Architettura degli elaboratori - modulo A
Anno Accademico 2000 / 2001

Capitolo 6 - Memorie

Una ***cella di memoria*** in un sistema digitale è un "***qualcosa***" in grado di ***memorizzare*** il valore booleano che una variabile ha assunto in un determinato istante.
La caratteristica fondamentale di un ***elemento di memoria*** è dunque quella di conservare il valore booleano che la variabile d'ingresso ha assunto al tempo t_0, anche se essa successivamente si è portata al valore logico opposto.
Quello che avviene, quindi, è che, se la variabile I al tempo di scrittura t_0 era, ad esempio, al valore booleano "1", al tempo di lettura t_1, successivo, la variabile di uscita dalla cella di memoria che chiameremo "O" è ancora "1", anche se nel frattempo I ha assunto il valore "0".

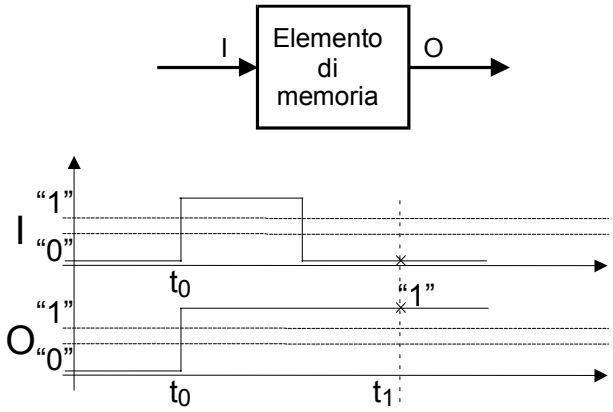

Fig. 1

Esistono elementi di memoria basati su principi di funzionamento molto diversi tra loro ma una delle possibilità più usate nei sistemi digitali, e già citata in precedenza, è quella di usare la struttura logica nota come "**flip-flop**".
Abbiamo visto, però, che il capostipite della famiglia, quello che costituisce la più semplice rete sequenziale, il flip-flop Set/Reset, ha due ingressi separati S ed R, il primo per portare l'uscita Q al valore logico "1" ed il secondo per riportarla a "0", mentre l'elemento di memoria deve avere un unico ingresso I (Input), che deve poter portare l'uscita O (Output) al valore booleano di I, sia che esso sia "0", sia che sia "1".
Non è difficile, però, adattare il flip-flop R/S alla funzione richiesta, basta collegare l'ingresso I direttamente all'ingresso S del flip-flop S/R ed, attraverso un invertitore, all'ingresso R, esattamente come nel flip-flop di tipo D.

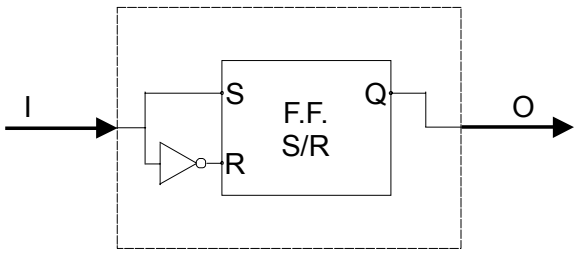

Fig. 2

Possiamo dire di avere ottenuto così un elemento di memoria?. Certamente ancora no! Perché nel circuito di figura l'uscita diretta del flip-flop Set/Reset, che di solito si chiama Q, ma in questo caso chiamiamo O (da Output per ricordarci che è l'uscita dell'elemento di memoria) segue nel tempo, più o meno istantaneamente, l'andamento di I. Se I, infatti, da "1" si porta a "0", all'ingresso R sarà subito applicato un "1" logico e ciò porterà immediatamente O al livello basso.
Affinché il circuito possa assumere la funzione di ***elemento di memoria*** è necessario introdurre la possibilità di abilitare gli ingressi solo nell'istante in cui si vuole "scrivere" un valore booleano nella cella elementare di memoria.
Un elemento di memoria in grado di funzionare secondo le specifiche assegnate potrebbe avere, in linea di principio, la struttura logica di figura 3.

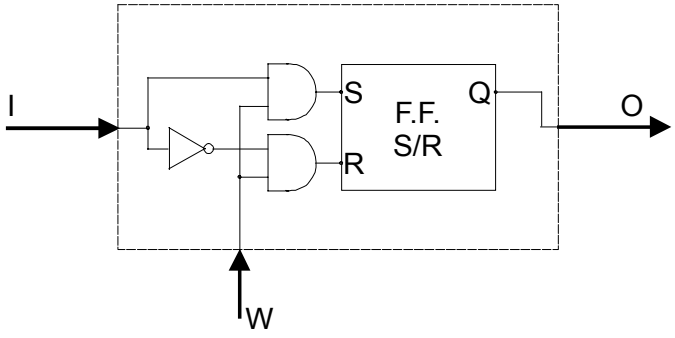

Fig. 3

L'ingresso di controllo svolge la funzione di abilitazione alla scrittura e viene in gergo detto ingresso di "Write".
Ci si potrebbe chiedere, a questo punto, che differenza c'è tra il circuito di figura 3 ed il flip-flop di tipo D con abilitazione, visto in precedenza. Non c'è nessuna differenza di principio ma sarebbe impreciso dire che un sistema di memoria è fatto da flip flop di tipo D perché, come vedremo, l'elemento di memoria (S/R) si comporta da flip-flop di tipo D utilizzando elementi logici aggiuntivi che sono presenti solo sui pin d'ingresso e servono, quindi, tutte le celle di un sistema. **Utilizzare come elementi di memoria flip-flop di tipo D, significherebbe moltiplicare inutilmente le funzioni logiche aggiuntive per il numero di celle del sistema, a discapito della compattezza del circuito integrato**.

Il circuito di figura 3 possiede tutte le funzionalità indispensabili per memorizzare lo stato logico di una variabile isolata (memoria da 1 bit) ma di solito in un sistema digitale si ha necessità di memorizzare centinaia o migliaia di "parole", ciascuna formata da insiemi ordinati di molti bit.
Quello che chiameremo d'ora in poi un ***sistema di memoria statica, a lettura e scrittura ed ad accesso casuale*** (Static Random Access Memory, **SRAM**) è una struttura che ha molte affinità di principio con gli scaffali portalibri di una biblioteca, o meglio, con gli scaffali portacassette VHS di una videoteca, infatti una memoria digitale, così come una videoteca VHS ha la caratteristica fondamentale di essere composta da contenitori di uguale volume, adatti a contenere lo stesso identico numero di elementi, di ingombro fisso; nel caso di una memoria, gli elementi sono i bit.
Ogni palchetto di questa ideale scaffalatura è una "***locazione***" o "***posizione***" di memoria è può contenere un numero fissato di bit che può andare da 1 ad 8 byte o anche di più.
Ogni "*locazione*", per poter essere individuata durante una operazione di scrittura o lettura, deve essere associata, in maniera univoca, ad un codice (ovviamente binario) che si chiama "***indirizzo***".

Quando si opera su una *locazione* che corrisponde ad un determinato *indirizzo*, tutti e soltanto i bit di quella locazione sono coinvolti nell'operazione.

In parole povere, la ***lettura*** di una certa *locazione* in una memoria organizzata a "*parole*" (per esempio di 32 bit) **fornisce in uscita tutti e soltanto i 32 bit di quella *locazione* nell'ordine in cui erano memorizzati**. Se si effettua, invece, una ***scrittura*** in memoria, **i 32 bit forniti al sistema, in ingresso, vengono ordinatamente memorizzati nella *locazione* corrispondente all'indirizzo selezionato**.
L'operazione non modifica il contenuto di nessun altra locazione. **Se la locazione selezionata conteneva precedentemente altre informazioni binarie, il contenuto precedente si perde**, in quanto la nuova sequenza ordinata di bit (si chiama anche "***stringa***") viene scritta sopra alla vecchia. Si dice in gergo che la nuova parola è stata "***soprascritta***" alla vecchia.

Dalle considerazioni generali che precedono si deduce, evidentemente, che un sistema deve contenere tante celle elementari di memoria quanti sono i bit totali da conservare. Il numero totale di bit memorizzabile, dato dal numero di locazioni moltiplicato per il numero di bit della parola, costituisce la ***capacità della memoria***.
La memoria deve, però, anche contenere **una struttura logica in grado di individuare, di volta in volta, la locazione nella quale depositare l'informazione o dalla quale essa va prelevata**.
La struttura svolge insieme la funzione delle etichette sugli scaffali della videoteca e dell'addetto alla sistemazione ed al prelievo delle cassette, con la particolarità che la collocazione (o il prelievo) interessa sempre l'intero palchetto.
La locazione, come abbiamo già detto è univocamente associata ad un codice binario detto ***indirizzo***. Quello che serve è, allora, una logica di decodifica dell'indirizzo che porti a selezionare la locazione corrispondente. Proprio per questo la struttura di individuazione della *locazione* su cui operare viene anche detta ***logica di selezione***.

Per quanto riguarda il legame matematico che deve correre tra il numero degli elementi da individuare ed il numero di bit del corrispondente codice, non c'è niente da aggiungere a quanto a suo tempo detto per le decodifiche. **Il numero di bit d'indirizzo è pari al logaritmo in base due del numero di locazioni che si vogliono nel sistema di memoria, arrotondato per eccesso al numero intero successivo**. La conseguenza ovvia di ciò è che il numero di locazioni di un sistema di memoria è sempre uguale ad una potenza del due.
Una memoria da 4096 locazioni, ad esempio, richiede un indirizzo di 12 bit.

Con l'elemento di memoria finora messo insieme aggiungendo qualche porta logica al flip-flop S/R, è possibile costruire un sistema di memoria?
La risposta, anche questa volta è negativa, perché nel circuito di figura 3 **manca una variabile booleana in grado di attivare o disattivare l'intero dispositivo a comando**.
Questo è un problema centrale perché il modo di operare del sistema di selezione non può che essere quello di **attivare la sola locazione individuata dall'indirizzo, mentre tutte le altre sono disabilitate**.
La necessità di questo modo di operare di una memoria ad accesso casuale deriva dal fatto che essa deve avere un unico canale per i bit che costituiscono l'informazione in ingresso, anzi, benché la cosa non sia indispensabile, di solito, ha **un unico registro, a tanti bit quanti sono quelli delle locazioni. In tale registro si trova, prima di ogni operazione di scrittura, la parola da memorizzare, oppure, a fine operazione di lettura, quella che era registrata nella locazione letta**.
Questo canale di ingresso/uscita viene di solito detto **Imput/Output Register** o **Imput/Output Buffer**.
Quest'ultima dizione si usa quando si vuole sottolineare che l'informazione è in transito verso una collocazione più stabile.
L'uscita di ciascuno dei bit del registro d'ingresso deve, quindi, essere collegata agli ingressi dei bit omologhi di tutte le locazioni della memoria. Cioè, tutti i bit in posizione 0 di tutte le locazioni sono

collegati al bit 0 del registro, indipendentemente dal fatto che esso sia il più significativo o il meno significativo. I bit in posizione 1 corrispondono al bit 1 del registro, e così via fino all'ultimo bit, sia esso il 15, il 31 o il 63, a seconda della lunghezza della "*parola*".
Naturalmente questa struttura comporta che se non si vuole registrare la stessa stringa in tutte le locazioni, solo la locazione che corrisponde all'indirizzo in ingresso alla logica di selezione deve essere attivata, ma non solo, deve anche avere l'ingresso delle sue celle elementari abilitate in scrittura.
Allo stesso modo, le uscite delle celle in posizione corrispondente in tutte le locazioni debbono essere *logicamente connesse* agli ingressi omologhi del registro di uscita.
Attenzione alla espressione "***logicamente connesse***": essa sottolinea che le uscite dei flip-flop delle locazioni di memoria non possono essere, come gli ingressi, ***elettricamente connessi*** ma debbono confluire al corrispondente ingresso del registro attraverso un OR logico o qualcosa che svolga una funzione equivalente.

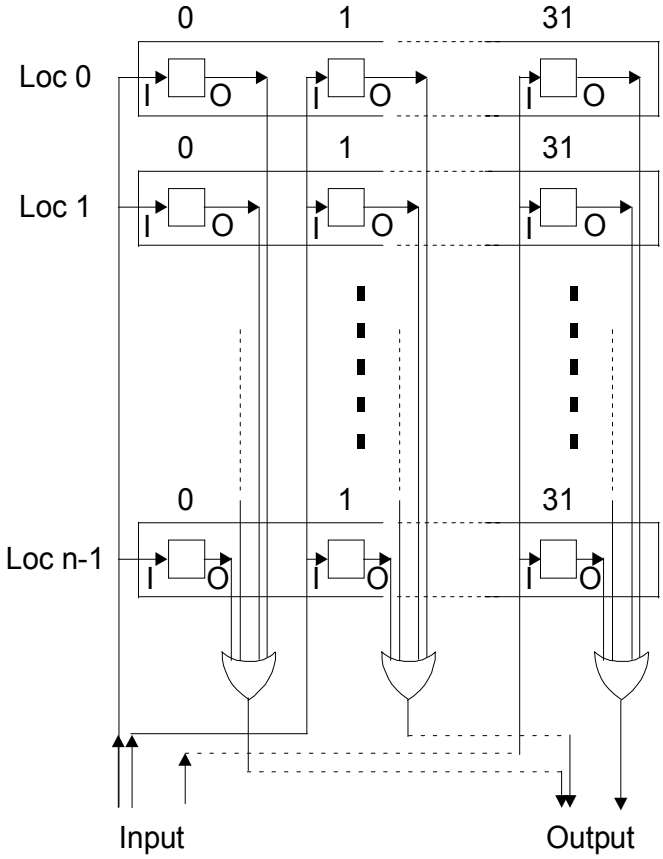

Fig. 4

Il tipo di connessione che ne deriva è schematizzato in figura 4.
E' evidente che anche per l'uscita dei dati è necessario che una sola locazione sia attivata ed abilitata in lettura altrimenti ciascun bit del registro d'uscita riceverebbe l'OR dei bit omologhi di tutte le locazioni e questa informazione è chiaramente priva di senso.

Non è difficile costruire intorno ad un flip-flop una struttura logica in grado di bloccarne completamente il funzionamento, basta bloccarne contemporaneamente gli ingressi e le uscite, come in figura 5, con una variabile logica S, che sta per "select".

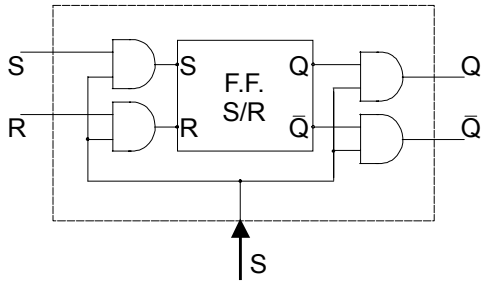

Fig. 5

Se si immagina di riempire con elementi logici come quelli di figura 5 le celle della struttura di figura 4, di aggiungere una matrice di selezione (decodifica d'indirizzo) che gestisce le linee di selezione delle "*locazioni*", ottenute collegando insieme le linee S di tutti i bit della posizione e si fanno, poi, piccoli adattamenti, si ottiene la struttura base di un sistema di memoria.

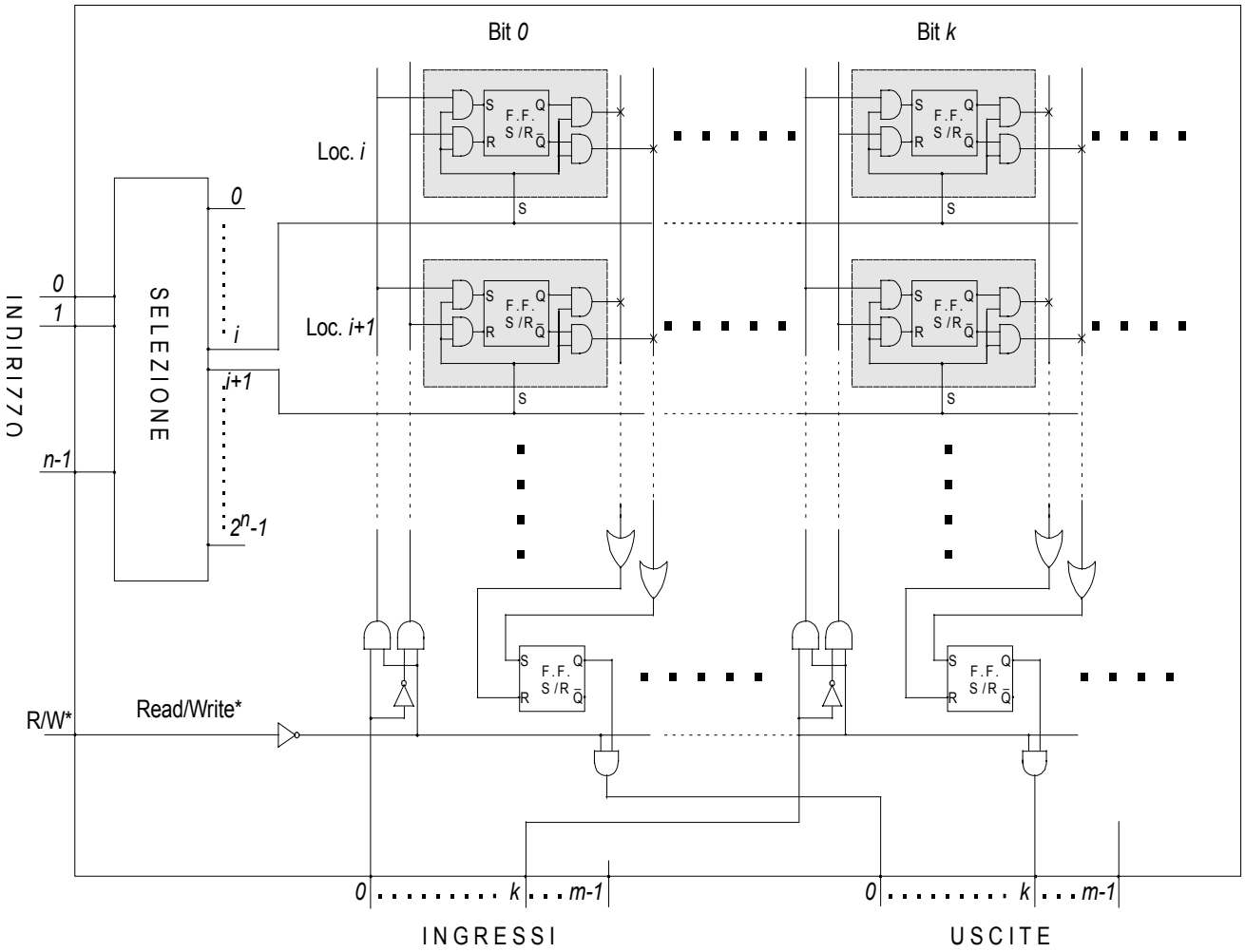

Fig. 6

La struttura logica di figura 6 contiene quasi tutti gli elementi costitutivi di in chip di memoria. Mancano soli pochi dettagli.
Il pin di controllo dell'operazione funziona in logica combinata, nel senso che il livello "alto" abilita la lettura e quello "basso" la scrittura.

E' importante notare che l'uscita del sistema di memoria di figura 6 è in "***logica invertita***". Come dire che se un bit di una certa locazione è ad "1" logico,. dopo la lettura c'è in quella posizione della parola d'uscita, un livello "basso" (0 Volt). Questo è una tradizione nei chip di memoria statica che fu introdotta quando, come vedremo, il circuito d'uscita, per consentire di combinare più chip per formare una memoria di maggiore capacità, era del tipo "***open collector***".
Ora che, come vedremo, si usa per le uscite dei dati il cosiddetto "***driver tristate***", la logica invertita non è più indispensabile ma la tradizione della "***logica negata***" continua.
E' ovvio che, se il sistema digitale che utilizza la memoria opera in "***logica diretta***" ("1" corrisponde al livello di tensione "alto" e "0" a quello "basso") si deve prevedere una inversione di logica nel collegamento dati.
Lo schema a blocchi di un chip di memoria come quello di figura 6 si presenterà come in figura 7.

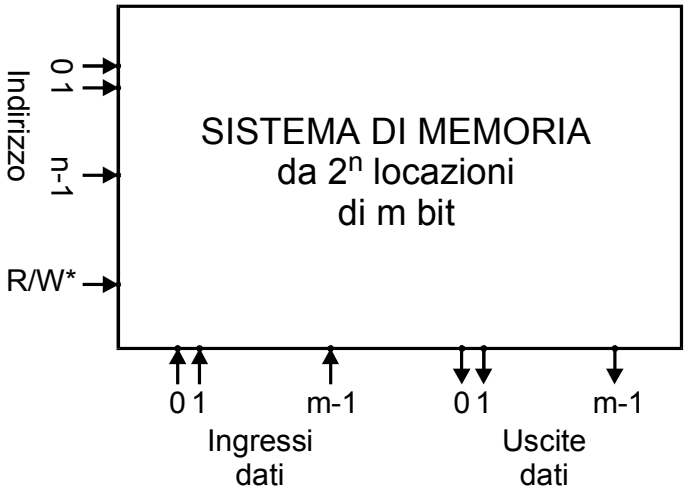

Fig. 7

Se, per esempio, in una certa memoria n = 12, il numero di locazioni presenti è 4096 e se m = 8 il numero totale di bit contenuti nel chip (la capacità) è 32.768 che per brevità si arrotonda, per difetto, al migliaio di bit e si dice che il chip ha, una ***capacità*** di 32 K.

	Indirizzo (12 bit)	Contenuto locazioni (8 bit)
(0)	000000000000	00110101
(1)	000000000001	01001011
(2)	000000000010	00010001
(3)	000000000011	00100100
	⋮	⋮
(256)	000100000000	00000000
	⋮	⋮
(4094)	111111111110	10000010
(4095)	111111111111	00000000

Fig. 8

Dal punto di vista dell'utilizzatore il chip si presenta come una scatola nera alla quale si invia un indirizzo ed un bit di funzione. Se il bit è "1" la scatola presenta in uscita il contenuto della locazione interessata, se invece esso è "0" i valori booleani presenti ad un certo momento sulle linee di ingresso dati vengono memorizzati in quella locazione.

Se si potesse, ipoteticamente, vedere contemporaneamente il contenuto di tutti i bit di un chip di memoria da 4096 locazioni di 8 bit, si vedrebbe la tabella della colonna di destra di figura 8.
Nella realtà il contenuto della tabella può essere estratto dalla memoria solo ad una parola per volta, in 4096 cicli di lettura.
Per farlo davvero, si può pensare di collegare alle linee di indirizzo le uscite di un registro a 12 bit, il cui contenuto venga incrementato di una unità dopo ogni lettura.

La struttura logica di figura 6 può rappresentare sia una parte della struttura interna di un chip di memoria statica integrato, sia una parte di un sistema di memoria realizzato con funzioni logiche "***discrete***". In entrambi i casi essa va completata con un ***registro di ingresso dati*** e con un ***registro di uscita dati*** o, come già detto, con un unico ***registro di ingresso/uscita***. In ogni caso il numero dei bit dei registri o del registro deve essere pari al numero *m* dei bit delle locazioni.
Anche facendo riferimento al caso più semplice di registri d'ingresso e d'uscita separati, per rendersi conto del funzionamento di un sistema di memoria bisogna mettere a fuoco la dinamica dei segnali elettrici all'interno del sistema.
Praticamente tutti i sistemi di memoria sono "sincroni", nel senso che i loro cicli di "***lettura***" e "***scrittura***" sono guidati da un "***segnale di cadenza***" o "***clock***" a frequenza fissa e la durata dei cicli è espressa in numero di periodi del clock. I cicli del diagramma temporale di figura 9 durano 4 periodi di clock.

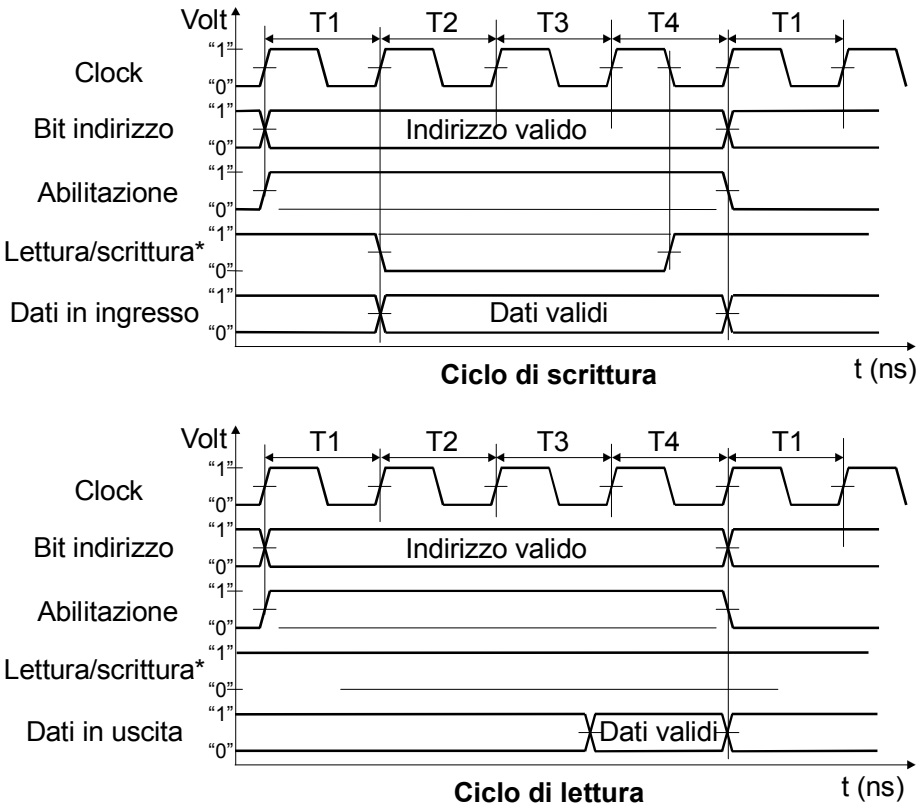

Fig. 9

Da notare che nel ciclo di scrittura il sistema che utilizza la memoria deve rendere disponibili i dati da memorizzare non oltre l'inizio di T2 ma, ovviamente, i dati saranno effettivamente memorizzati nella locazione prescelta quando il segnale di abilitazione alla scrittura si sarà propagato attraverso il sistema di memoria. Per evitare il rischio che possano essere scritte nella locazione di memoria indirizzata dei dati errati, l'abilitazione alla scrittura viene tolta con un mezzo periodo di clock di anticipo rispetto al limite di disponibilità dei dati.
Nel ciclo di lettura, la effettiva disponibilità dei dati per il sistema che ha chiesto la lettura in memoria arriva con un ritardo asincrono rispetto al clock perché esso dipende dai tempi di propagazione dei bit attraverso le linee ed i circuiti d'uscita della memoria. In ogni caso l'abilitazione al registro del sistema che utilizza i dati deve essere generato con un congruo anticipo rispetto all'istante in cui essi perdono validità (fine di T4).

Sistemi di memoria di elevata capacità (multichip)

Quando bisogna costruire un sistema di memoria di elevata capacità, avendo a disposizione dei chip di memoria integrati di capacità più piccola, bisogna considerare tre casi:

1. Il chip ha un numero di locazioni pari a quello richiesto, ma ha un numero di bit per locazione insufficiente per la lunghezza della "*parola*" del sistema utilizzatore, sia esso un computer o qualsiasi altro tipo di sistema digitale. Potrebbe accadere, ad esempio, che la memoria che si deve costruire debba avere 16 K locazioni da 16 bit, mentre il chip, pur avendo 16 K locazioni (14 bit d'indirizzo), ha solo 4 bit per locazione.
2. Il chip di memoria ha capacità adeguata per quanto riguarda il numero di bit per locazione ma ha un numero di locazioni insufficiente. Per esempio, per costruire la stessa memoria del caso precedente, supponiamo che il chip disponibile contenga 2 K locazioni da 16 bit
3. il chip ha un numero di locazione insufficiente alle esigenze ed un numero di bit per posizione che è solo una frazione della lunghezza della parola richiesta. Per esempio il chip ha 4 K locazioni da 4 bit. Inutile dire che questo è di gran lunga il caso più frequente.

Espansione del numero di bit disponibile per ciascuna locazione.

Il caso 1 è il più semplice, basta utilizzare "***in parallelo***" tanti chip di memoria quanti sono necessari per raggiungere la lunghezza della parola richiesta.

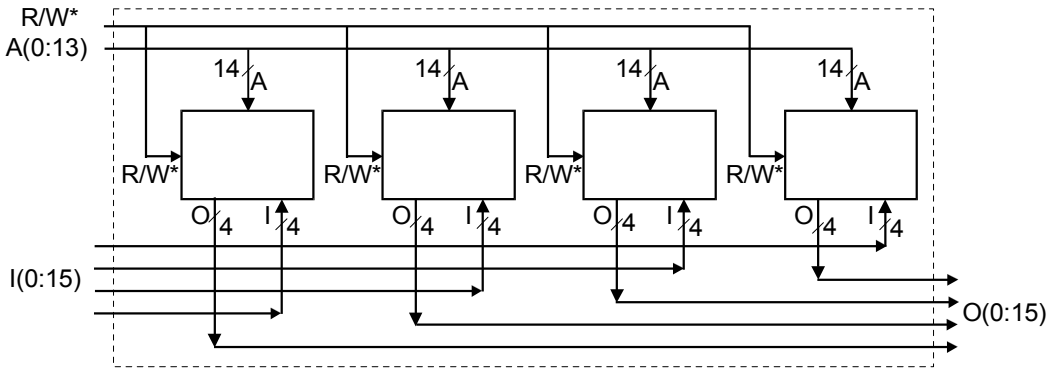

Fig. 10

L'utilizzazione dei chip "***in parallelo***" si ha quando si collegano direttamente tra loro i corrispondenti bit d'indirizzo dei chip. Per esempio nel caso ipotizzato per raggiungere 16 bit di parola sono necessari 4 chip. A ciascuno di essi si mandano nello stesso ordine gli stessi 14 bit

d'indirizzo e lo stesso segnale di lettura/scrittura, mentre gli ingressi e le uscite dei dati vengono affiancati. In pratica tutte e quattro le ***matrici di selezione*** individueranno all'interno dei 4 chip lo stesso indirizzo, ma, mentre il primo contiene i 4 bit più significativi, il secondo i successivi 4 e così via, fino al quarto, che contiene i 4 bit meno significativi.
E' utile osservare che la struttura di figura 10 contiene una apparente incongruenza, in quanto usa quattro diverse matrici di selezione per puntare allo stesso identico indirizzo. Sembra una tecnica assurda, ma, purtroppo non ha alternative, perché **la matrice deve necessariamente essere interna al chip, essa, infatti, riduce il numero di pin del contenitore del chip per l'indirizzamento, in ragione del logaritmo in base 2 del numero di locazioni**. Per esempio per indirizzare 4096 locazioni bastano 12 pin invece di 4096.
E' anche interessante osservare che se si ha un chip da 4096 locazioni da 8 bit e si volesse esaminare la possibilità di raddoppiarne la capacità, portando i bit per locazione da 8 a 16, si avrebbe bisogno di 8 pin in più nel contenitore, se si utilizza la tecnica di sovrapporre ingressi ed uscite, altrimenti, ce ne vogliono addirittura 16, mentre è possibile raddoppiare il numero di locazioni e, quindi, la capacità, con un solo pin d'indirizzo in più. Bisogna passare da 12 a 13.

Espansione del numero di locazioni.

Il caso 2, invece, pone due problemi, uno relativo alla gestione degli indirizzi ed uno relativo alle uscite dati da una struttura logica del tipo di figura 6, che non è stato ancora affrontato.
Facendo riferimento al caso pratico già ipotizzato, la memoria da costruire ha un numero di locazioni che è otto volte quello del chip, che ne contiene 2 K. Per individuare una delle locazioni disponibili nel chip servono 11 bit d'indirizzo perché 2^{10} = 2048, mentre la memoria nel suo complesso deve usare indirizzi da 14 bit.
Se si prendono gli 11 bit meno significativi dell'indirizzo a 14 bit del sistema di memoria e si collegano ordinatamente agli 11 pin indirizzo di tutti gli 8 chip, ignorando i 3 bit più significativi si realizza una corrispondenza che associa, a ciascun configurazione logica di indirizzo, caratterizzata dall'avere uguali gli 11 bit meno significativi (sono 8), un indirizzo in ciascuno degli 8 chip.
Se si ipotizza che i chip di memoria abbiano un pin di abilitazione, che se non è attivato ne blocca completamente il funzionamento, il problema della selezione si può facilmente risolvere inviando i 3 bit più significativi ad una decodifica che per ogni combinazione dei 3 bit attiverà una soltanto delle 8 uscite. A questo punto, ciascuna delle uscite della decodifica può essere collegata al pin di abilitazione di uno degli 8 chip, in modo che a ciascuna delle 8 combinazioni dei 3 bit corrisponda il campo d'indirizzamento (in decimale) della tabella 1.

0 0 0	da	0	a	2.047	$(2^{10}-1)$
0 0 1	da	2.048	a	4.095	$(2^{11}-1)$
0 1 0	da	4.096	a	6.143	
0 1 1	da	6.144	a	8.191	$(2^{12}-1)$
1 0 0	da	8.192	a	10.239	
1 0 1	da	10.240	a	12.287	
1 1 0	da	12.288	a	14.335	
1 1 1	da	14.336	a	16.383	$(2^{13}-1)$

Tab. 1

Lo schema a blocchi del meccanismo di selezione è in figura 11. Si tenga presente che si tratta di uno schema parziale, relativo alla sola utilizzazione dei bit d'indirizzo

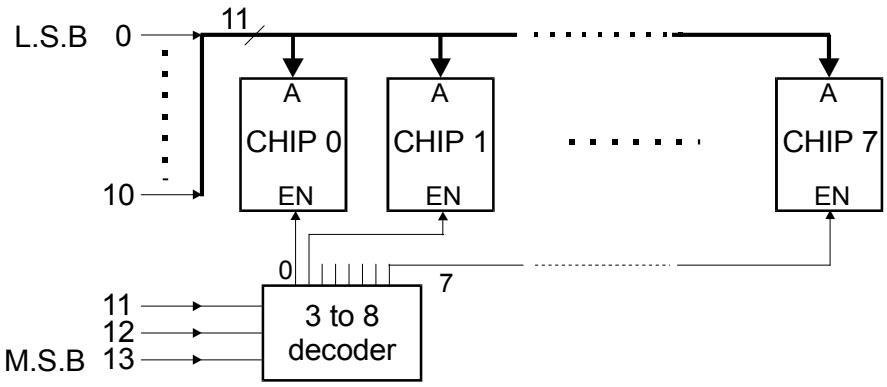

Fig 11

Poiché il sistema di memoria formato da 8 chip a 16 bit deve avere, per definizione, 16 ingressi dati e altrettante uscite, come vanno collegati gli ingressi e le uscite dei singoli chip?
Per gli ingressi non c'è problema, il pin 0 del sistema va collegato a tutti i pin 0 dei chip e così via, ordinatamente, fino ai pin 15.

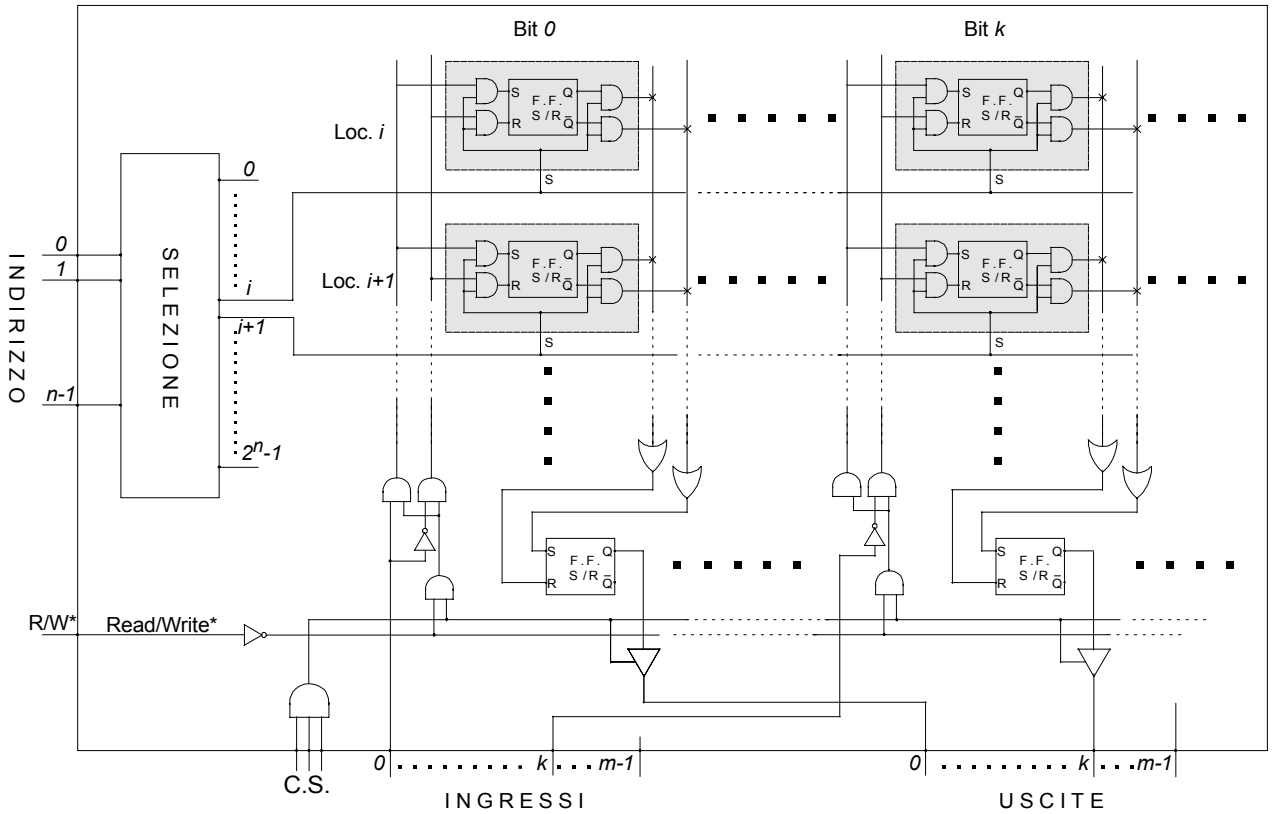

Fig. 12

Per le uscite invece il problema c'è ed è sostanziale, perché le uscite logiche normali impongono sia il livello "basso" che quello "alto" e per poter connettere logicamente le uscite dei chip ci vorrebbe un OR esterno, perché non si può prevedere a priori quanti chip è necessario combinare.
Per evitare ciò, come si è già accennato, i chip di memoria si facevano un tempo con uscite "open collector" ed in logica negata in modo da poter avere una struttura "wired OR". Oggi, invece, si introduce un "driver tristate" su ciascun bit di dati in uscita, collegato al pin di abilitazione, ed il problema è risolto.

Il pin di abilitazione si chiama di solito "***Chip Select***" (**CS**).
Esso spesso viene duplicato o triplicato, nel senso che viene inserito all'ingresso del chip una funzione logica AND a due o tre ingressi in modo che il segnale di CS venga generato solo quando tutti i pin di ingresso dell'AND sono ad "1". Questo semplice artificio logico consente di attuare una strategia di selezione del chip di memoria che si definisce, come vedremo più avanti, "***selezione a coincidenza***".
La logica completa del chip è mostrata in figura 12:

Con la struttura logica di figura 12 le uscite dei bit della parola possono essere messe semplicemente insieme ordinatamente, perché il meccanismo di selezione attraverso il CS garantisce che non ci saranno conflitti di livelli logici in uscita. Ad assumere un livello logico booleano (0 o1) saranno solo le uscite del chip che contiene l'indirizzo selezionato. Tutti gli altri saranno nello ***stato di alta impedenza*** (il cosiddetto ***terzo stato***).

Quando i driver ***tristate*** non esistevano, il problema veniva risolto, a costo di prestazioni molto scadenti, usando in uscita una funzione logica con lo stadio finale "***open collector***", che, come sappiamo, elimina i danni elettrici del conflitto di livello ma peggiora nettamente la velocità di commutazione.

Espansione del numero delle locazioni e di quello dei bit.

Il sistema di memoria previsto nel **caso 3** richiede evidentemente una combinazione delle due soluzioni già viste.
Prima bisogna costruire quattro "***banchi***" di memoria da 4 K *locazioni* da 16 bit usando 4 chip, collegati come nel caso 1 e, poi, collegarli seguendo la tecnica del caso 2 in cui ciascun *banco* viene usato come un chip di capacità quadrupla.
Ovviamente tutti i chip debbono avere il CS e le uscite *tristate*.
Questa ipotesi è assolutamente scontata; tutti i chip in commercio sono così.
Il passaggio attraverso la struttura di figura 6 per arrivare a quella di figura 12 è stato fatto solamente per sottolineare, in chiave didattica, l'importanza del concetto del CS e dei *driver tristate* sulle uscite.
Lo schema a blocchi della memoria relativa al caso 3, si presenterà esattamente come quello di figura 11, a patto di sostituire a ciascun chip della figura, un blocco di 4 chip del tipo previsto.

Il concetto di "*pagina*" di memoria.

Il metodo generale di gestione dell'indirizzo di un sistema di memoria prevede due matrici di selezione, una interna ai chip od ai banchi di locazioni, che utilizza i bit meno significativi dell'indirizzo ed un'altra, esplicita, che usa i bit più significativi per selezionare il gruppo di locazioni a cui corrisponde l'indirizzo completo, attraverso il CS. **Questo meccanismo di ricerca può essere assimilato a quello da usare in un libro, in cui ciascuna informazione occupi sempre un rigo (e non più di un rigo) di una pagina e ciascuna pagina contiene sempre e comunque un certo numero di righi**.
A questo punto qualsiasi informazione può essere univocamente trovata quando si conosce il numero della pagina ed il numero del rigo.
Il gruppo di bit meno significativi dell'indirizzo, che viene distribuito identicamente ai blocchi di locazioni (banchi o chip) individuano la "***parola***" della memoria (il rigo nelle pagine del libro); gli altri selezionano la "***pagina***" giusta.
Il solo indirizzo di rigo selezionerebbe contemporaneamente lo stesso rigo in tutte le pagine del libro, dando luogo ad una informazione in uscita ottenuta dalla sovrapposizione dei contenuti di

tutti i righi selezionati. Solo se l'indirizzo di rigo è combinato con quello di pagina si ha l'informazione giusta.
E' evidente che in un sistema di memoria con 16 bit d'indirizzo e *pagine* da 4 K (4096 locazioni), tutte le locazioni di una *pagina* hanno i 4 bit più significativi dell'indirizzo uguali.
La pagina, nella quale si è letta o scritta l'ultima informazione, si dice di solito "***pagina corrente***".

Selezione a coincidenza.

La tecnica per individuare una locazione in un sistema di memoria complesso è, a tutti gli effetti, una ***selezione a coincidenza***. La parola "coincidenza" non è altro, infatti, che una delle possibili traduzioni italiane del concetto di AND logico. La ricerca della locazione che è selezionata dai bit meno significativi dell'indirizzo e, contemporaneamente (AND logico), è contenuta nella *pagina* selezionata dai bit più significativi, è tecnicamente una *selezione a coincidenza*.
Ma, come avviene per molti meccanismi in elettronica digitale, la *selezione a coincidenza* può essere usata a più livelli nella stessa struttura.

Per maggiore chiarezza affrontiamo il discorso, partendo da una affermazione di tipo molto generale: **qualsiasi dispositivo che per funzionare ha bisogno del contemporaneo verificarsi di due distinte condizioni, può essere oggetto di una *selezione a coincidenza***.
Esempio: una lampada ad incandescenza da 12 Volt si accende quando uno dei suoi contatti elettrici è al potenziale di riferimento (***terra***), e l'altro è alla tensione nominale attraverso un interruttore chiuso.
In figura 13 è mostrato come una lampada può essere selezionata con la tecnica della "***coincidenza***".

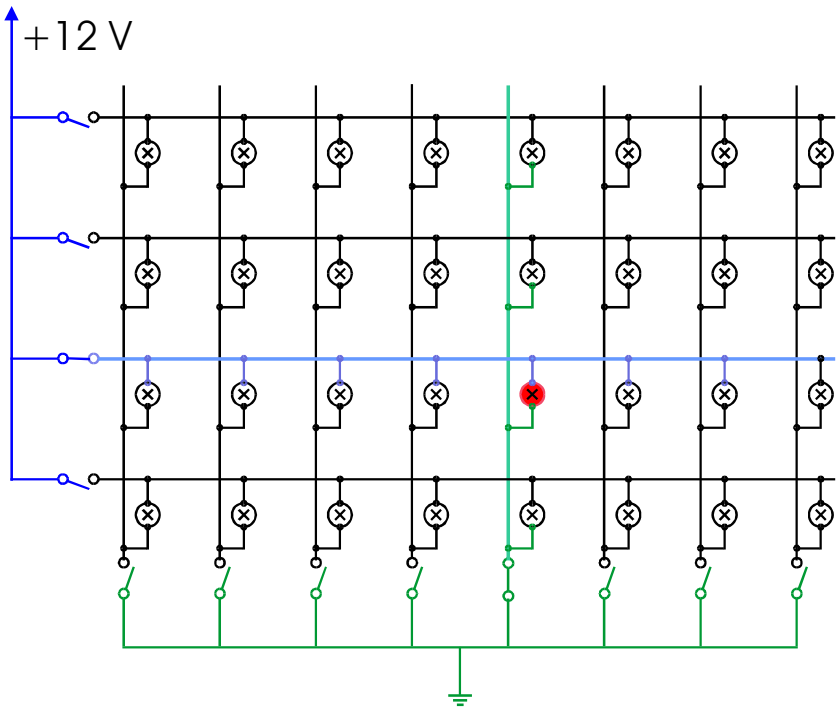

Fig. 13
L'unica lampada che è comune alla riga alimentata con 12 V ed alla colonna che ha un capo a "***terra***" è accesa, tutte le altre "***semi selezionate***" sono spente.

Un'altra semplice considerazione è che qualsiasi dispositivo logico può essere adattato alla selezione a coincidenza con l'inserimento di un AND logico.
La figura 14 mostra come un decodificatore in grado di selezionare 1 elemento tra 64 (6 bit d'indirizzo) può essere realizzata mediante due decodificatori da 1 su 8 (3 bit d'indirizzo) ed una matrice di 64 AND a 2 ingressi.

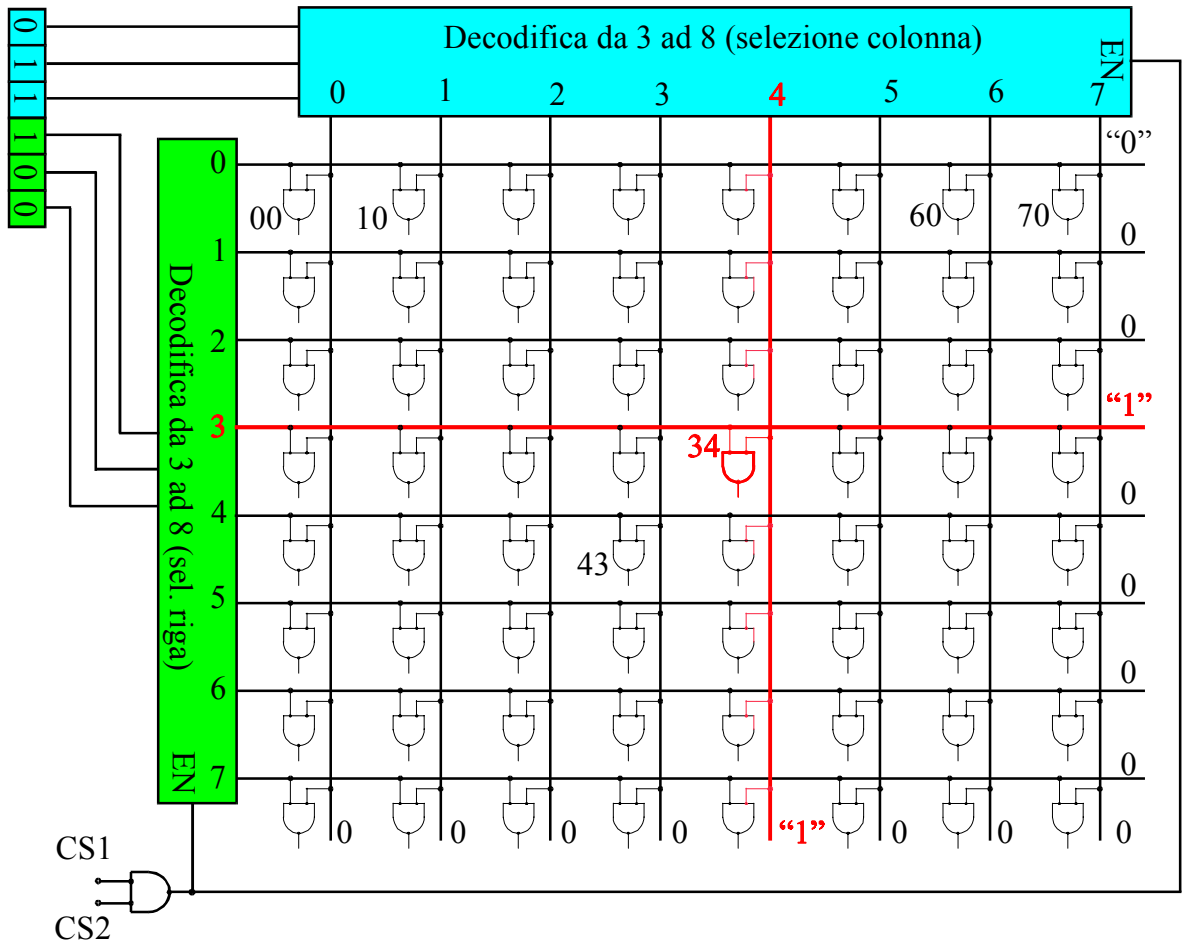

Fig. 14

E' molto facile capire che il concetto può essere estrapolato in 3 dimensioni, usando 3 decodificatori a 2 bit ed una matrice di 16 AND a 3 ingressi.

In pratica la stessa tecnica, viene usata, magari a più livelli, nella matrice di selezione interna del chip e poi in quella esterna per la decodifica completa dell'indirizzo nel sistema di memoria.

Tecniche alternative per la sintesi di una funzione booleana
A conclusione di questo capitolo dedicato alle memorie si può finalmente riprendere un discorso gia sviluppato in precedenza sulle tecniche alternative di sintesi di una funzione booleane

La tecnica base per la sintesi di una funzione booleana è quella di estrarre dalla tabella di verità la forma canonica disgiuntiva o congiuntiva e, dopo averla minimizzata, implementarla con le funzioni logiche elementari disponibili.

Abbiamo già visto che esistono soluzioni alternative, quella che usa una decodifica ed un OR, quella che usa un multiplexer con un numero di bit di selezione pari alle variabili o ridotto di una unità.
Tutte queste tecniche hanno in comune la caratteristica che la funzione viene implementata direttamente in forma canonica.
Abbiamo visto infatti che, **qualora la funzione non sia in forma canonica le espressioni algebriche che non contengono tutte le variabili vanno scomposte in prodotti completi**

Tabella funzione (Look-up table)
Esiste un'altra tecnica di sintesi di una funzione booleana che è ancora più semplice e diretta di quelle viste, si applica registrando in una memoria con un adeguato numero di locazioni direttamente la tabella di verità.
Supponiamo di avere una funzione di 4 variabili, assegnata attraverso la sua tabella di verità. Se si dispone di una memoria, meglio se si tratta di una ***Read Only Memory*** (**ROM**) ***programmabile*** (**PROM**), a 16 locazioni da 1 bit, basta **scrivere in ciascuna locazione, corrispondente ad una combinazione delle variabili interpretata come indirizzo, "0" o "1", secondo il contenuto della tabella di verità e la funzione è inplementata.**
Ponendo sulle linee d'indirizzo una combinazione delle variabili ed attivando la linea di lettura si promuoverà in uscita il contenuto della cella corrispondente e, cioè 0 o 1 secondo la tabella di verità.
Nella figura 15 è rappresentata l'implementazione di una funzione con la tecnica della "look-up table"

I	A_3	A_2	A_1	A_0	f
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

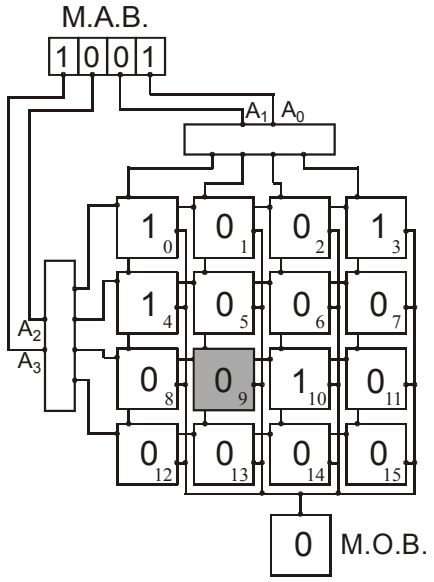

Fig. 15

Il registro Memory Address Buffer ricevendo la combinazione delle variabili d'ingresso A_3 A_2* A_1* A_0, indirizza la cella 9 e nel registro Memory Output Buffer compare il contenuto della locazione che è 0.
E' ovvio che l'ingresso del M.O.B. è collegato alla linea che collega tra loro i buffer tristate di uscita di tutte le locazioni mono-cella della memoria. La cosa è legittima perché il sistema di selezione garantisce che non saranno mai attivate contemporaneamente due celle.