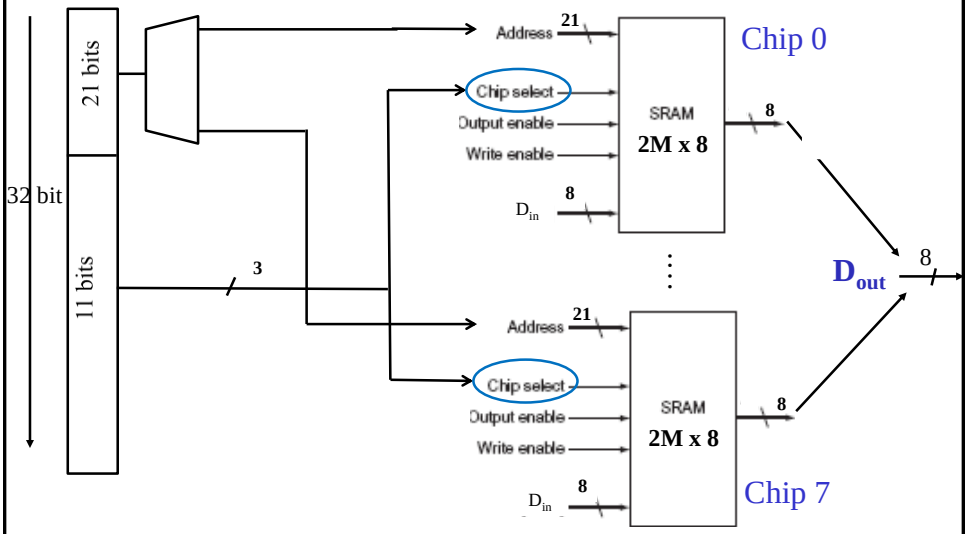


Tempo di accesso:
da Address a Dout.

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

Memoria SRAM come insieme di chip

Capacità totale = 8 Blocchi * 2 MByte => Capacità di 16 MByte

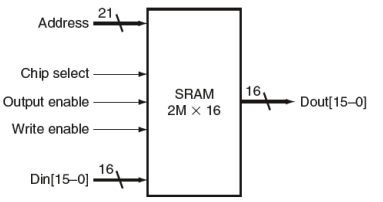


Principio di località: i dati all'interno di un chip verranno utilizzati negli istanti di tempo successivi. Abilitazione e disabilitazione sono eventi relativamente poco frequenti.

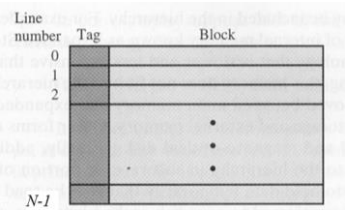
Esempio

Memoria cache ad accesso diretto con linee di 4 word e word di 4 Byte.
Costruzione mediante 32 chip di SRAM da 2 M x 16
lw $t1, 172($0)

Chip di SRAM



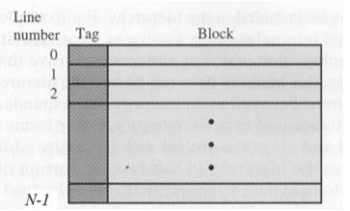
Ampiezza 2 Byte



Ampiezza = 4x4 Byte = 16 Byte

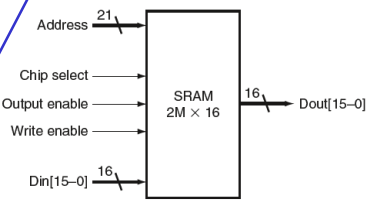
Dimensionamento

Memoria cache ad accesso diretto con linee di 4 word e word di 4 Byte.
Costruzione mediante 32 chip di SRAM da 2 M x 16 = 128 Mbyte in totale
lw $t1, 172($0)



Ampiezza = 4x4 Byte = 16 Byte

Chip di SRAM

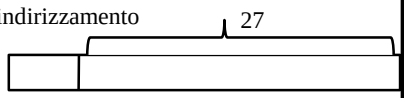


Ampiezza 2 Byte

Servono 8 chip di SRAM affiancati per realizzare una linea di cache.
Ho a disposizione 32 / 8 = 4 schiere di chip => 2M linee x 4 = 8 M linee
Cache sarà di 8M x 16 Byte = 128 Mbyte
I chip saranno disposti in una matrice 8 x 4.

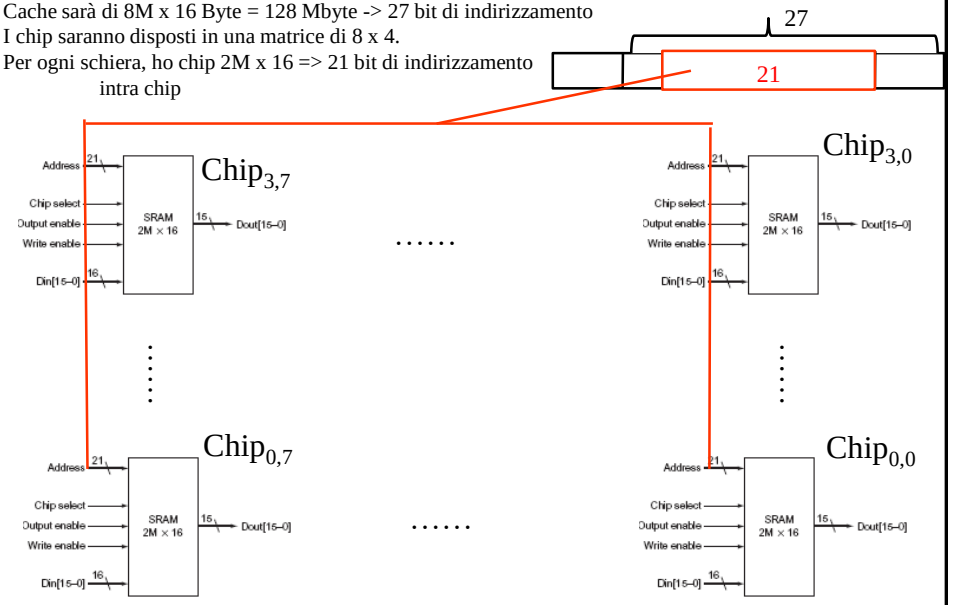
Indirizzamento

Cache sarà di 8M x 16 Byte = 128 Mbyte -> 27 bit di indirizzamento
I chip saranno disposti in una matrice di 8 x 4
(4 schiere di 8 chip)



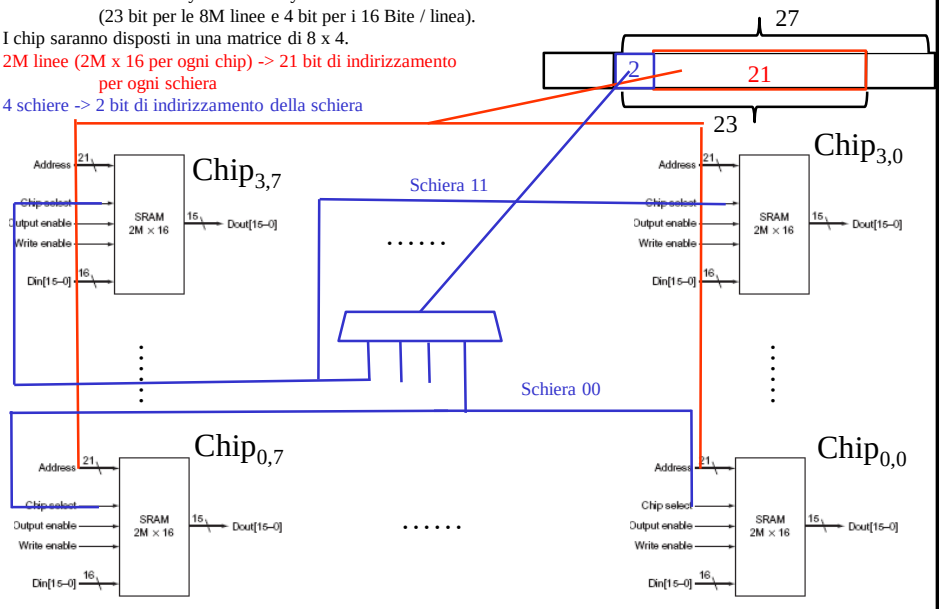
Selezione della linea

Cache sarà di 8M x 16 Byte = 128 Mbyte -> 27 bit di indirizzamento
I chip saranno disposti in una matrice di 8 x 4.
Per ogni schiera, ho chip 2M x 16 => 21 bit di indirizzamento intra chip



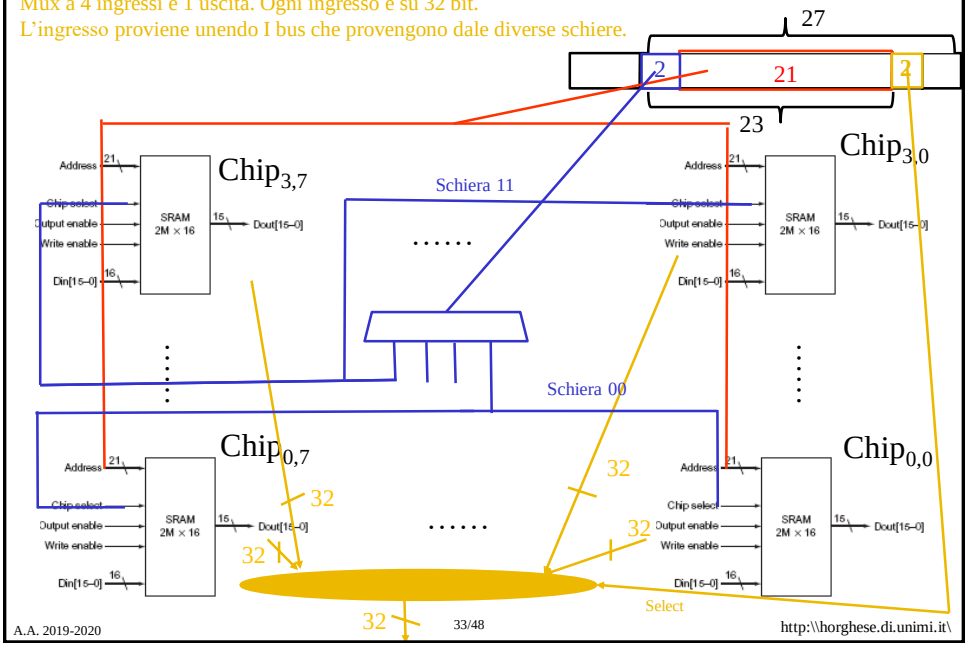
Selezione della schiera di chip

Cache sarà di 8M x 16 Byte = 128 Mbyte -> 27 bit di indirizzamento
(23 bit per le 8M linee e 4 bit per i 16 Bite / linea).
I chip saranno disposti in una matrice di 8 x 4.
2M linee (2M x 16 per ogni chip) -> 21 bit di indirizzamento per ogni schiera
4 schiere -> 2 bit di indirizzamento della schiera



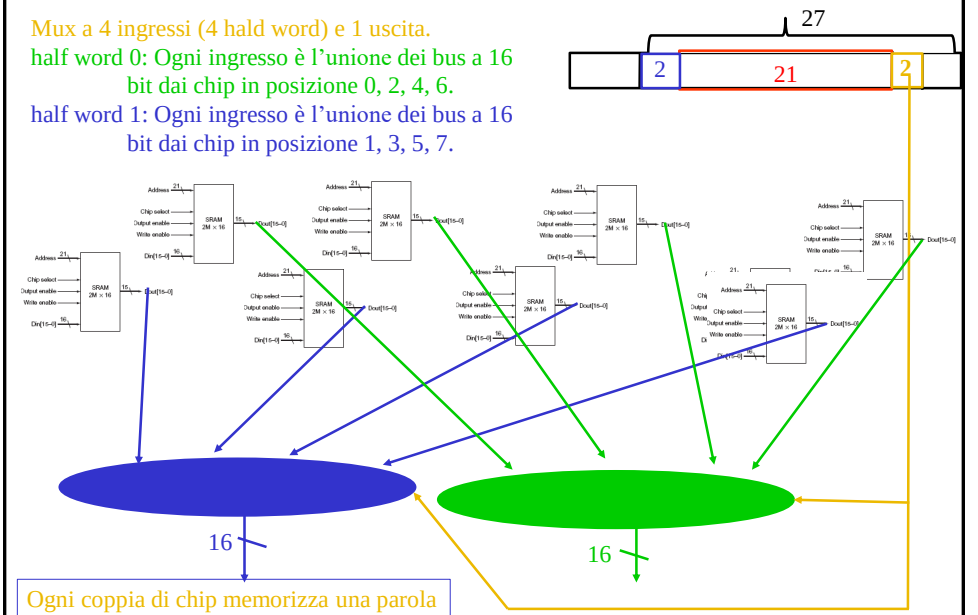
Selezione della colonna

Mux a 4 ingressi e 1 uscita. Ogni ingresso è su 32 bit.
L'ingresso proviene unendo I bus che provengono dale diverse schiere.



Selezione della Colonna: dettaglio

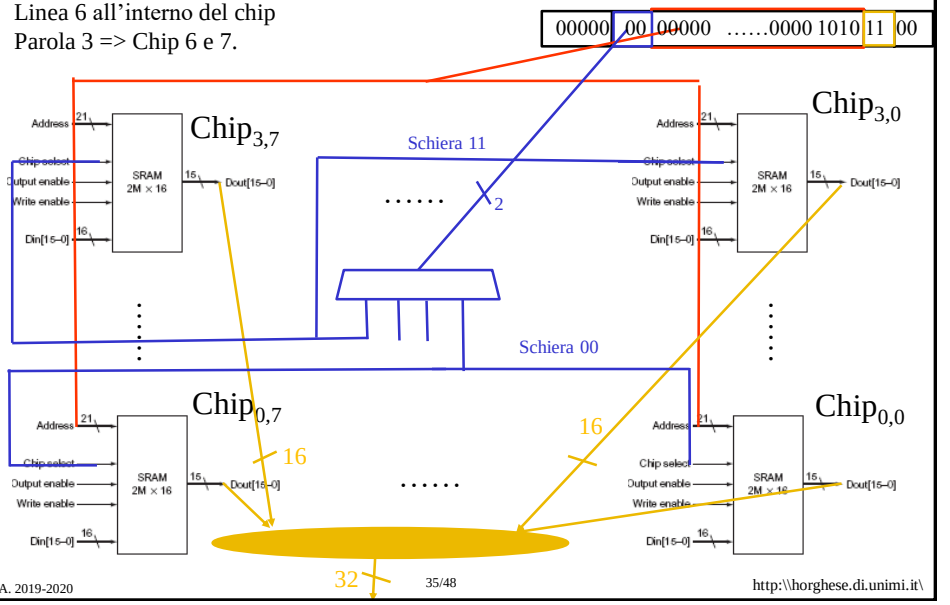
Mux a 4 ingressi (4 hald word) e 1 uscita.
half word 0: Ogni ingresso è l'unione dei bus a 16 bit dai chip in posizione 0, 2, 4, 6.
half word 1: Ogni ingresso è l'unione dei bus a 16 bit dai chip in posizione 1, 3, 5, 7.



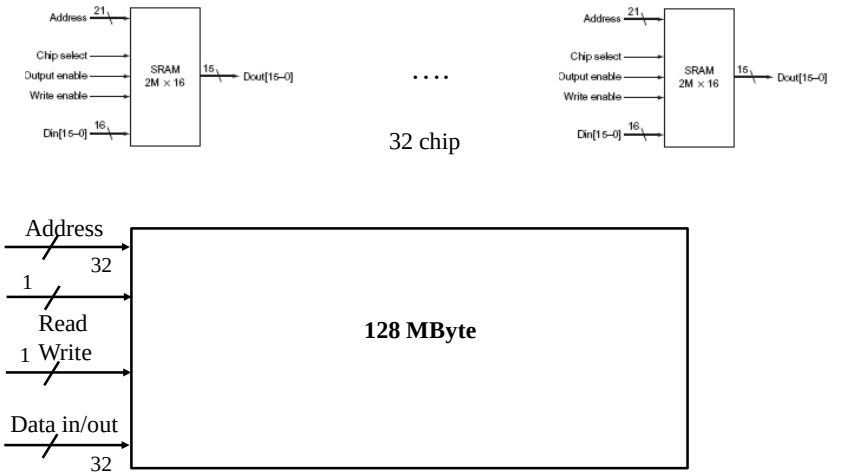
Ogni coppia di chip memorizza una parola

Posizione della parola

lw $t1, 172($0)
Schiera 0 dei chip
Linea 6 all'interno del chip
Parola 3 => Chip 6 e 7.



Indirizzamento fisico e logico



Sommario

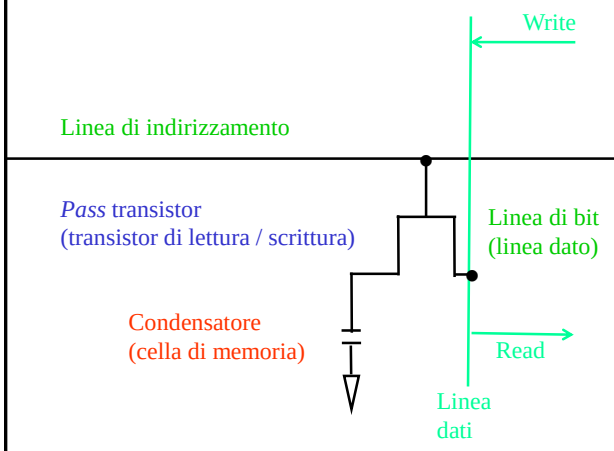
Ottimizzazioni software

SRAM

DRAM

Memorie DRAM

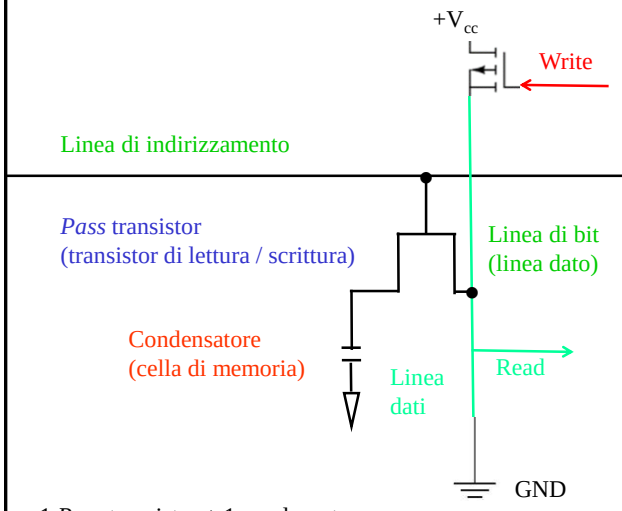
Dynamic RAM. 1 transistor per bit (contro 4-6 transistor della SRAM).



1 *Pass* transistor + 1 condensatore.

.

Memorie DRAM - scrittura



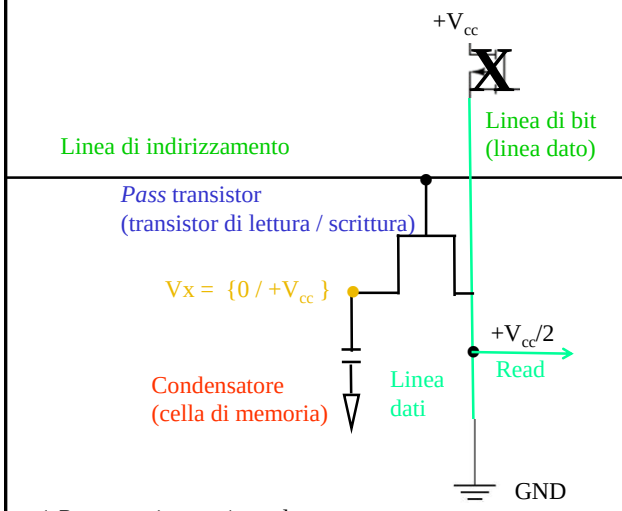
Scrittura: carica / scarica

La scrittura:
carica il condensatore da Vcc (memoria = 1) o
scarica il condensatore a GND (memoria = 0)

1 *Pass* transistor + 1 condensatore.

.

Memorie DRAM - lettura



Lettura:

Se Vx > $V_{cc}/2$ la corrente è **entrante** nella line Read

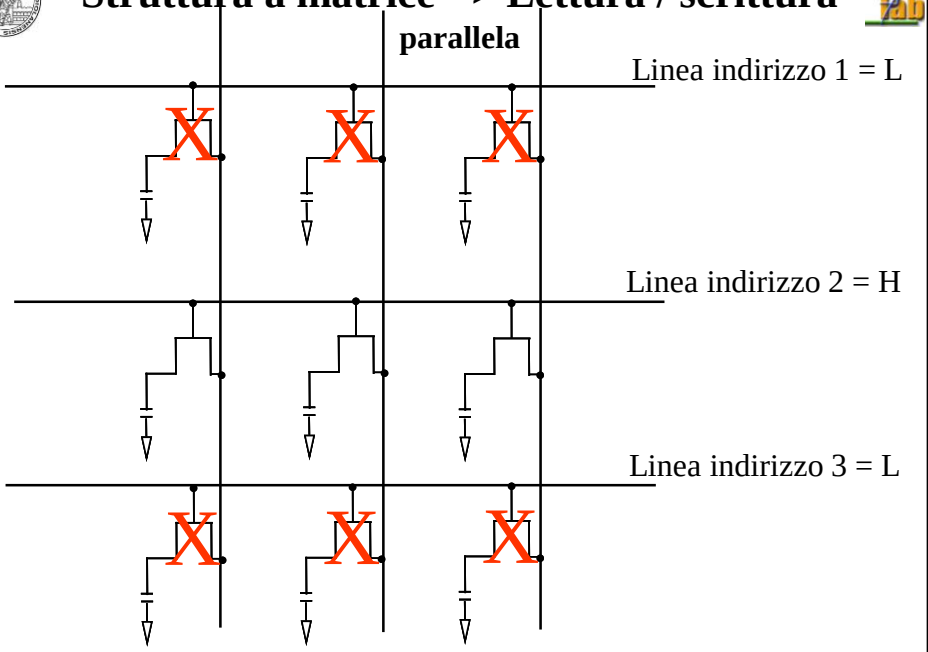
Se Vx < $V_{cc}/2$ la corrente è uscente dalla linea read.

=> Le variazioni di segno della corrente vengono trasformate in 0 / 1 da un amplificatore differenziale

1 *Pass* transistor + 1 condensatore.

.

Struttura a matrice => Lettura / scrittura



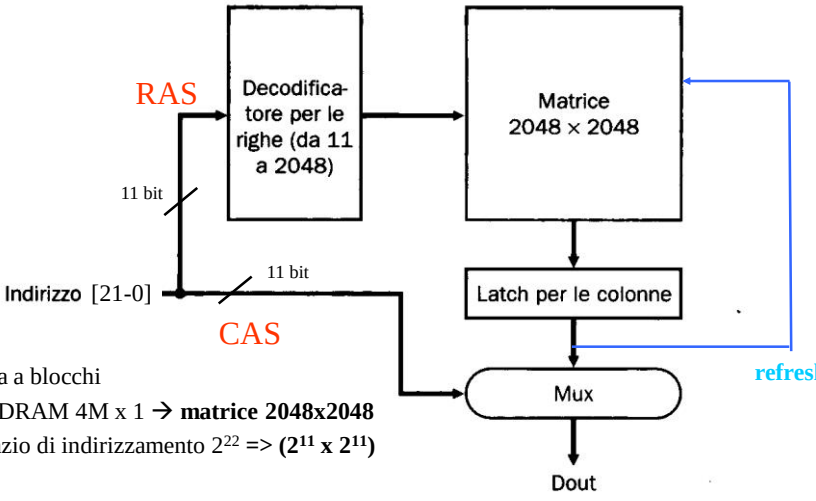
I problemi delle DRAM

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} => **(2^{11} x 2^{11})**

•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.

Osservazioni

Latch di Colonna (MDR)

- Refresh
- Trasferimento in modalità burst (solo MUX di uscita, viene nascosta la latenza di lettura)
- Modalità burst: indirizzo iniziale + lunghezza del burst.
- Blocking aiuta.

SDRAM (Syncronous Dynamic RAM) sono memorie sincronizzate. I latch sono sincronizzati.

- Esiste un tempo certo per la scrittura e la lettura globale.
- Non c'è bisogno di procedure di conferma (hand-shaking).
- Il processore può fare altro.

DDR SDRAM (Double Data Rate SRAM). La sincronizzazione del funzionamento avviene su entrambi i fronti del clock. Velocità di trasferimento di picco di 3,2 Gbyte/s.

Specifiche delle DRAM

Anno introduzione	Dimensione chip	$ per GiB	Tempo accesso totale a nuova riga / colonna	Tempo medio di accesso alla riga esistente
1980	64 Kibibit	$1 500 000	250 ns	150 ns
1983	256 Kibibit	$ 500 000	185 ns	100 ns
1985	1 Mebibit	$ 200 000	135 ns	40 ns
1989	4 Mebibit	$ 50 000	110 ns	40 ns
1992	16 Mebibit	$15 000	90 ns	30 ns
1996	64 Mebibit	$10 000	60 ns	12 ns
1998	128 Mebibit	$ 4000	60 ns	10 ns
2000	256 Mebibit	$ 1000	55 ns	7 ns
2004	512 Mebibit	$ 250	50 ns	5 ns
2007	1 Gibibit	$ 50	45 ns	1,25 ns
2010	2 Gibibit	$ 30	40 ns	1 ns
2012	4 Gibibit	$ 1	35 ns	0,8 ns

Tempo totale di accesso (dall'arrivo dell'indirizzo al dato in uscita): 35 ns
Tempo di accesso in modalità burst (accesso al buffer di linea di uscita): 0.8 ns

40 volte più veloce!

Esercizio

Memoria cache ad accesso diretto con linee di 8 word e word di 8 Byte.
Costruzione mediante 64 chip di SRAM da 1 M x 16
lw $t1, 1452($0)

Organizzazione della memoria

Organizzazione a matrice (cf. cache)

Organizzazione gerarchica.

Sommario

Ottimizzazioni software

SRAM

DRAM