RITARDATORE

INFORMAZIONI DI CARATTERE GENERALE

Questa sezione di progetto si propone di realizzare un modulo per la gestione della memoria *SDRAM* presente sulla scheda *XSA-50*. Questo lavoro si appoggia in parte a quello svolto dalla *Xess Corporation* (http://www.xess.com) la quale implementa un modulo per far sembrare la *SDRAM* una semplice *RAM statica*.

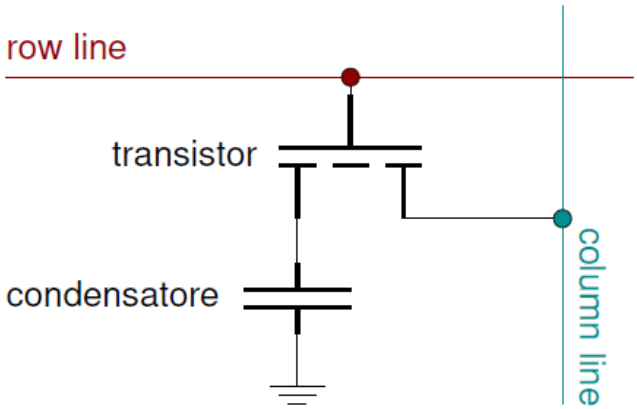
La scheda *XSA-50* è equipaggiata di una memoria *Hynix HY57V641620HG*, ovvero una 67'108'864-bit *CMOS Synchronous DRAM* organizzata in 4 *banchi* di memoria da 1M × 16 bit (in totale 8 Mbyte).

CONCETTI BASE DELLE MEMORIE SDRAM

IL TERMINE

La *SDRAM* (*Syncronous Dynamic RAM*) è una *memoria* ad *accesso casuale dinamica* e *sincrona*: *dinamica* in quanto, per come è costruita, ha bisogno di essere costantemente "rinfrescata", *sincrona* perché utilizza un segnale di *clock esterno* per la *sincronizzazione* delle operazioni di *I/O*; questo permette un incremento delle prestazioni e una maggiore efficienza. La natura sincrona del *chip* fa sì che si possano implementare complessi modi operativi, *pipeline* interne e *trasferimenti di dati a blocchi* (*burst*).

LE CELLE DI MEMORIA



Come mostrato in figura, l'*informazione binaria* è immagazzinata in unità che consistono in un *transistor* e in un piccolissimo *condensatore* del valore di circa 20÷40 fF (Femtofarad, 0.020-0.040 pF) per ogni cella. Un condensatore carico ha valore logico 1, uno scarico ha valore logico 0.

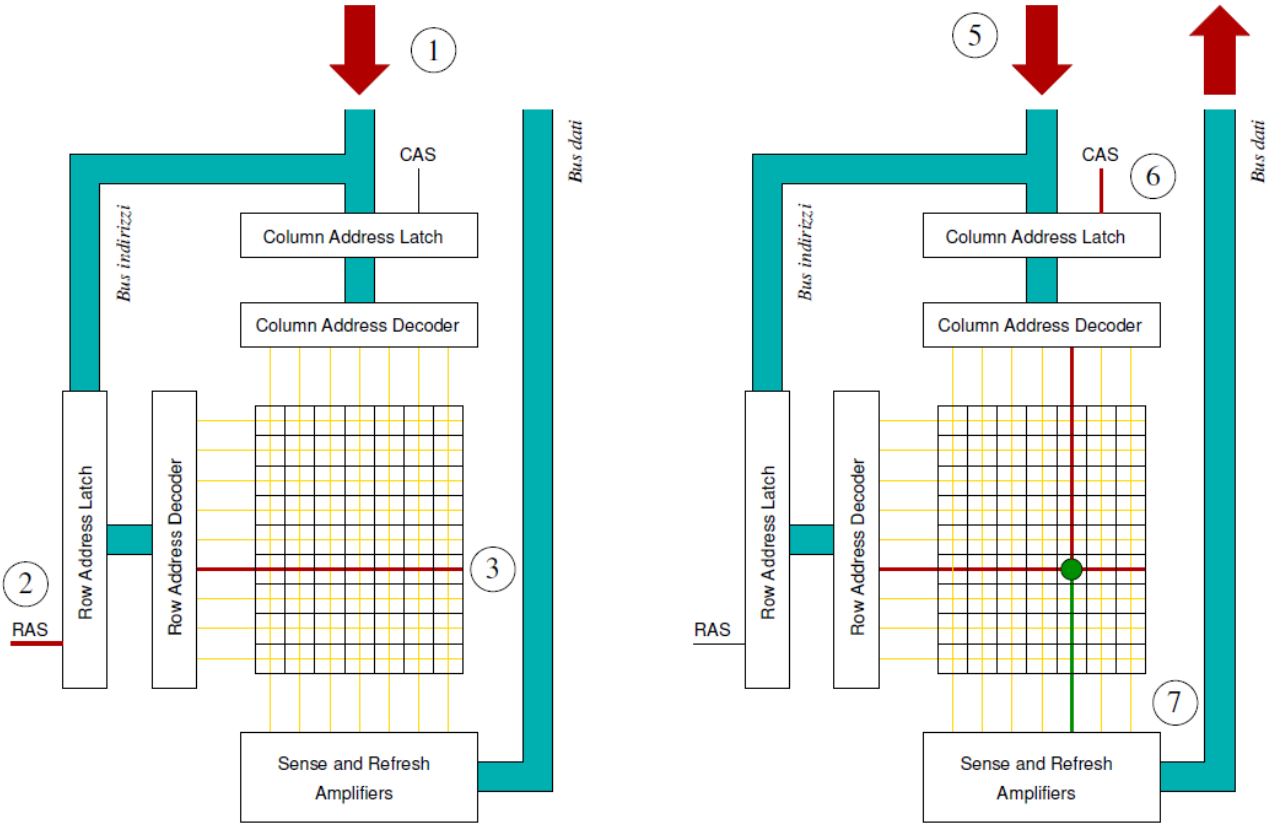
Sfruttando l'*effetto capacitivo* dei *CMOS* si riesce a memorizzare un bit di informazione utilizzando pochissimi transistors. Il prezzo da pagare, per questo risparmio in complessità realizzativa, è che il valore inserito in una cella di memoria resta "memorizzato" solo per una quantità di tempo relativamente breve (frazioni di secondo), dopo di che viene "dimenticato" irrimediabilmente. L'idea che consente di realizzare dei *moduli RAM dinamici* consiste nel "rinfrescare la memoria" al dispositivo un attimo prima che il valore memorizzato vada perso. L'operazione di *refresh* di una memoria dinamica avviene leggendo il contenuto e riscrivendo immediatamente lo stesso valore appena letto, in modo che questo possa essere mantenuto per un'ulteriore frazione di secondo. L'operazione di *refresh* deve essere

ovviamente ripetuta periodicamente per tutti i bit memorizzati in un *modulo SDRAM*, ed il ciclo di refresh di tutte le celle deve completarsi in un tempo inferiore rispetto a quello necessario ad una cella per svuotarsi.

L'ACCESSO IN MEMORIA

Essendo la *SDRAM* una memoria di grande capacità, i progettisti, per risparmiare linee, hanno deciso di spezzare la trasmissione dell'indirizzo in due parti, dette *Column Address* e *Row Address*. Queste rappresentano la riga e la colonna dove si trova il bit nelle matrici interne di memoria. Tale soluzione comporta vantaggi e svantaggi: se da un parte ha l'enorme pregio di far risparmiare molto spazio, dall'altra rende più complicato l'*interfacciamento* con questo tipo di memorie.

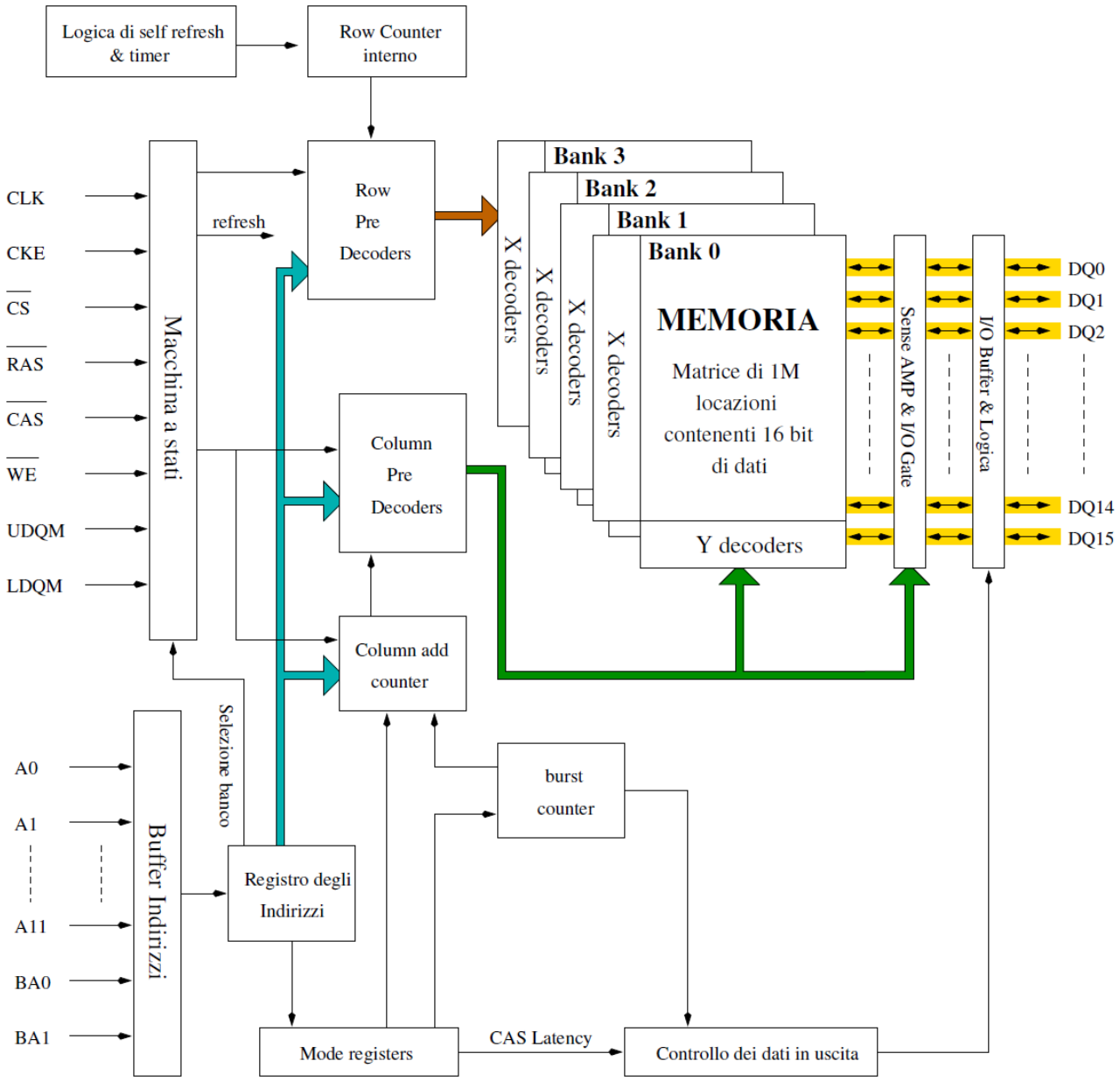
Per accedere ad una locazione di memoria è quindi necessario prima specificare il *Row Address* (che viene memorizzato dal *chip* in un *buffer interno*) a cui segue la trasmissione del *Column Address* usando le stesse linee. Se gli accessi successivi sono nella stessa "pagina" (ossia hanno lo stesso *Row Address*) non è necessario specificare di nuovo il *Row Addrerss* ma solo il *Column Address*. Vediamo brevemente lo schema di accesso alla memoria:



- si immette sul *bus indirizzi* il *Row Address*;
- si immette sul *bus indirizzi* il *Row Address*;
- il valore memorizzato nel *Latch* viene *decodificato* ed identifica una specifica riga (*Row*) nella matrice di memoria;
- i *segnali* ed il *bus* vendono disinseriti;
- si immette sul *bus indirizzi* il *Column Address*;
- si attiva il segnale *CAS* (*Column Address Strobe*) che agendo su un apposito *Latch* memorizza il *Column Address*;
- il valore immagazzinato permette di individuare la colonna in cui si trova il *dato*. L'incrocio tra colonna e riga individua univocamente la locazione di memoria e in caso di lettura il suo contenuto viene inviato sul *bus dati*, in caso di scrittura il contenuto del *bus dati* viene scritto nella locazione.

STRUTTURA E INTERFACCIAMENTO CON L'ESTERNO

In figura possiamo vedere il diagramma funzionale a blocchi della nostra *memoria* (*Hynix HY57V641620HG*). In particolare sono immediatamente visibili i quattro *banchi* da 1M x 16bit, i *bus* degli *indirizzi* (in azzurro il *bus condiviso*, in verde l'*indirizzamento* delle *righe* e in arancio quello delle *colonne*) e il *bus dati* (in giallo).



Per quanto riguarda invece l'interfacciamento con l'esterno, *DQ0-DQ15* è il *bus bidirezionale* che trasporta i dati da e verso la memoria, *A0-A11* è il *bus* degli *indirizzi*, mentre *BA0* e *BA1* servono a *selezionare* il *banco*. Gli altri otto ingressi sono rispettivamente:

- *CLK*: il *master clock* della memoria (SDRAM, memoria sincrona);
- *CKE*: il *clock enable*, serve per abilitare il clock;
- *CS*: *chip select*, attiva o disattiva il chip (queste memorie spesso sono organizzate in banchi – si pensi alla classica RAM dei PC – per cui CS serve a indicare per quale chip sono destinati i comandi che viaggiano sul bus condiviso);
- *RAS*: *Row Address Strobe*, prende i dati presenti nel registro degli indirizzi e li fa interpretare al chip come indirizzi di riga;

- *CAS*: *Column Address Strobe*, prende i dati presenti nel registro degli indirizzi e li fa interpretare al chip come indirizzi di colonna;
- *WE*: *Write Enable*, indica se l'operazione da fare in memoria è di lettura o scrittura;
- *LDQM*/*UDQM*: *Data Input*/*Output Mask*: i *DQM* controllano i buffer tristate presenti sui pin dei dati;
- *LDQM* controlla le linee dati dal *DQ*0 al *DQ*7, viceversa *UDQM* controlla le linee dati da *DQ*8 a *DQ*15: i due byte vengono gestiti in modo indipendente.

Per quanto concerne gli indirizzi, 1M locazioni a banco significa uno spazio di indirizzamento di 20 bit ($2^{20} \simeq$ 1M): avendo un bus indirizzi di 12 bit (*A*0-*A*11) ogni locazione è identificata usando 12 bit per la riga e 8 bit per la colonna; a questo si aggiungono 2 bit per identificare il banco.