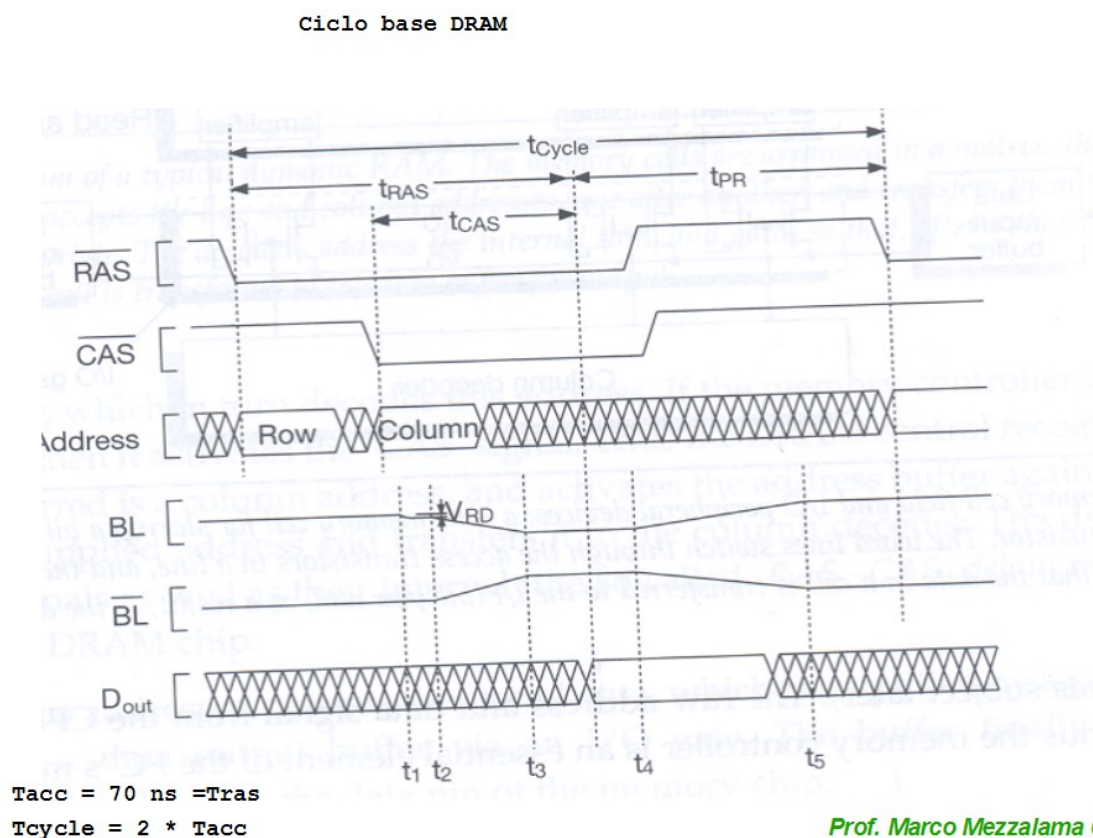


## LEZIONE 08/10/08 (slide memorie 6-21)

### Funzionamento delle memorie DRAM (slide 6-10)

Il funzionamento delle memorie DRAM prevede alcuni cicli: il ciclo di *read*, relativo alla lettura di un dato; il ciclo di *write*, corrispondente ad una scrittura; il ciclo di *refresh*, necessario per evitare di perdere i dati contenuti in memoria a causa della diminuzione della carica presente all'interno del condensatore di cui è costituita ogni singola cella di memoria; infine vi è il ciclo *fast operative*, il quale viene utilizzato per ottenere tempi di accesso più piccoli a dati consecutivi presenti in memoria, nella pratica permette di leggere tutte le celle associate ad una riga.

Grazie alla disposizione a matrice delle celle di memoria, il numero di linee utilizzate per l'indirizzamento delle celle di memoria è pari alla metà di quello che si avrebbe nel caso in cui queste fossero disposte in modo vettoriale: prima viene inviato sull'address bus l'indirizzo relativo alla riga da attivare e quindi in seguito quello relativo alla colonna (ed essendo la memoria organizzata come matrice, basta la metà dei piedini per poter trasmettere tali indirizzi), quindi i segnali notRAS e notCAS permettono di comprendere a cosa si riferisce l'indirizzo che viene trasmesso in quell'istante (in pratica vengono multiplexati sullo stesso bus gli indirizzi di riga e di colonna, con i relativi segnali di strobe per indicare quando il valore è stabile).



La slide 7 raffigura un ciclo base di una memoria DRAM. Il ciclo di memoria inizia quando il segnale RAS scende, che significa che da questo momento in poi sull'address bus è presente la riga della memoria da leggere. In seguito è posto sull'address bus il numero di colonna della memoria da leggere e viene abbassato anche il segnale CAS. Il tempo di accesso alla memoria è indicato con  $t_{ras}$  ed indica il tempo necessario, da quando viene iniziata la richiesta sul bus, per ottenere sul data bus i dati richiesti. Il ciclo nella memoria non termina però con il  $t_{ras}$  poiché segue una fase, di lunghezza  $t_{pr}$  (pre-charge time), in cui la memoria non è in grado di rispondere ad altre richieste. Il

tempo di pre-charge è dovuto a motivazioni tecnologiche relative alla memoria, visto che sono presenti amplificatori, condensatori, ed altri elementi che richiedono del tempo prima di poter essere nuovamente utilizzati. Il tempo di ciclo della DRAM è dunque pari a  $t_{\text{cycle}} = t_{\text{ras}} + t_{\text{pr}}$ , che complessivamente risulta circa doppio rispetto al tempo di accesso  $t_{\text{ras}}$ . In generale, se non si adotta un qualche meccanismo, il fatto che il tempo di ciclo è circa doppio rispetto al tempo di accesso peggiora sensibilmente le prestazioni della DRAM.

## Interleaving

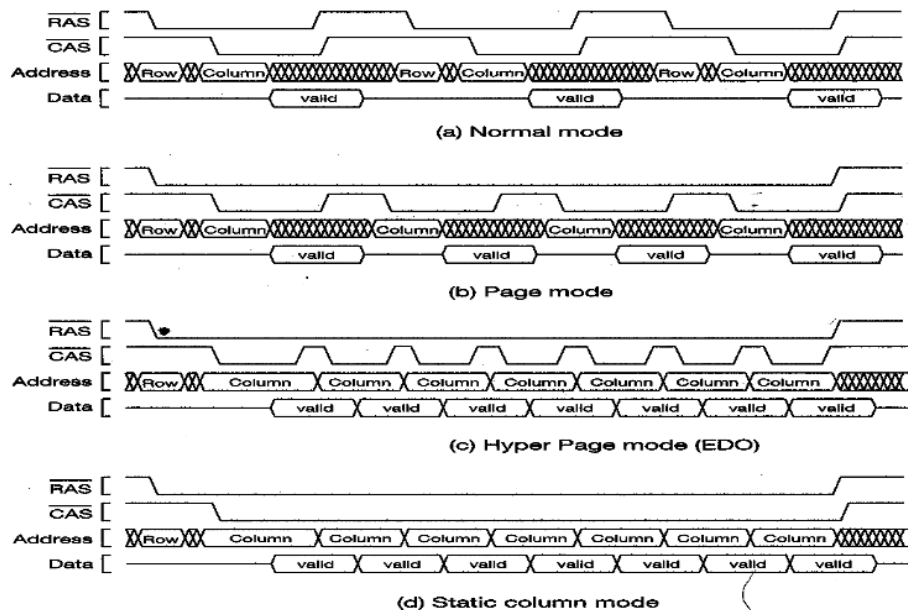
E' possibile eliminare l'impatto sulle prestazioni, dovuto al tempo di pre-charge, utilizzando l'*interleaving dei banchi di memoria*. Se non si adotta una soluzione, infatti, un primo tentativo di accesso alla memoria viene soddisfatto nel tempo  $t_{\text{ras}}$  ma il successivo accesso, se richiesto immediatamente dopo, deve attendere il tempo  $t_{\text{pr}} + t_{\text{ras}}$  per essere soddisfatto. La soluzione dell'interleaving prevede la presenza di più di un banco o chip di memoria; nell'ipotesi di accessi alla memoria sequenziali, si indirizzano i byte di modo che il blocco successivo di lettura, come indirizzo, si trovi nel chip successivo, eliminando quindi l'attesa dovuta al pre-charge. Ad esempio si possono ipotizzare due chip di memoria e letture di un byte alla volta, in questa configurazione al primo chip sarebbero associati tutti gli indirizzi di byte pari e al secondo chip tutti gli indirizzi di byte dispari. Se gli accessi sono sequenziali (ad esempio per caricare una linea della memoria cache) *non* avviene un accesso sequenziale allo stesso chip di memoria, ma avvengono accessi alternati fra i due chip, di modo che non sia necessaria l'aggiunta di cicli di wait tra un accesso ed il successivo, cicli che sarebbero stati necessari qualora si fosse dovuto attendere che la memoria tornasse nuovamente disponibile. La soluzione dell'interleaving consente quindi di calibrare il bus di sistema con il tempo  $t_{\text{ras}}$  e non più con il tempo (più lungo) relativo ad un ciclo completo della memoria.

Sebbene si sia preso l'esempio di due banchi di memoria, questo ragionamento può essere generalizzato ad un numero  $N$  di banchi (o chip). Il guadagno prestazionale ottenuto comporta però una maggiore complessità nell'operazione di decodifica degli indirizzi da parte del memory controller.

## Fast operative

La slide 9 tratta le *fast operative*, cioè un sistema con cui è possibile migliorare le prestazioni di una memoria quando l'operazione di lettura o scrittura coinvolge alcune celle adiacenti (in generale tutte le celle associate ad una riga). Queste condizioni sono spesso vere quando l'accesso alla memoria serve ad aggiornare una linea della memoria cache del processore. Le fast operative sono miglioramenti dovuti a soluzioni implementative nei chip di memoria: quando viene abilitata una riga della memoria, tutti i condensatori ad essa associati vengono abilitati. Tale meccanismo dipende fortemente dall'architettura interna del chip e non dal modo in cui viene utilizzato (come invece accade con l'interleaving). Le fast operative prevedono che la circuiteria di selezione delle righe rimanga fissa su una riga mentre il multiplexer delle colonne commuti rapidamente di modo da leggere tutte le celle della riga. In altre parole il RAS resta fisso ed il CAS varia sulle celle della linea. Il primo accesso richiede il tempo  $t_{\text{ras}}$  ma i successivi accessi richiedono decisamente meno tempo poiché non è più necessario specificare la riga su cui si vuole operare.

Un problema di questo meccanismo è che si ha un miglioramento nelle prestazioni solo quando è necessario leggere tutti i dati presenti su una riga. Tuttavia è possibile verificare con analisi statistiche che gli accessi alla memoria sono eseguiti prevalentemente per aggiornare la cache del processore (e questo comporta accesso a celle consecutive di memoria), quindi il sistema delle fast operative offre generalmente miglioramenti prestazionali.



Prof. Marco Mezzalama 08/09

La slide 10 mostra alcune implementazioni delle fast operative a livello dei segnali che interessano il bus della memoria. Il primo caso (caso b) mostra come il segnale RAS resti fisso (abbassato) durante tutta la comunicazione ed il CAS commuti rapidamente per selezionare tutte le celle della riga di memoria. Un possibile miglioramento di questa tecnica (caso c) consiste nell'utilizzare sia i fronti di salita che i fronti di discesa del segnale CAS. Infine è possibile utilizzare direttamente un clock interno alla memoria (caso d) al posto del segnale CAS.

## La famiglia delle DRAM (slide 11-17)

Vi sono stati due salti generazionali nell'evoluzione delle memorie RAM: prima vi è stato quello relativo ai meccanismi di interleaving e fast operative (relativi alle memorie EDO e BEDO RAM) quindi il salto di generazione successivo si è avuto con l'introduzione delle memorie SDRAM.

La slide 12 illustra i pregi ed il funzionamento delle memorie SDRAM (synchronous DRAM). Questa tipologia di memorie migliora le prestazioni sincronizzando il proprio funzionamento con quello della CPU e del bus di memoria. Se la CPU e la memoria sono sincronizzate, infatti, è rimossa la possibilità che le richieste generate dalla CPU arrivino non sincronizzate con il clock della RAM e debbano attendere. In precedenza le memorie RAM avevano un clock interno asincrono rispetto a quello del bus di sistema, il che portava a problemi di sincronizzazione di fase dei due clock (anche nel caso in cui entrambi avessero pari frequenze, il problema persisteva in quanto erano generati da due unità differenti, in modo del tutto indipendente).

La notazione x/y/z/w indica i periodi di clock (del bus) utilizzati in un ciclo burst di memoria, il primo numero è più grande e gli altri a seguire sono più piccoli. Il ciclo burst è caratterizzato dall'avere 4 cicli di dato in uscita (ogni / indica un ciclo di bus). Conoscendo il parallelismo del bus si può conoscere il throughput della memoria. Le memorie di tipo SDRAM permettono di ottenere dei cicli burst 5/1/1/1, dove il primo accesso richiede 5 cicli di bus, ma i successivi 3 accessi

richiedono un solo ciclo. Un caso ideale è quello in cui non si hanno attese per l'accesso al dato richiesto, e quindi il primo tempo è solo 2 (un ciclo per il trasferimento del RAS e uno per il trasferimento del CAS), in generale però l'accesso richiede 3 cicli di wait, da cui i 5 cicli di bus.

Con le memorie DDR e DDR2 (double data rate) per aumentare la quantità di dati trasferiti, non potendo alzare sostanzialmente il clock (a causa dei problemi che si riscontrano quando si aumenta troppo tale valore), si utilizzano dei meccanismi interni di prefetching e si avvalgono sia della fase ascendente sia di quella discendente del segnale di clock per trasferire dati.

## ***Il controller DRAM (slide 19-21)***

Il controller della memoria svolge fondamentalmente tre funzioni: si occupa dell'interfacciamento con l'address bus, provvede alla generazione dei cicli di refresh e si occupa della decodifica degli indirizzi di memoria (generazione dei segnali di abilitazione del banco). Il controller si occupa inoltre della sincronizzazione e della temporizzazione dei vari banchi di memoria tra di loro e con il bus di sistema, in generale questo comporta la presenza di un clock all'interno del controller.

Il DRAM controller può essere suddiviso in tre parti, in base alla funzioni svolte:

- refresh;
- multiplexer dal bus;
- timing e segnali di controllo (per i chip ed il bus), determinati dal timing del bus e dai segnali di inizio ciclo (ADDRESS STROBE).

In un microprocessore di tipo Pentium, essendo il parallelismo del data bus pari a 32 bit, si ottiene sempre, sul bus, tutta la riga di memoria (costituita dai 4 byte corrispondenti ai 32 bit di parallelismo), indipendentemente dal numero di byte effettivamente richiesti.

L'indirizzo specifica unicamente la riga che deve essere abilitata o, in altre parole, l'indirizzo del primo byte della riga, e *non* specifica quali colonne (sulla riga) abilitare. Per specificare un byte in particolare sono presenti dei segnali di *byte enable* che permettono di abilitare i byte effettivamente richiesti, sia per le operazioni di lettura che per quelle di scrittura.

Nel caso in cui l'indirizzo corrisponda ad una cella presenta a metà di una riga (questo risulta essere assolutamente lecito), i segnali di byte enable dovranno essere gestiti correttamente in modo da prendere i veri dati di interesse; da notare che nel caso in cui l'indirizzo di partenza si trovi a metà di una riga, saranno necessari due cicli di memoria prima di avere i dati stabili sul data bus.

In generale i compilatori allineano le variabili ed i dati ad indirizzi di memoria tali da offrire le massime prestazioni, quindi multipli del parallelismo del bus della memoria. Sono spesso inoltre presenti direttive o parole chiave per richiedere al compilatore di allineare una variabile ad un preciso set di indirizzi (ad esempio si può fare in modo che una variabile sia allineata ad indirizzi multipli di 8 byte o 16 byte).

# Schema di memoria dram

**ABUS:**

**ABUS0-1:** indirizzo di partenza nella parola (da 4 byte)

**ABUS2-27:** indirizzo parola nel banco, con profondità 64M

**ABUS28-29:** indirizzo del banco



*Prof. Marco Mezzalama 08/09*

La slide 21 mostra come vengono utilizzati i 32 bit di indirizzo quando è presente una memoria da 1GB. Dei 32 bit ne vengono utilizzati solamente 30 poiché sono sufficienti ad indirizzare tutta la memoria disponibile, i bit sono utilizzati nel seguente modo:

- 2 bit (blu): abilitazione del/i byte all'interno della parola (da 4 byte);
- 26 bit (verdi): la parola di interesse all'interno del banco di memoria;
- 2 bit (gialli): il banco di memoria.