

LA RAM

025 50:56

L'organizzazione logica della memoria è PIATTA! È UN modello piatto, ossia posso vedere la mia memoria come un "VECTORE" di byte, in cui ciascuna cella grossa 1 byte è identificata da un indirizzo.

Se ho un vettore di 32 celle avrò bisogno di 5 bit per definire il mio indirizzamento:

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	1	1	0	0	0	1	1	1	1	0	0
1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	1	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	0	0
1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	1	0	0	1	1	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

DATO UN INDIRIZZO, IO PRECISO AD ANDARE esattamente ad 1 byte!

QUESTA ASTRAZIONE SCHIACCHIA nel momento in cui andiamo a ragionare che io della memoria non leggo un solo byte e fijo ad ora, nel corso, NON ABBIANO mai fatto questa astrazione, perché dicevamo: "vado ad indirizzo e da lì leggo 1, 2, 4, 8 byte, ossia la dimensione dei tipi primitivi"

Abbiamo visto che se io do l'indirizzo sull'address bus, mi viene fornito il dato sul data bus

Questo significa che se io do l'indirizzo 00001, la memoria mi tirerà fuori questo byte.

Se voglio leggere più byte, con questa organizzazione logica, io dovrei passare più indirizzi! (x volta). Perché i vari byte sono associati ai vari indirizzi!

QUINDI QUESTA ORGANIZZAZIONE NON PUÒ FUNZIONARE!

Perché ad esempio per leggere 4 byte, io non voglio fare 4 accessi in memoria!
NE VOGLIO FARE 1 SOLO.

AUORA TIPICAMENTE le memorie vengono organizzate in moduli!
che cosa è un modulo? È UN insieme di byte!

Le 32 celle di memoria di prima, vengono organizzate in maniera tale da stare su moduli differenti.

• Problema:
• il processore vuole leggere dati di 1, 2, 4, 8 byte
• la memoria è indirizzata al byte
• Soluzione:
• organizzazione in moduli
• possibilità di recuperare più byte in parallelo
• I tre bit meno significativi identificano il modulo, quelli più significativi la riga

L'indirizzo 0 si troverà sul primo modulo.

L'indirizzo 1 sul secondo.

È possibile scegliere un'organizzazione a 8 moduli!

a ₄ a ₃ a ₂ a ₁ a ₀	7	a ₄ a ₃ a ₂ a ₁ a ₀	6	a ₄ a ₃ a ₂ a ₁ a ₀	5	a ₄ a ₃ a ₂ a ₁ a ₀	4	a ₄ a ₃ a ₂ a ₁ a ₀	3	a ₄ a ₃ a ₂ a ₁ a ₀	2	a ₄ a ₃ a ₂ a ₁ a ₀	1	a ₄ a ₃ a ₂ a ₁ a ₀	0		
0 0 1 1 1	15	0 0 1 1 0	14	0 1 1 1 0	22	0 1 1 1 0	21	0 0 1 0 1	13	0 1 0 1 1	20	0 1 0 1 0	19	0 0 0 1 1	11	0 1 0 0 1	10
0 1 1 1 1	23	1 0 1 1 0	30	1 0 1 1 0	29	1 1 0 1 1	28	1 0 1 0 0	12	1 0 1 0 0	27	1 0 0 1 0	26	1 1 0 1 0	25	1 0 0 0 1	9
1 1 1 1 1	31	1 1 1 1 0		1 1 1 0 1		1 1 0 1 0		1 1 0 0 0		1 1 0 0 0		1 0 0 1 0		1 1 0 0 0		1 0 0 0 0	8

questo significa che se io prendo un indirizzo di memoria 00000, posso andare a prendere insieme 8 byte, selezionando tutte le righe che cominciano con 00 in tutti quanti i moduli!

Posso andare a specificare come indirizzo di riga "01", in quel caso succede che ho letto contemporaneamente tutte e 8 byte!

QUESTA ORGANIZZAZIONE mi permette di leggere contemporaneamente byte differenti andando ad utilizzare un unico indirizzo di partenza mascherando i 3 bit meno significativi.

MA INFESTO NON SONO GRADO DI SPARIRE ADRI 8 BYTE CONTEMPORANEAMENTE.

AD UTILIZZARE UN UNICO INDIRIZZO DI PARTENZA MASCHERANDO I 3 BIT MENO SIGNIFICATIVI.

IN QUESTO MODO SONO IN GRADO DI SPARARE FUORI 8 BYTE CONTENUTO RANCAMENTE.

MA, SE QUESTA ESTRACCIONE FOSSE VERA, IN REALTÀ NON MI BASTEREbbe, perché questa soluzione mi fa leggere 8 byte, ma io vorrei poterne leggere anche 4 di byte.

Questa organizzazione mi permette di leggere solo la Taglia massima.

In generale, se io ho N moduli, quello che vorrei poter fare è abilitare la lettura o la scrittura soltanto da un sottosistema!

Per esempio, se io sto leggendo soltanto 4 byte, voglio che una lettura avvenga dal modulo 0, modulo 1, modulo 2 e modulo 3.

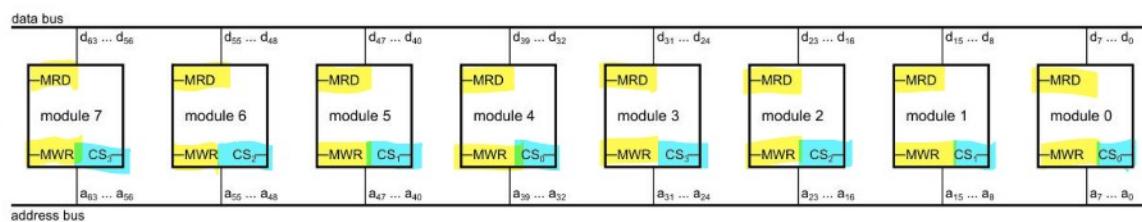
L'ADDRESS BUS è da 8 byte ossia 6 bit, e se io sto scrivendo 1 byte solo?

Voglio poter dire soltanto al modulo zero di leggere i dati dal DATA-BUS per effettuare una SCRITTURA!

Al DATA BUS e all'ADDRESS BUS così come li abbiamo studiati, dobbiamo aggiungere dei segnali di

Chip Select (CS)

In funzione della scrittura o della lettura che voglio andare ad effettuare, soltanto un sottosistema dei moduli verrà effettivamente pilotato.



Posso accedere solo a 4 moduli per leggere 4 byte.

Deve ARRIVARE anche un'informazione della dimensione del dato che voglio leggere, cosicché i segnali corretti di chip select vengano ABILITATI per permettere la lettura o la SCRITTURA.

ORA, SE IO POSSO ANDARE AD INDIRIZZARE UNA MEMORIA AL BYTE E POSSO DEFINIRE UNA STRUCT DI QUALSIASI TIPO, CHI MI GARANTISCE CHE IL DATO CHE VOGLIO ANDARE A LEGGERE, PARTA DAL Modulo 0 ???

4 BYTE; I PRIMI due SU UNA RIGA e gli ultimi due sulla successiva : potrai avere questa situazione; Anche 2 byte possono essere disallineati.

IN questa situazione l'hardware deve effettuare più letture e ricombinare i dati!

LA MEMORIA DISALLINEATA È UN PROBLEMA!

NON VOGLIO FARE PIÙ ACCESSI PER LEGGERE UN INTERO!

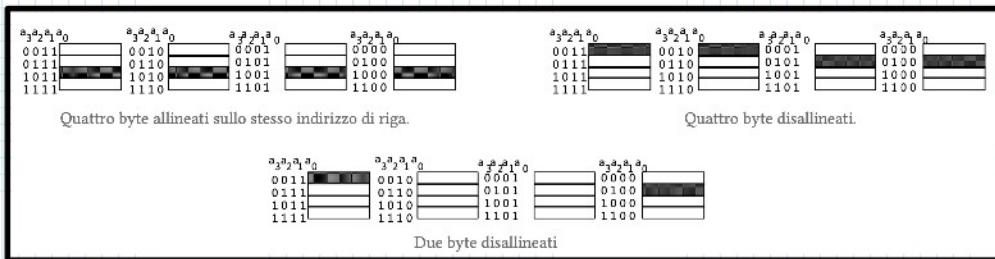
LA SOLUZIONE È IL PADDING!

Si cerca di riempire con le informazioni doppie, in modo che

LA SOLUZIONE E IL PADDING!

Si cerca di riavvicinare le letture delle informazioni da memoria!

Se accedo due volte per leggere due righe in memoria, sto spendendo il doppio del tempo;
ovviamente rialineando i dati ci metterà più tempo!

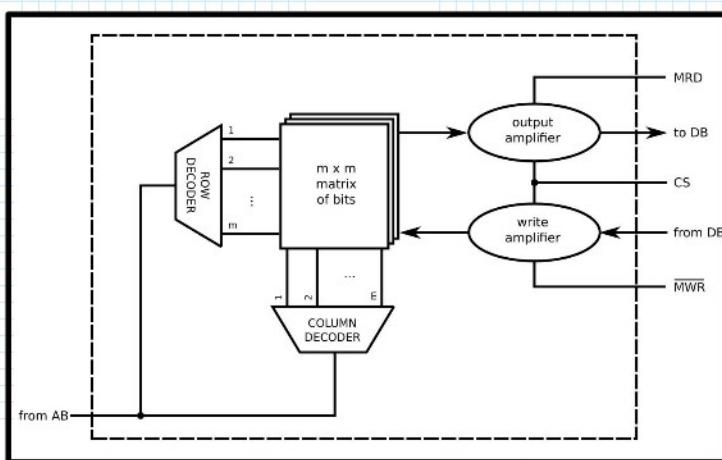


In realtà, anche questa organizzazione a moduli rischia abbastanza bene come veleggi realizzate concettualmente le RAM.

Ma avere un'organizzazione a moduli di questo tipo, dal punto di vista dello spazio, è molto costoso;

QUINDI COME VENGONO REALIZZATE LE RAM?

Sono organizzate in matrici, di $N \times N$ bit! che rappresentano un pochettino queste singoli moduli!

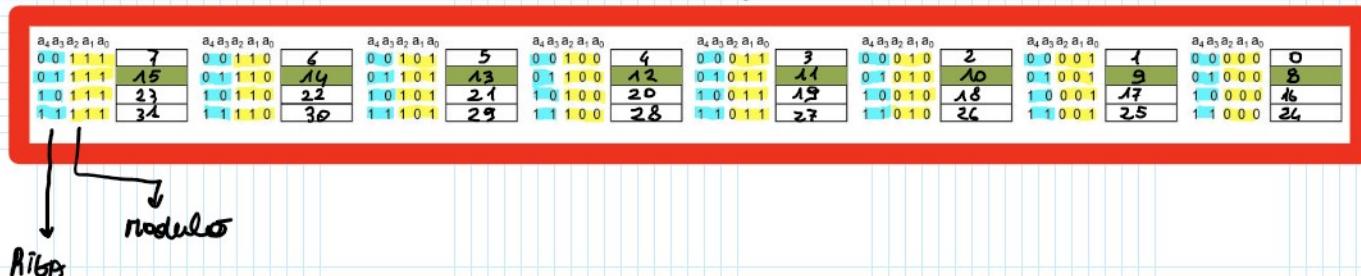


ALL'INTERNO DI UN BANCO DI MEMORIA, ABBIAMO K DI QUESTE MATRICI! (una dopo l'altra)

Utilizzo un decoder di riga per andare a selezionare una riga su tutte quelle che ha le matrici.

Utilizzo un decoder di colonna per selezionare quale di queste pagine della memoria RAM, è contenuta in una lettura o in una scrittura;

IL DECODER DI COLONNA VIENE PILOTATO DALL'ADDRESS BUS, L'AB mi manda l'indirizzo, questo indirizzo viene decomposto in un certo numero di bit per pilotare il suo decoder di riga e gli altri bit possono essere usati per andare a selezionare effettivamente la colonna (quali dei moduli sono interessanti per un op. di lettura).



I dati da e verso il DATA BUS vengono amplificati e i segnali di controllo, insieme all'ADDRESS BUS determineranno se sto leggendo o scrivendo una o più colonne di una determinata riga!

Però immaginiamo se io voglio leggere una sequenza di dati contigui e magari voglio

www.ancientscript.com Rigv.

PERÒ immaginiamo se io voglio leggere una sequenza di dati contigui e magari voglio leggerle un rettangolo di dimensioni.

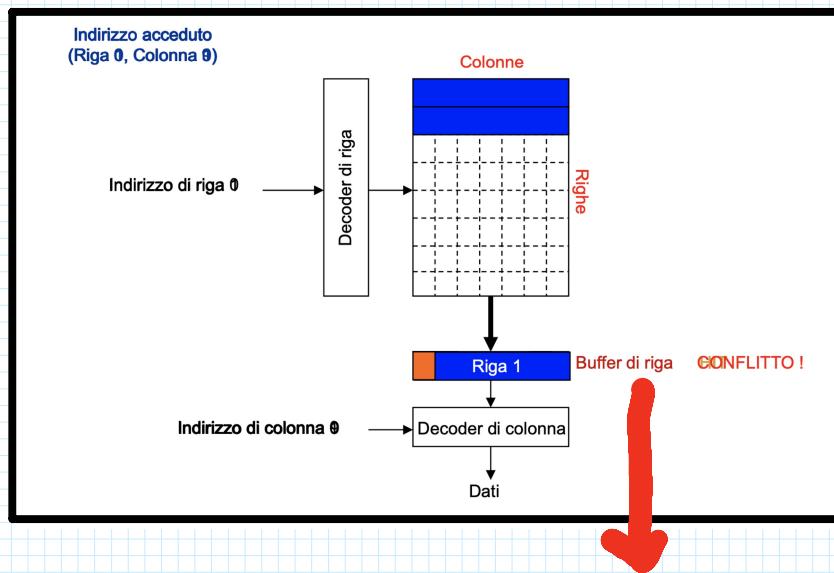
Mi leggo il primo carattere: mando l'indirizzo di riga, mando l'indirizzo di colonna e mi tira fuori il mio byte, poi ripetore mi leggo il secondo byte del mio vettore.

LA RIGA NON È CAMBIATA, STO SEMPRE NELLA RIGA 0.

Quindi se io mi limiterò ad accedere al singolo byte significa che io dovrò rimettere un altro indirizzo per una colonna successiva. Non è bene!

Io NON devo continuare a leggere qui volta le stesse Riga se ad esempio sto scandendo un rettore, se effettivamente l'ho già letto!

COSA FACCIO? REGISTRO TAMPONE!



Metto subito un Registro Tempore!
così faccio la copia delle Regole!

Ad un certo punto il processore vuole leggere un byte (Riga 0, colonna 0). Il decoder di Riga si prende dall'address bus l'indirizzo della riga e seleziona la riga.

QUESTA RIGA VIENE LETTA E COPIATA NEL BUFFER DI RIGA.

Il decoder di colonna quindi non legge i dati direttamente dalla memoria, ma li legge dal Buffer di RIGA.

nel quarto caso il decoder di Rion prenderà il primo byte, lo prende e lo manderà sul DATA BUS.

CHE succede se devo leggere il Byte successivo della stessa Riga?

HO GIÀ LA RICARICA ZERO, QUINDI NON HA SENSO RICARICARLA!

E A QUESTO PUNTO MI BASTA PASSARE AL DECODER DI COLONNA L'ADIRIZZO DI COLONNA 1.

Così anche dopo il byte successivo esce mano sul data bus;

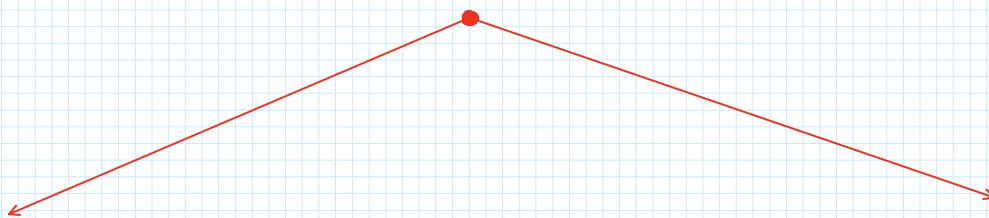
E così via, fino all'indirizzo di colonna 9.

SE ARRIVA UNA RICHIESTA PER LA RIGA 1, NEL BULLER HO LA RIGA 0: CONFUSO!

Dovrò riscrivere il contenuto del dossier di Riga della memoria;

Dov'è riservata la memoria di riga della memoria; fatto questo metto la colonna zero e lo mando sul databus!

Cochiammo di capire come è fatto uno di questi rettangolini qui.

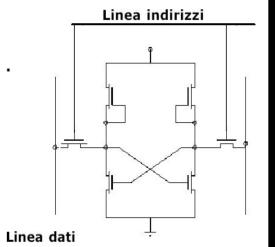


RAM STATICHE

RAM DINAMICHE

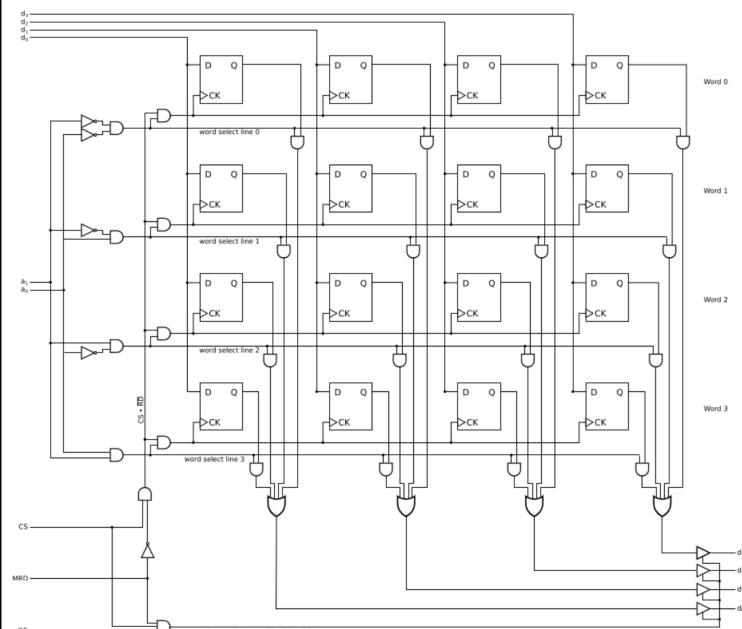
Le memorie RAM statiche

- La cella elementare è costituita da 6 transistori MOS che formano un flip-flop
- L'informazione permane stabile in presenza della tensione di alimentazione
- Tempi di accesso rapidi
- Costi elevati



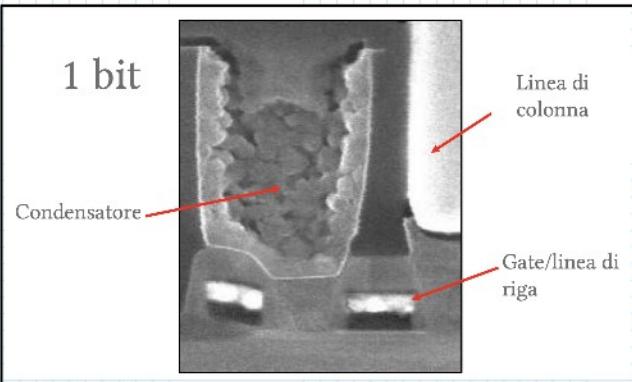
LA MATRICE SERVE PER MEMORIZZARE CIASCUN bit delle mie parole!

Organizzazione logica di una memoria RAM statica



- La matrice di bit è realizzata da un insieme di flip-flop
- I flip-flop sono circondati da un circuito combinatorio per la lettura/scrittura dei dati

Ma i flip flop sono troppi e li sostituiamo con dei condensatori;
DRAM



DRAM Refresh

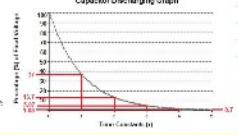
REFRESH

Il condensatore
si scarica

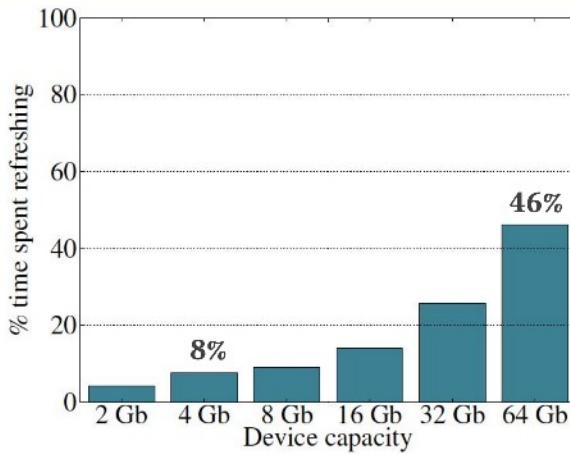
- Il controllore della memoria deve "rinfrescare" periodicamente tutte le righe per ripristinare la carica
- Valore tipico dell'intervallo: 64 ms
- Problematiche del refresh:
 - Consumo energetico*: ogni refresh consuma energia
 - Degradazione delle prestazioni*: durante il refresh, una riga non è disponibile
- Lunghe pause nella possibilità di accesso alla memoria durante il refresh
- Le dimensioni e le velocità delle memorie sono direttamente limitate dalla necessità di refresh

→ CONSUMO MOLTO CORRONE

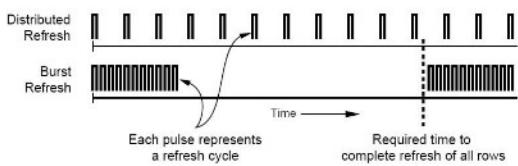
- Problema: i condensatori si scaricano naturalmente
- Costante temporale: il tempo necessario a far scaricare il condensatore del 63%
- Dopo questo tempo, l'informazione è persa
 - Non è più possibile distinguere tra 0 e 1



QUINDI SI FA UN REFRESH DISTRIBUITO!



- Refresh a burst: ogni 64ms si effettua il refresh di *tutte* le righe
- Refresh distribuito: le righe sono soggette a refresh in istanti temporali differenti



- Il refresh distribuito elimina (statisticamente) le lunghe pause

UN SINGOLO MODULO È QUESTO:

Interazione con la memoria

- Funzionalmente è caratterizzata dai seguenti segnali
 - Indirizzo della parola da leggere/scrivere
 - MR, affermato se si vuole leggere
 - MW, affermato se si vuole scrivere
 - CS, Abilita l'intero modulo (Chip Select)
 - Dati
- Il tempo richiesto dalla memoria per fornire un dato determina la *latenza di accesso*

