Capitolo 1

Dispostivi Logici Programmabili

Il progetto di sistemi digitali, fino a non molti anni fa, era basato sull'utilizzo di circuiti logici standard a bassa ed a media scala di integrazione, della serie 54/74. Come mostra la Tabella 1.1 i circuiti integrati della serie 54/74 includono porte logiche, funzioni combinatorie come multiplexer e decodificatori, circuiti aritmetici, registri, contatori ecc.

Questi circuiti integrati furono inizialmente prodotti in tecnologia TTL standard ed in seguito in tecnologia TTL Schottky e Schottky low-power. Attualmente sono disponibili anche in versione CMOS. Nei circuiti integrati commercialmente disponibili, la x in Tabella 1.1 è sostituita da una o più lettere che consentono di identificare la tecnologia con la quale il circuito è stato realizzato. Le sigle corrispondenti sono riportate in Tabella 1.2. A titolo di esempio, un circuito integrato caratterizzato dalla sigla *74ACT02* include quattro porte NOR a due ingressi, realizzate in tecnologia CMOS "avanzata", con livelli logici TTL compatibili. I due prefissi "74" e "54" vengono utilizzati per distinguere rispettivamente le versioni commerciali e militari; queste ultime possono operare in un campo di temperature e di tensioni di alimentazioni più esteso.

Ovviamente, alle diverse tecnologie realizzative corrispondono differenti caratteristiche elettriche (tempi di propagazione, potenza dissipata, fan-out ecc.). Nonostante ciò, per motivi storici, è invalso l'uso di chiamare le porte logiche standard come "dispositivi TTL", indipendentemente dalla tecnologia adoperata.

Il progetto di un sistema digitale basato su circuiti logici standard prevede fondamentalmente i tre passi seguenti:

- Definizione delle specifiche.
- Descrizione del sistema mediante interconnessione di circuiti logici standard della serie 54/74.
- Assemblaggio dei circuiti integrati su di un circuito stampato.

Questo approccio, al crescere della complessità del sistema da progettare, richiede un numero sempre maggiore di circuiti SSI o MSI, con aumenti di costo, potenza dissipata e ritardi di propagazione oltre che di ingombro. Un sistema costituito da dispositivi 54/74 può inoltre essere facilmente "clonato", visto che ogni circuito integrato realizza una funzione logica ben nota.

Serie 54/74	Descrizione
74x00	Quattro porte NAND a due ingressi
74x02	Quattro porte NOR a due ingressi
74x30	Porta NAND ad otto ingressi
74x138	Decodificatore $3 \rightarrow 8$
74x153	Doppio multiplexer a 4 ingressi
74x283	Addizionatore a 4 bit
74x74	Doppio flip-flop D positive-edge-triggered
74x174	Registro a 6 bit
74x163	Contatore a quattro bit
74x194	Registro a scorrimento a 4 bit

Tabella 1.1: Sigle di alcuni circuiti logici standard della serie 54/74

Sigla	Famiglia logica
S	TTL Schottky
LS	TTL Low-power Schottky
AS	TTL Advanced Schottky
ALS	TTL Advanced Low-power Schottky
F	TTL Fast
HC	CMOS High speed
HCT	CMOS High speed, TTL compatibile
AC	CMOS Advanced
ACT	CMOS Advanced, TTL compatibile
FCT	CMOS Fast, TTL compatibile

Tabella 1.2: Famiglie logiche della serie 54/74

Per sopperire a queste limitazioni sono stati introdotti i dispositivi logici programmabili (PLD), che vengono impegati sempre più frequentemente al posto dei circuiti integrati TTL.

Un PLD è un circuito ad elevata scala di integrazione che può essere opportunamente *programmato* o *personalizzato* dall'utente finale, in modo da realizzare una specifica funzione.

I vantaggi legati all'utilizzo di PLD sono molteplici:

- Un singolo PLD può sostituire numerosi circuiti a bassa o media scala di integrazione, con significativi miglioramenti in termini di area occupata sul circuito stampato, di affidabilità e di costi.

- Le interconnessioni a livello di circuito stampato vengono sostituite da collegamenti all'interno di un singolo circuito integrato. La riduzione delle capacità parassite consente di migliorare sia i tempi di propagazione che la potenza dissipata.

- Un sistema basato su PLDè molto più flessibile rispetto ad uno realizzato con componenti logici discreti. Molti PLD sono infatti *riprogrammabili* elettricamente anche dopo essere stati collegati su di un circuito stampato

(*in-system programmability*). Ciò consente di modificare la funzionalità di un sistema digitale senza dover aggiungere o rimuovere componenti.

- Dopo aver programmato un PLD è possibile impedire che la configurazione interna del dispositivo sia leggibile dall'esterno, proteggendo in questo modo la riservatezza del progetto.

Il progetto di un sistema digitale basato su PLD consta di poche fasi, ben definite:

- Definizione delle specifiche.
- Descrizione del sistema, che può essere effettuata o mediante uno schema elettrico (*schematic-entry*) o mediante un opportuno linguaggio per la descrizione dell'hardware (*hardware description language, HDL*) o in un mix di *schematic-entry* e di HDL.
- Sintesi, ovvero passaggio dalla descrizione funzionale del sistema ad una descrizione di più basso livello che rispecchia la struttura interna del PLD.
- Ottimizzazione del circuito sintetizzato.
- Implementazione nel dispositivo logico programmabile prescelto o *fitting*.

Le fasi di sintesi, ottimizzazione e *fitting* sono realizzate con l'ausilio di opportuni sistemi di sviluppo che consentono di ottenere in maniera quasi del tutto automatica la mappa di programmazione del PLD a partire da una descrizione ad alto livello. In questo modo il ciclo di sviluppo di un sistema digitale diviene simile a quello di un programma software in cui si partire da una descrizione in un linguaggio di programmazione (C, FORTRAN ecc.) per ottenere, dopo una fase di compilazione ed ottimizzazione, una descrizione in linguaggio macchina.

Così come un programma descritto in un linguaggio ad alto livello può essere compilato e quindi eseguito su piattaforme hardware differenti, la descrizione del sistema digitale mediante *schematic-entry* o HDL è (quasi del tutto) indipendente dal particolare PLD prescelto per l'implementazione finale.

Se si adotta uno *schematic-entry*, gli elementi primitivi di cui si dispone vanno dalle semplici porte logiche elementari fino a funzioni più complesse come decodificatori, addizionatori, contatori, registri a scorrimento ecc. Molto spesso i nomi di questi blocchi logici sono gli stessi delle porte TTL, in modo da semplificare al massimo la "migrazione" di un sistema da logiche standard a PLD.

Linguaggi standard per la descrizione dell'hardware sono il VHDL ed il VERILOG. Accanto a questi sono molto diffusi altri linguaggi, sviluppati da case costruttrici di dispositivi programmabili o di sistemi di sviluppo. Nel seguito studieremo in qualche dettaglio il linguaggio ABEL (acronimo di *Advanced Boolean Expression Language*), sviluppato dalla Data I/O Corporation e divenuto uno standard molto diffuso a livello industriale grazie alla sintassi molto semplice ed intuitiva.

I primi PLD che studieremo consentono di realizzare semplici funzioni combinatorie. Vedremo quindi come è possibile estenderne le funzionalità utilizzando delle opportune *macrocelle* di uscita. Passeremo quindi all'analisi dei circuiti logici programmabili più avanzati (*Complex PLD* o CPLD) per accennare infine alla struttura interna delle matrici di porte programmabili (*Field-Programmable Gate array* o FPGA).

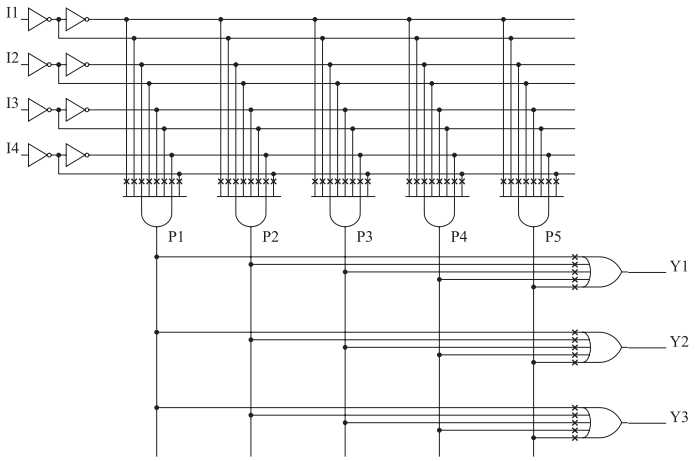

Figura 1.1: Schema di una PLA con 4 ingressi, 5 mintermini e 3 uscite

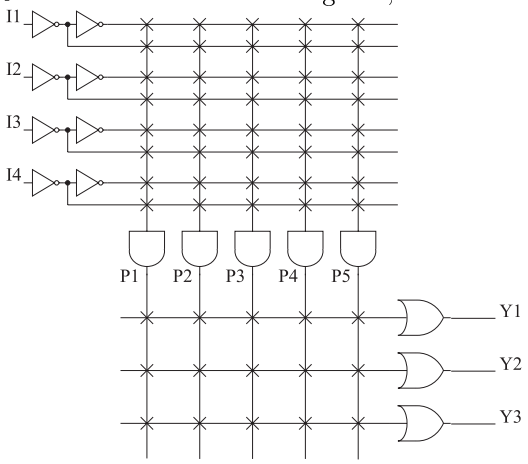

Figura 1.2: Schema compatto di una PLA con 4 ingressi, 5 mintermini e 3 uscite

1.1 Matrici Logiche Programmabili (PLA)

I dispositivi logici programmabili più semplici prendono il nome di PLA, acronimo di *Programmable Logic Array*. Un PLA è in grado di realizzare funzioni logiche combinatorie, espresse come somma di prodotti. I parametri che caratterizzano una PLA sono:

- Il numero di ingressi n.
- Il numero di termini prodotto o *mintermini* p.
- Il numero di uscite m.

In generale, il numero p di mintermini disponibili in una PLA è molto minore rispetto a tutti i possibili mintermini realizzabili con n variabili, pari a 2^n. Pertanto una PLA non può realizzare una qualsiasi funzione logica ad n ingressi ed m uscite; la sua utilità è limitata alle funzioni logiche esprimibili mediante un numero di mintermini minore o uguale a p.

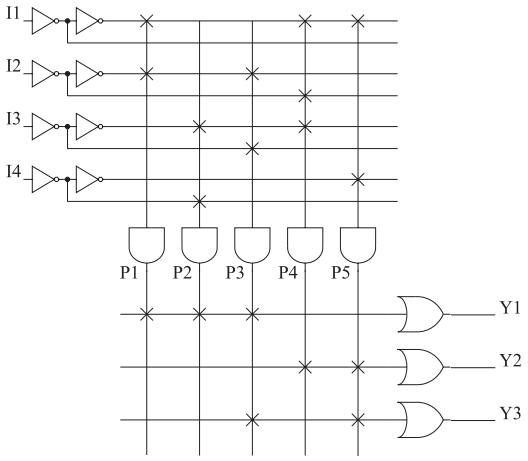

Figura 1.3: PLA programmata.

La Figura 1.1 mostra lo schema di una piccola PLA con quattro ingressi, cinque mintermini e tre uscite ($n = 4$, $p = 5$ e $m = 3$).

Tramite una coppia di invertitori, ogni ingresso ed il proprio negato sono collegati all'ingresso di un *piano AND*, costituito da p porte AND a $2n$ ingressi. Le 'X' sulle linee di ingresso delle porte AND denotano dei *collegamenti programmabili* mediante i quali è possibile ottenere p distinti termini prodotto. Qualora un ingresso delle porte AND non sia collegato ad alcun segnale, un circuito di pull-up (non mostrato in figura) mantiene l'ingresso stesso a livello logico alto.

Le uscite del piano AND sono inviate in ingresso al *piano OR*, costituito da m porte OR a p ingressi. Anche in questo caso le 'X' sulle linee di ingresso delle porte OR denotano dei collegamenti programmabili, mediante i quali è possibile sommare fra loro i termini prodotto per ottenere le n uscite. In questo caso, se un ingresso delle porte OR non è collegato ad alcuna uscita del piano AND, un circuito di pull-down (non mostrato in figura) mantiene l'ingresso della OR a livello logico basso.

La Figura 1.2 mostra uno schema semplificato della stessa PLA, in cui gli ingressi di ogni porta AND ed OR sono rappresentati mediante un singola linea.

La PLA di Figura 1.2 può essere utilizzata per realizzare fino a tre funzioni logiche di 4 ingressi esprimibili come somma di al massimo 5 minitermini, come ad esempio:

$$
\begin{aligned}
Y1 &= I1I2 \ + I3\overline{I4} \ + I2\overline{I3} \\
Y2 &= I1\overline{I2}I3 \ + I1I4 \\
Y3 &= I2\overline{I3} \ + I1I4
\end{aligned}
$$

Si noti che queste equazioni hanno un totale di 7 mintermini. Peraltro il fattore $I2\overline{I3}$ è comune alla prima ed alla terza equazione, mentre il fattore $I1I4$ è comune alla seconda ed alla terza equazione. In definitiva si hanno soltanto 5 distinti termini prodotto e le funzioni sono pertanto realizzabili con la PLA di Figura 1.2, come evidenzia la Figura 1.3 che mostra le connessioni necessarie.

La PLA di Figura 1.2 ha un numero di ingressi, uscite e termini prodotto troppo limitato per poter essere effettivamente utilizzata. Il numero di termini prodotto disponibili in PLA commerciali è generalmente compreso fra 4 e 16,

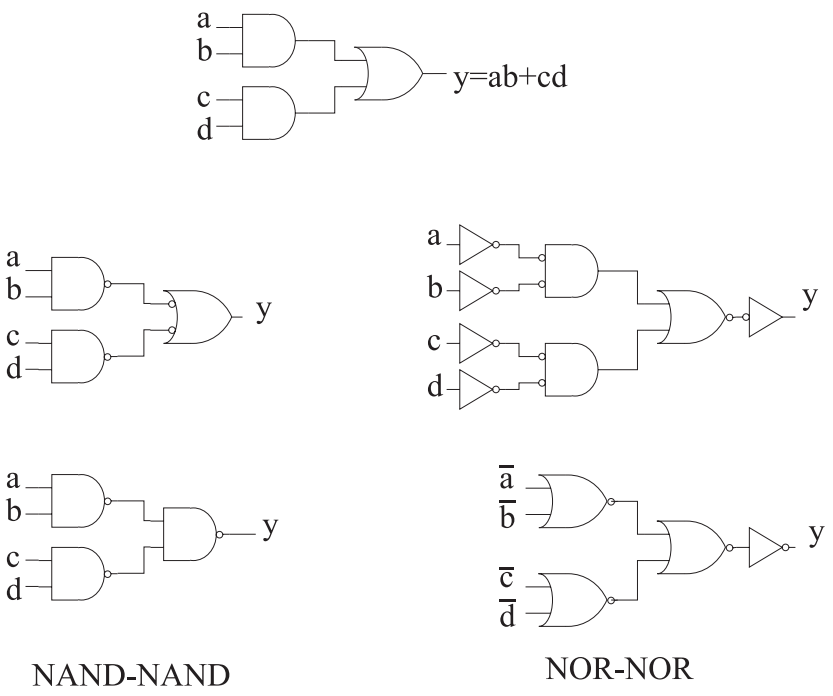

Figura 1.4: Trasformazione di una funzione logica espressa come somma di prodotti in forma NAND-NAND o NOR-NOR.

indipendentemente dal numero di ingressi, in modo da limitare il numero di interconnessioni programmabili e la complessità del piano AND.

1.2 Tecnologie Realizzative

Abbiamo visto che una PLA è costituita da un insieme di porte AND e di porte OR opportunamente interconnesse. Per semplificare la realizzazione circuitale di una PLA è senz'altro opportuno trasformare preventivamente le funzioni logiche in forma NAND-NAND oppure NOR-NOR. Consideriamo, ad esempio, la funzione:

$$y = ab + cd$$

La Figura 1.4 mostra i semplici passi da compiere per trasformare la funzione assegnata in modo da utilizzare soltanto porte NAND oppure soltanto porte NOR. Si noti che la realizzazione in forma NOR-NOR richiede di effettuare una negazione dei segnali di ingresso. Nel caso di una PLA ciò non comporta alcuna complicazione aggiuntiva, grazie alla presenza della coppia di invertitori all'ingresso del piano AND (vedi Figura 1.1).

1.2.1 Cicuiti Bipolari

Nel caso di PLA realizzate in tecnologia bipolare si utilizzano due piani di tipo NAND. Entrambi vengono realizzati mediante delle matrici di diodi (che realizzano funzioni di tipo AND) con invertitori in uscita.

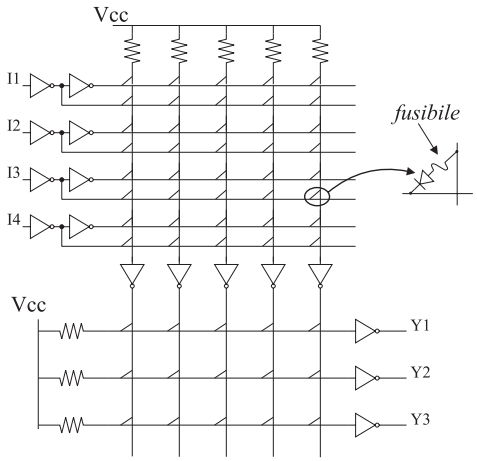

Figura 1.5: PLA in tecnologia bipolare.

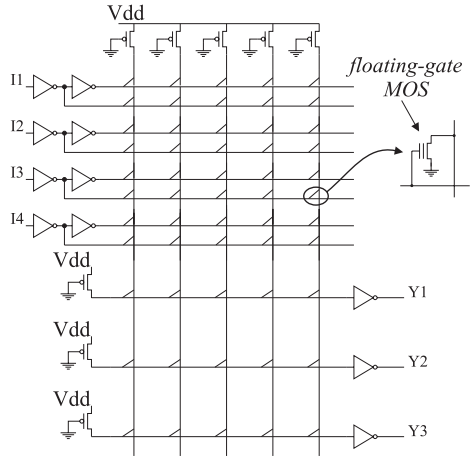

Figura 1.6: PLA in tecnologia MOS.

Un esempio di PLA a diodi è mostrato in Figura 1.5. Si noti che in serie ai diodi è presente un opportuno *fusibile*, realizzato con un collegamento metallico di sezione opportunamente ridotta. Per programmare una PLA di tipo bipolare è necessario selezionare, mediante opportuni segnali di ingresso, i fusibili da disconnettere ed applicare quindi un'opportuna tensione di programmazione (10-30V) ai loro capi. In questo modo i collegamenti metallici dei fusibili selezionati si interrompono, programmando la PLA. Ovviamente, l'operazione di programmazione di una PLA bipolare è irreversibile: il circuito è programmabile una sola volta.

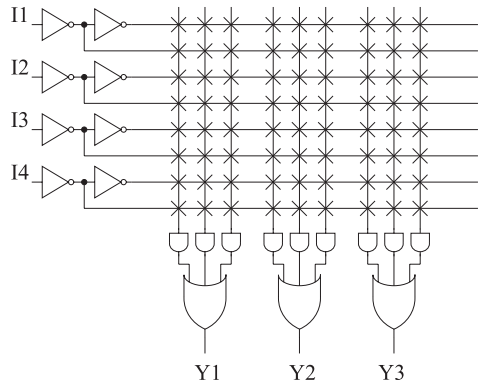

Figura 1.7: Schema compatto di una semplice PAL.

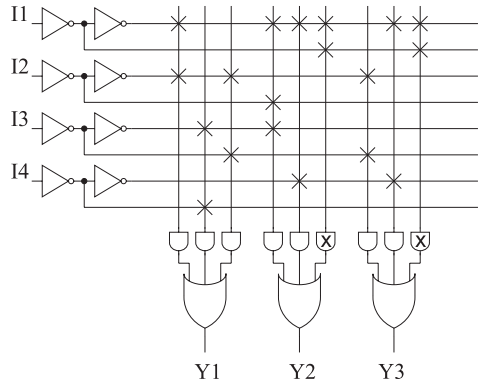

Figura 1.8: PAL programmata per realizzare le stesse funzioni della PLA di Figura 1.3.

1.2.2 Circuiti MOS

Per minimizzare l'occupazione di area, le PLA in tecnologia MOS utilizzano due piani di tipo NOR , realizzati in logica NMOS o pseudo-NMOS.

Come mostra la Figura 1.6, per programmare la PLA vengono spesso utilizzati dispositivi MOS con doppia gate, una delle quali isolata (gate "fluttuante" o *floating-gate*). Questa tecnologia, mutuata da quella delle memorie EPROM ed EEPROM, consente di riprogrammare più volte la PLA. Si rimanda al capitolo sulle memore per una disamina delle modalità di programmazione e del principio di funzionamento delle memorie EPROM ed EEPROM.

1.3 Dispositivi PAL

Gli svantaggi principali delle prime PLA disponibili commercialmente erano gli elevati costi di produzione ed i notevoli tempi di propagazione. Queste limitazioni sono dovute alla presenza di due piani AND ed OR *entrambi programmabili*. La presenza di matrici programmabili, infatti, richiede una elevata occupazione di area, una tecnologia più avanzata e comporta sensibili ritardi di propagazione.

Per ovviare a questi inconvenienti sono stati introdotti i dispositivi PAL

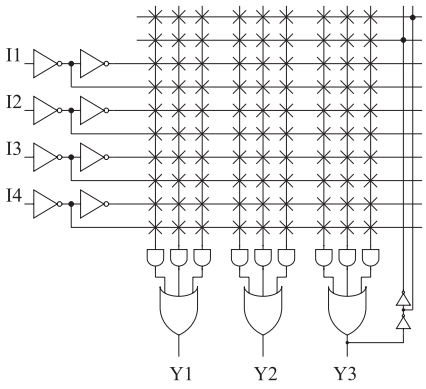

Figura 1.9: Schema di una PAL che consente di realizzare una logica in due passi.

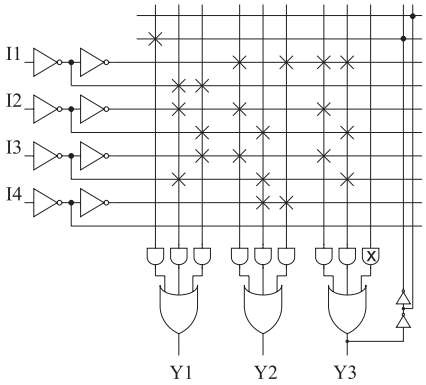

Figura 1.10: Programmazione della PAL di Figura 1.9

(acronimo di *Programmable Array Logic*). Mentre in una PLA sono programmabili sia il piano AND che il piano OR, una PAL è caratterizzara dall'avere il piano OR *fisso*, per cui il solo piano ANDè programmabile.

La Figura 1.7 mostra un esempio di PAL. Per il piano AND si è utilizzata la notazione compatta di Figura 1.2. Nella semplice PAL di Figura 1.7 si hanno 4 ingressi e 3 uscite mentre il numero di termini prodotto per ogni uscita è pari a 3. Le uscite del piano AND non possono essere condivise da più porte OR. Se un termine prodotto è necessario per due uscite deve essere calcolato due volte, come mostra la Figura 1.8 che riporta la PAL programmata per realizzare le stesse funzioni della PLA di Figura 1.3. Si noti che per le uscite $Y2$ ed $Y3$ un termine prodotto è collegato sia alla variabile $X1$ che a $\overline{X1}$ di modo che l'uscita della corrispondente porta AND sia sempre 0. Come evidenzia la Figura 1.8, una AND programmata per avere uscita sempre 0 viene evidenziata con una 'X' sullo schema circuitale.

1.3.1 Logica in due passi

Nelle PAL commerciali, il numero di termini prodotto disponibili per ogni uscita è compreso fra 8 e 16. Per consentire la realizzazione di funzioni logiche

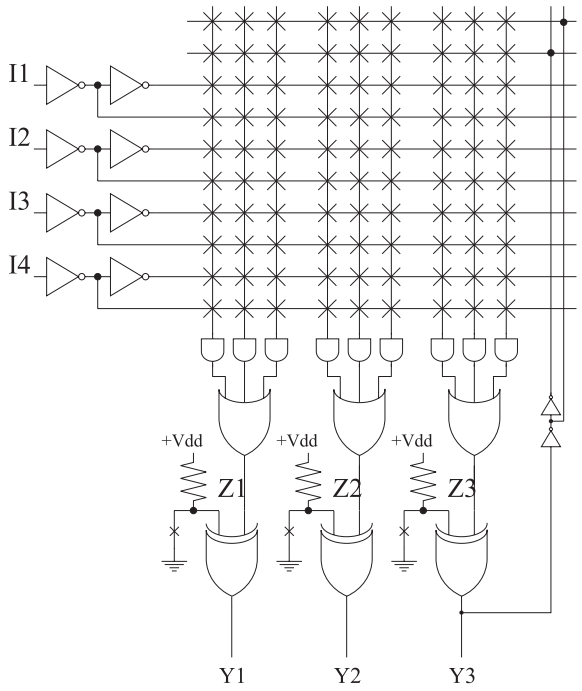

Figura 1.11: PAL con polarità dell'uscita programmabile.

che richiedono un numero maggiore di mintermini si utilizza una *logica in due passi*. Come mostra la Figura 1.9, alcune delle uscite della PAL (in questo caso la sola $Y3$) vengono nuovamente inviate come ingressi del piano AND; ciò consente di realizzare funzioni più complesse al prezzo di un maggior ritardo di propagazione.

Consideriamo, ad esempio, il caso in cui la PAL di Figura 1.9 debba essere utilizzata per realizzare le due funzioni seguenti:

$$Y1 = I1I2I3 + I1\overline{I2}\,\overline{I3} + \overline{I1}I2\overline{I3} + \overline{I1}\,\overline{I2}I3$$
$$Y2 = I1I2I3 + \overline{I2}\,\overline{I3}I4 + I1I4$$

L'uscita $Y2$ richiede tre termini prodotto e rientra nei limiti della PAL. Il numero di termini prodotto per l'uscita $Y1$ è pari a 5, ed è pertanto maggiore dei termini prodotto disponibili nella nostra PAL per ogni uscita. Dobbiamo pertanto riscrivere l'espressione per l'uscita $Y1$, utilizzando come termine di "aiuto" (*helper term*) l'uscita $Y3$:

$$Y3 = I1I2I3 + I1\overline{I2}\,\overline{I3}$$
$$Y1 = Y3 + \overline{I1}\,I2\overline{I3} + \overline{I1}\,\overline{I2}I3$$

La corrispondente mappa di programmazione è mostrata in Figura 1.10.

La logica a due passi aumenta la flessibilità di una PAL, a prezzo di un maggior ritardo di propagazione. Per la PAL di Figura 1.10, il ritardo per l'uscita $Y1$ è all'incirca il doppio del ritardo relativo all'uscita $Y2$.

1.3.2 Programmazione della polarità dell'uscita

Supponiamo di dover utilizzare la PAL di Figura 1.9 per realizzare le due funzioni seguenti:

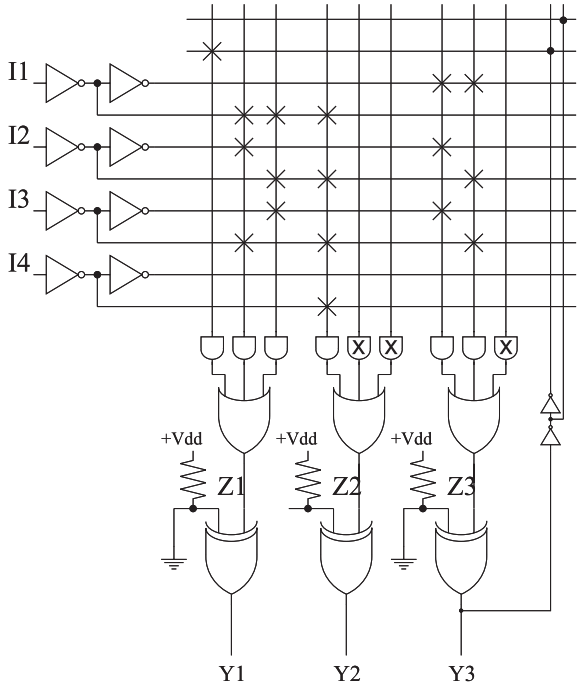

Figura 1.12: Programmazione della PAL di Figura 1.11.

$$Y1 = I1I2I3 \; + I1\overline{I2}\;\overline{I3} \; + \overline{I1}I2\overline{I3} \; + \overline{I1}\;\overline{I2}I3$$
$$Y2 = I1 \; + I2 \; + I3 \; + I4$$

Rispetto all'esempio del paragrafo precedente, il numero di termini prodotto per l'uscita $Y2$ è pari a 4, ed è pertanto maggiore dei termini prodotto disponibili nella nostra PAL. Inoltre, non è possibile utilizzare l'uscita $Y3$ come come termine di "aiuto", dato che $Y3$ è necessaria al calcolo di $Y1$. In definitiva, le due funzioni non possono essere realizzate con la PAL di Figura 1.9. D'altro canto, se dovessimo calcolare $\overline{Y2}$ invece di $Y2$ ogni problema sarebbe risolto, visto che $\overline{Y2}$ richiede la valutazione di un solo termine prodotto:

$$\overline{Y2} = \overline{I1}\;\overline{I2}\;\overline{I3}\;\overline{I4}$$

Molti PLD dispongono della possibilità di poter *programmare la polarità dell'uscita*, come mostra la Figura 1.11. Ogni uscita è provvista di una porta XOR. Un ingresso della XOR è collegato all'uscita del piano OR. L'altro ingresso della XOR è collegato a livello logico alto tramite un resistore di pull-up, ma può essere portato a massa in fase di programmazione del dispositivo, tramite un'opportuna interconnessione programmabile. In questo modo è possibile invertire l'uscita della PAL. Infatti, indicando con Z l'uscita del piano OR e con Y l'uscita della XOR, quando il secondo ingresso della XOR è a zero risulta:

$$Y = Z \; XOR \; 0 = Z$$

mentre se il secondo ingresso della XOR è programmato in modo rimanere a livello logico alto si ha:

$$Y = Z \; XOR \; 1 = \overline{Z}$$

La Figura 1.12 mostra la mappa di programmazione della PAL, per realizzare le due funzioni $Y1$ ed $Y2$ assegnate.

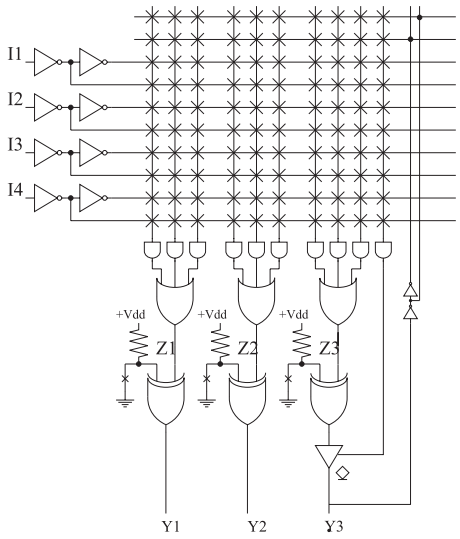

Figura 1.13: PAL con I/O programmabile.

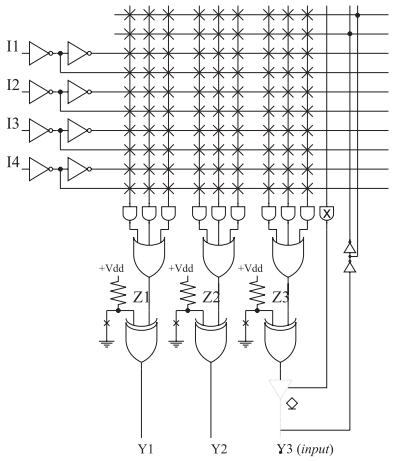

Figura 1.14: Utilizzo di un pin di I/O programmabile come ingresso.

1.3.3 Terminali di Ingresso/Uscita programmabili.

La Figura 1.13 mostra la PAL di Figura 1.11 in cui l'uscita $Y3$ è utilizzata come terminale di ingresso/uscita programmabile (*I/O pin*). Dopo la porta XOR che ralizza l'inversione programmabile è presente un buffer di uscita tristate. Il segnale di abilitazione della porta tristate è pilotato da una uscita dedicata del piano AND; se quest'uscita è 0 il buffer si porta in condizioni di alta impedenza. Un pin di I/O può essere utilizzato in tre modi diversi:

1. Se il circuito è programmato in modo tale che il segnale di abilitazione del buffer tristate sia sempre 0 (figura 1.14), il pin corrispondente può essere utilizzato come ingresso, sfruttando la coppia di invertitori necessaria per realizzare la logica a due passi.

2. La figura 1.15 mostra il caso in cui tutti gli ingressi della porta AND che pilota il segnale di abilitazione del buffer tristate sono disconnessi. In questo caso degli elementi di pull-up, non mostrati in figura, mantengono a livello logico alto gli ingressi della porta AND, per cui il segnale di abilitazione della porta tristate è sempre 1. Il pin corrispondente può essere utilizzato come uscita oppure come *helper* per realizzare una logica a due passi.

3. In figura 1.15 il segnale di abilitazione del buffer tristate dipende dagli ingressi del PLD. In questo caso il pin può essere dinamicamente utilizzato come terminale di ingresso o di uscita. Una tipica applicazione è quella di un'interfaccia collegata ad un bus condiviso con altre periferiche.

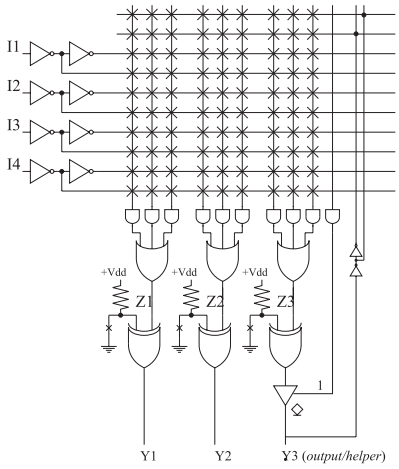

Figura 1.15: Utilizzo di un pin di I/O programmabile come uscita o come termine di *helper* pr logiche in due passi.

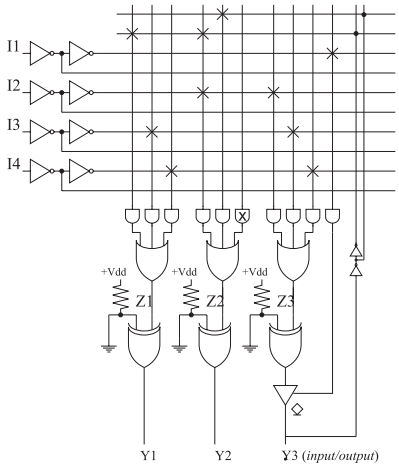

Figura 1.16: Terminale utilizzabile dinamicamente come ingresso o come uscita. Quali sono le funzioni logiche realizzate da questa PLD?

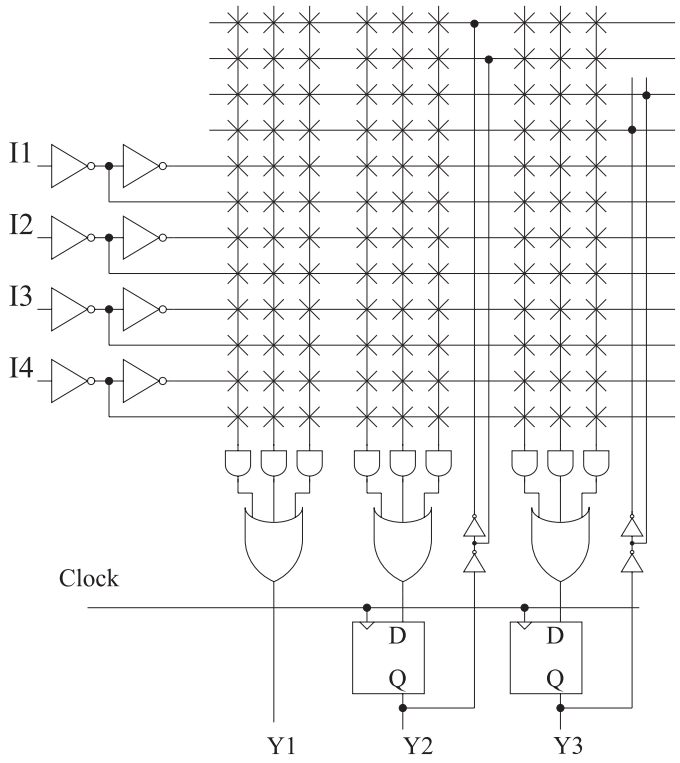

Figura 1.17: PAL sequenziale.

La disponibilità di terminali di I/O programmabili aumenta notevolmente la flessibilità di un PLD. Ad esempio una PLA con 10 terminali di I/O programmabili può essere utilizzata, a seconda delle esigenze, per realizzare 6 funzioni logiche di quattro ingressi, oppure quattro funzioni logiche di quattro ingressi con due termini di *helper* oppure tre funzioni logiche di quattro ingressi con tre termini di *helper* ecc.

1.3.4 PLD sequenziali

I circuiti logici programmabili sequenziali includono al loro interno dei flip-flop, in modo da poter realizzare sistemi sequenziali come contatori, registri a scorrimento ecc.

Un esempio di PAL sequenziale è riportato in Figura 1.17. In corrispondenza di $Y2$ ed $Y3$ sono inseriti due flip-flop. Le uscite dei due flip-flop, oltre che essere disponibili come terminali di uscita della PAL, sono inviate come feedback all'interno del piano AND, in modo da consentire la realizzazione di macchine a stati. Si noti che il segnale di clock *è comune a tutti i flip-flop*, come dovrebbe essere in ogni sistema sincrono ben progettato.

1.3.5 Macrocelle di uscita

Tutti gli accorgimenti visti nei paragrafi precedenti per aumentare la flessibilità delle PAL combinatorie possono essere utilizzati anche nelle PAL sequenziali. La Figura 1.18 mostra l'introduzione dell'inversione programmabile dell'uscita e dei terminali di I/O programmabili per la PAL di Figura 1.17.

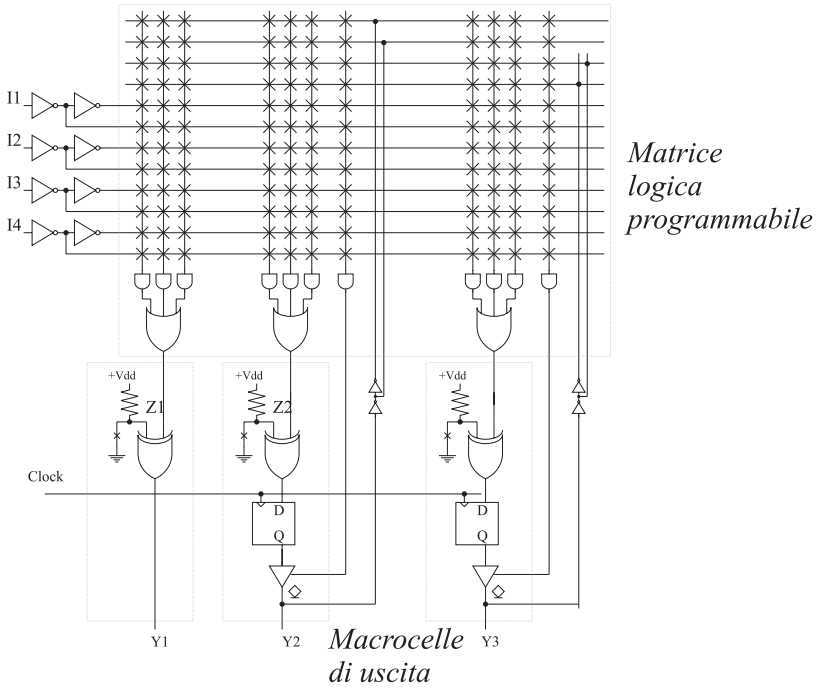

Figura 1.18: PAL sequenziale con macrocella di uscita.

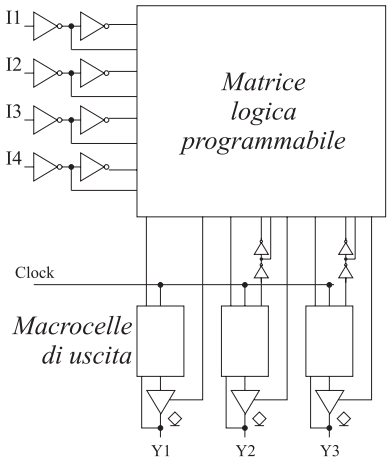

Figura 1.19: Schema a blocchi semplificato della PAL di figura 1.18.

La Figura 1.18 evidenzia come una PAL possa in generale essere suddivisa in:

- Una matrice logica programmabile, costituita da un piano AND programmabile e da un piano OR fisso. Gli ingressi della matrice programmabile sono costituiti sia da ingressi primari del circuito che da eventuali termini di feedback.
- Un insieme di *macrocelle* di uscita, anch'esse programmabili.

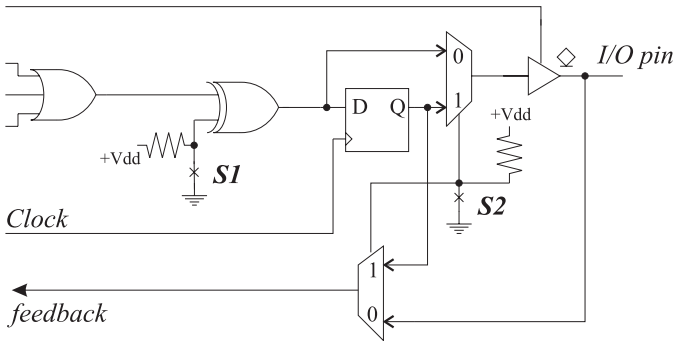

Figura 1.20: Macrocella di uscita di una PAL.

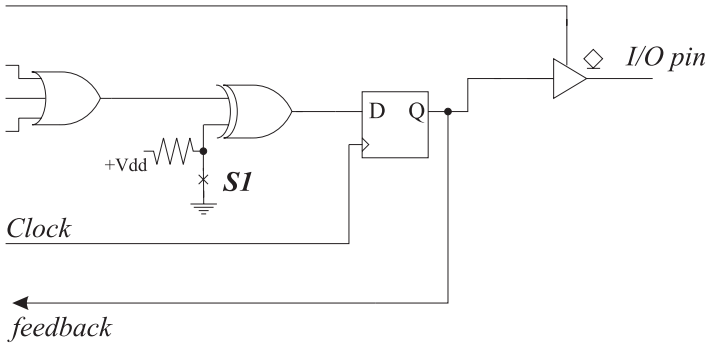

Figura 1.21: Macrocella di uscita in modalità "sequenziale".

La Figura 1.19 riporta un diagramma a blocchi della PAL di Figura 1.18.

Nell'esempio di Figura 1.18 le macrocelle di uscita consentono di invertire la polarità dell'uscita e di controllare i terminali di I/O programmabili. Peraltro, a causa dell'introduzione dei registri di uscita, non è possibile realizzare una logica in due passi. Inoltre non è possibile realizzare una macchina a stati le cui uscite possano essere poste in una condizione di alta impedenza. Ciò richiede, infatti, che i termini di feedback siano prelevati *all'ingresso* del buffer tristate.

Per risolvere questi inconvenienti è sufficiente modificare le macrocelle di uscita. La Figure 1.20 mostra lo schema di una macrocella più generale di quella di Figura 1.18. Due multiplexer, comandati dalla connessione programmabile S2, consentono di selezionare la modalità di funzionamento della macrocella.

In modalità "sequenziale" (vedi Figura 1.21) il segnale di feedback corrisponde con l'uscita Q del flip-flop. Si noti che in questa modalità operativa l'uscita della macrocella può essere posta in condizione di alta impedenza, grazie alla presenza del buffer tristate.

In modalità "combinatoriale" (vedi Figura 1.22) l'uscita della porta XOR (che realizza l'inversione programmabile di polarità) viene inviata direttamente al buffer di uscita. Il segnale di feedback corrisponde con l'uscita del buffer, in modo da poter realizzare una logica a due passi oppure utilizzare il pin come teminale di ingresso della PAL.

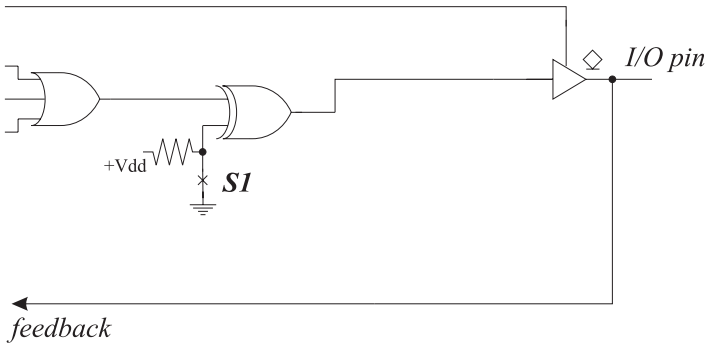

Figura 1.22: Macrocella di uscita in modalità "combinatoriale".

1.4 PAL commerciali

Esistono numerose versioni di PAL, che si differenziano in funzione del numero di ingressi, di uscite, di termini prodotto e della complessità della macrocella di uscita. Una fonte primaria di informazioni è ovviamente costituita dai *data sheet* dei costruttori, quasi tutti ormai disponibili on-line agli indirizzi internet citati in bibliografia.

Una delle più semplici PAL disponibili commercialmente è contraddistinta dalla sigla 16L8. Le due cifre nella sigla stanno ad indicare che il numero di ingressi del piano AND è pari a 16, mentre il dispositivo può essere configurato in modo da disporre di un numero massimo massimo di 8 uscite. La struttura interna di questa PAL, di tipo combinatoriale, è simile a quella riportata in Figura 1.13, senza peraltro la possibilità di inversione dell'uscita.

Come mostra lo schema semplificato di figura 1.23, Il dispositivo ha 10 terminali di ingresso, 2 terminali di uscita ed 6 terminali di I/O programmabili che possono essere utilizzati anche come *helper* per realizzare logiche a due passi. Il numero di termini prodotto per ogni uscita è pari a 7, mentre un'ulteriore AND con ingressi programmabili viene utilizzata per pilotare il buffer tristate di uscita.

Lo schema semplificato della PAL 16R4 è anch'esso riportato in Figura 1.23. La struttura è molto simile alla 16L8, con l'unica differenza data dalla presenza di quattro registri in corrispondenza di altrettanti terminali di uscita. Anche nella PAL 16R4 il numero di ingressi del piano AND è pari a 16, di cui 8 provengono dagli ingressi primari, 4 da terminali di I/O programmabili ed infine 4 sono i termini di feedback dai registri di uscita.

Sia la PAL 16L8 che la PAL 16R4 sono rappresentative delle prime generazioni di PLD. Un circuito più recente è costituito dalla PAL 16V8, il cui schema semplificato appare Figura 1.24. Le macrocelle di uscita sono molto simili a quelle di Figura 1.20, per cui ogni terminale di uscita può essere programmato in modo da operare come uscita "combinatoria" o "sequenziale", con o senza inversione di polarità.

La PAL 22V10 di Figura 1.25 è uno dei dispositivi più comunemente adoperati. Le due cifre della sigla indicano che la matrice AND programmabile ha un totale di 22 ingressi, mentre il circuito può essere programmato in modo da fornire fino ad un massimo di 10 terminali di uscita.

Dalla Figura 1.25 osserviamo che il dispositivo conta 12 terminali di ingresso, uno dei quali è utilizzato anche come clock, e 10 terminali di I/O programmabili.

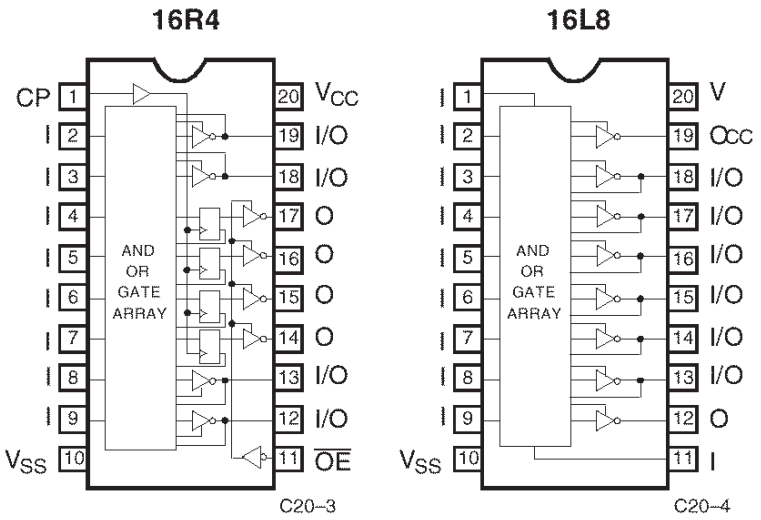

Figura 1.23: Schema semplificato delle PAL 16L8 e 16R4.

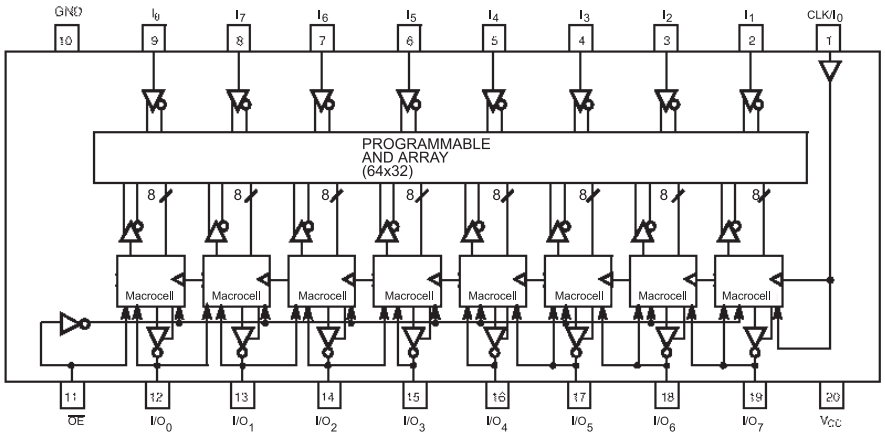

Figura 1.24: Schema semplificato delle PAL 16V8.

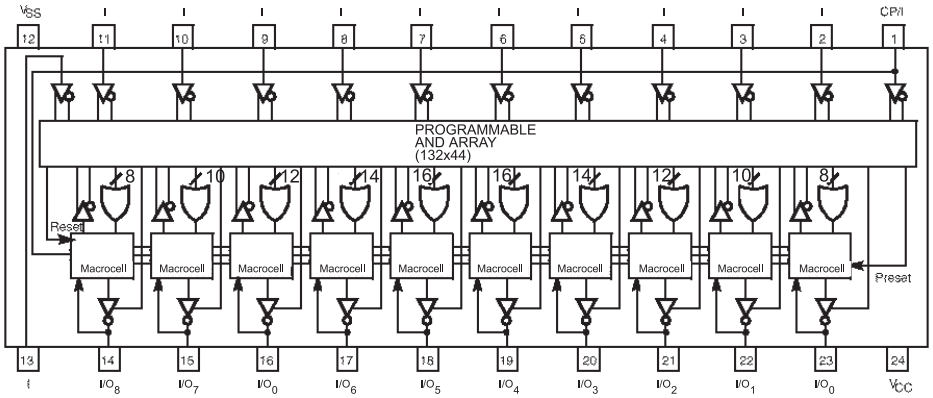

Figura 1.25: Schema semplificato della PAL 22V10.

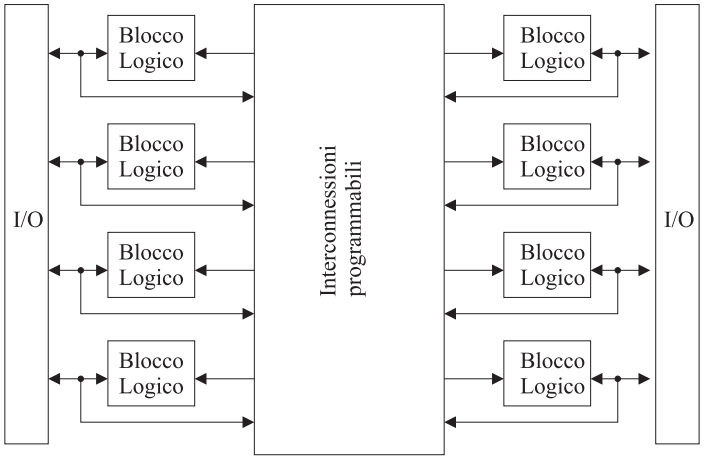

Figura 1.26: Schema di principio di una CPLD.

Da notare che il numero di termini prodotto disponibile per ogni uscita varia da un minimo di 8 ad un massimo di 16. In questo modo alcuni pin possono essere dedicati al calcolo di funzioni particolarmente complesse, senza aumentare eccessivamente le dimensioni complessive della matrice programmabile, con effetti benefici in termini di potenza dissipata, ritardo di propagazione ed area di silicio.

Lo schema riportato in Figura 1.25, lascia trasparire la complessità della PAL22V10. La mappa di programmazione di un PLD di questo tipo non può in pratica essere ottenuta manualmente. è necessario affidarsi ad un opportuno sistema di sviluppo che, a partire dalla descrizione del circuito logico che intendiamo realizzare, può sfruttare al meglio le potenzialità del circuito.

1.5 PLD complessi (CPLD)

Il progressivo sviluppo della tecnologia CMOS ha consentito la realizzazione di circuiti programmabili di complessità sempre maggiore. Un parametro utilizzato molto di frequente per stabilire la *capacità* di una PLD è il numero di *porte equivalenti*. Diremo che un dato PLD ha una capacità di 1000 porte equivalenti se consente di realizzare delle funzioni logiche per le quali sarebbero necessarie 1000 porte NAND a due ingressi. La definizione, per quanto non rigorosa, è in pratica molto utile in quanto, nelle fasi iniziali di un progetto, consente di valutare in prima approssimazione quanti PLD sono necessari per realizzare un dato sistema. Ad esempio, la PAL22V10 del paragrafo precedente ha un capacità dell'ordine di 800 porte equivalenti.

Nel caso dei circuiti di tipo PAL, *aumentare la capacità corrisponde ad aumentare il numero di ingressi del piano AND programmabile*. Ciò comporta un aumento insostenibile del fan-in delle porte che compongono il piano AND, con significativi peggioramenti in termini di tempi di propagazione. Ad esempio, nella PAL22V10 il piano AND è costituito da 132 porte a 44 ingressi.

Invece di aumentare il numero di ingressi del piano AND è più conveniente aumentare la capacità di un dispositivo logico programmabile integrando più *blocchi logici*, ognuno simile ad una PAL, su di uno stesso chip. I vari blocchi logici sono collegati fra loro mediante opportune *interconnessioni programmabili*,

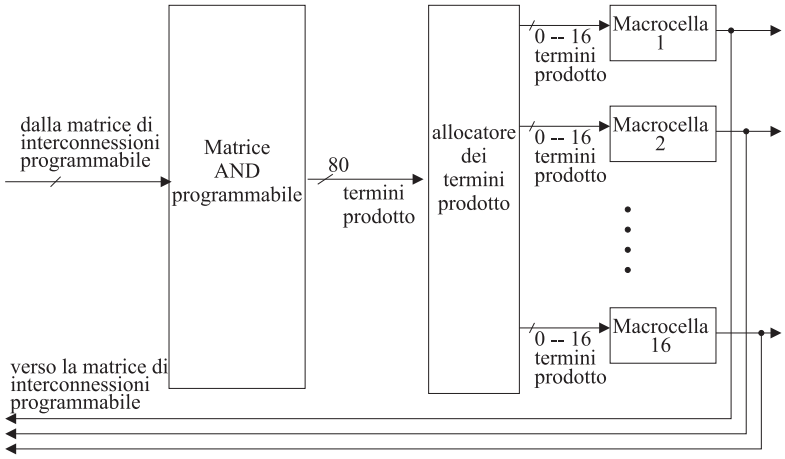

Figura 1.27: Schema semplificato del blocco logico delle CPLD della famiglia Cypress 370.

come mostra lo schema semplificato di Figura 1.26. Si parla, in questo caso di PLD Complesse o CPLD.

Le architetture delle varie CPLD attualmente disponibili, se possono tutte riportarsi allo schema di principio mostrato in Figura 1.26, differiscono sia per la struttura interna dei blocchi logici che della matrice di interconnessione. Una disamina approfondita delle molteplici varianti è ben al di la dello scopo di queste note.

Per quanto riguarda i *blocchi logici* essi sono sempre costituiti da una matrice AND programmabile e da un insieme di macrocelle di uscita. Come nelle PAL studiate nei paragrafi precedenti, ogni macrocella di un CPLD può essere programmata in modo da invertire eventualmente il segnale di uscita. Inoltre, ogni macrocella, includendo un flip-flop, può essere configurata come "combinatoriale" o "sequenziale". L'uscita della macrocella può essere inviata nuovamente in ingresso alla matrice AND in modo da realizzare un *feedback locale*. In un CPLD è inoltre possibile inviare l'uscita di una macrocella, attraverso il sistema di interconnessioni programmabili, all'ingresso del piano AND di un'altra macrocella, realizzando in questo modo un *feedback globale*. Alcuni blocchi logici includono delle *macrocelle nascoste* o *buried macrocells*, le cui uscite vengono utilizzate esclusivamente come feedback e non per pilotare i terminali di I/O.

I blocchi logici delle CPLD più avanzate consentono la *allocazione variabile* e la condivisione dei termini prodotto (*product term steering and sharing*). A mero titolo esemplificativo, in Figura 1.27 si riporta uno schema di principio del blocco logico delle CPLD della famiglia 370 della Cypress. L'uscita del piano AND, costituita da 80 termini prodotto, è inviata ad un blocco programmabile di allocazione che smista per ognuna delle 16 macrocelle di uscita, alcuni termini prodotto, in numero variabile da 0 a 16. In questo modo, alle macrocelle dedicate al calcolo di funzioni logiche "semplici" verranno inviati pochi termini prodotto, consentendo alle altre macrocelle il calcolo di funzioni logiche più complesse (*product term steering*). Inoltre, uno stesso termine prodotto può essere condiviso fra più macrocelle (*product term sharing*), come accadeva nelle PLA.

Negli ultimi anni, la capacità dei CPLD è aumentata in modo esponenziale passando da poche migliaia di porte equivalenti fino a valori superiori a 10^5

Caratteristiche	EPF10K10	EPF10K30	EPF10K50	EPF10K100
Capacità (gate equivalenti)	10 000	30 000	50 000	100 000
Numero di blocchi logici	576	1728	2888	49992
Numero di flip-flop	720	1968	3184	5392
Numero di pin di I/O	150	246	310	406

Tabella 1.3: Caratteristiche dei CPLD della famiglia Altera FLEX10K

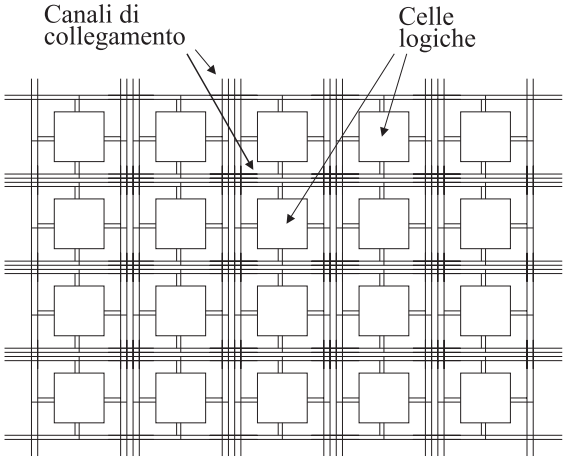

Figura 1.28: Schema di principio di una matrice di porte o *gate-array*.

porte equivalenti, con un numero di macrocelle variabile da alcune decine fino ad alcune centinaia.

A titolo di esempio, la Tabella 1.3 riporta le caratteristiche salienti delle CPLD della famiglia FLEX 10K della Altera.

Fortunatamente, accanto allo sviluppo dei CPLD (e degli FPGA che studieremo fra breve) si è assistito ad un corrispondente sviluppo dei programmi di progettazione assistita al calcolatore (CAD). In questo modo, il progettista può concentrarsi sulla definizione ad alto livello del sistema da sviluppare, lasciando ad opportuni programmi di sintesi, ottimizzazione e fitting il compito di di sfruttare al meglio le potenzialità dei dispositivi programmabili.

1.6 Matrici di porte programmabili (FPGA)

Un gate-array *programmabile a livello di maschera* è costituito da una matrice regolare di *celle logiche* tutte uguali fra loro, integrate su di uno stesso chip che prende il nome di *master*. Le varie celle logiche di un master non sono collegate fra di loro. Come mostra la Figura 1.28, ogni singolo *master* viene *personalizzato o programmato* realizzando le sole linee di metal (localizzate in opportuni *canali di collegamento* orizzontali e verticali) che interconnettono le varie celle logiche in modo da realizzare la funzione desiderata.

Rispetto alla progettazione *ex novo* di un intero circuito integrato, l'utilizzo di gate-array programmabili a livello maschera consente sia di *ridurre i costi* (in quanto i master sono tutti uguali fra loro e vengono quindi progettati una volta per tutte e prodotti in larga scala), sia di *ridurre i tempi di sviluppo* (in

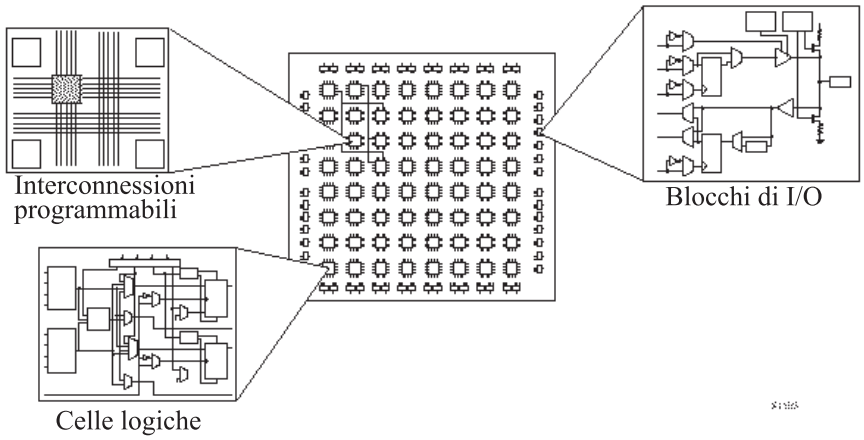

Figura 1.29: Struttura interna di un FPGA.

quanto per personalizzare un master è necessario realizzare le sole interconnessioni metalliche, mentre tutti gli altri passi di processo, quali la realizzazione delle diffusioni, dei transistori ecc., sono già stati realizzati in precedenza). Questi vantaggi sono bilanciati da un peggiore utilizzo dell'area di Silicio e da un aumento dei tempi di propagazione. Entrambi questi svantaggi sono dovuti al fatto che le celle logiche non sono ottimizzate per nessun specifica applicazione e che, inoltre, le interconnessioni sono molto più lunghe rispetto a quelle che si avrebbero in un circuito realizzato *ad hoc*, con un sensibile aumento delle capacità parassite.

Le matrici di porte programmabili "sul campo" (*Field-Programmable Gate Array* o *FPGA*) rappresentano uno sviluppo della tecnologia dei gate-array programmabili a livello maschera. In un FPGA sono "prefabbricate" non solo la matrice di porte logiche *ma anche le interconnessioni*. A differenza di un gate-array programmato a livello maschera, *le interconnessioni sono programmabili*, utilizzando tecnologie del tutto analoghe a quelle che abbiamo studiato nei paragrafi precedenti. La figura 1.29 mostra la struttura interna di un FPGA ed evidenzia la presenza di opportune celle di ingresso/uscita programmabili.

Da un punto di vista applicativo, molte caratteristiche degli FPGA e dei CPLD sono simili fra loro. In effetti, se confrontiamo le figure 1.26 e 1.29 vediamo che sia i CPLD che gli FPGA sono costituiti da interconnessioni programmabili e da opportune celle o blocchi logici. La principale differenza è legata al fatto che le celle logiche di un FPGA hanno in generale un funzionalità ridotta rispetto alla combinazione di piano AND programmabile e macrocella di un CPLD. D'altro canto il numero di celle logiche di un FPGA è molto maggiore rispetto al numero di blocchi logici di un CPLD. In sintesi, un FPGA ha un'architettura "a grana fine" rispetto all'architettura "a grana grossa" di un CPLD. Questa differenza comporta alcune differenze nelle caratteristiche di CPLD ed FPGA.

Ad esempio, la presenza di un elevato numero di celle logiche consente di rendere disponibili un numero maggiore di flip-flop in un FPGA rispetto ad un CPLD.

Un'altra differenza riguarda i tempi di propagazione.

La struttura "a grana fine" di un FPGA consente di realizzare funzioni estremamente complesse collegando opportunamente in cascata varie celle logiche,

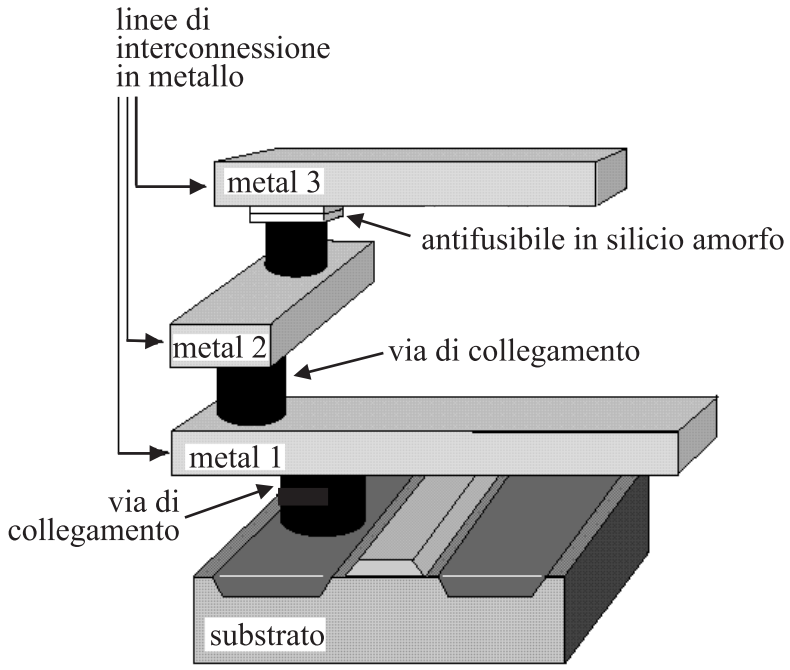

Figura 1.30: Antifusibile.

ognuna delle quali è costituita da un numero limitato di porte logiche e quindi ha un ritardo di propagazione ridotto. In definitiva, il ritardo di propagazione complessivo di un FPGA è determinato in larga misura dal numero di livelli logici necesari per realizzare una assegnata funzione e dai ritardi introdotti dalle interconnessioni programmabili presenti sul chip. Ne consegue che il ritardo di propagazione complessivo di un FPGA è molto variabile a seconda del particolare sistema da realizzare, e può essere calcolato solo al termine di tutte le fasi di sviluppo del progetto.

In un CPLD, invece, le funzioni vengono realizzate mediante logica a due passi (feedback locale) o, al più, mediante un fedback globale (un collegamento fra l'uscita di una macrocella e l'ingresso al piano AND di un'altra macrocella). Il ritardo di un blocco logico di una CPLD è peraltro maggiore rispetto a quello di una cella logica di un FPGA. Pertanto, il ritardo introdotto da una CPLD può essere valutato facilmente nelle fasi iniziali dello sviluppo di un progetto ed è poco dipente dalla complessità del progetto stesso.

1.6.1 Tecniche di programmazione.

Mentre nelle PAL e nei CPLD, come per le PLA, la tecnica di programmazione è basata sull'utilizzo di transistori *floating gate*, nelle FPGA si adoperano *antifusibili* (nel caso di circuiti programmabili una sola volta) oppure memorie di tipo SRAM (nel caso di FPGA riprogrammabili).

La Figura 1.30 mostra lo schema di un collegamento programmabile mediante antifusibile. Dalla Figura si nota che i contatti fra due livelli di metallo (in questo caso particolare, il secondo ed il terzo livello di metallo) comprendono un sottile strato di silicio amorfo, non conduttore. L'applicazione di una opportuna differenza di potenziale ai capi del contatto consente di rompere il sottile strato

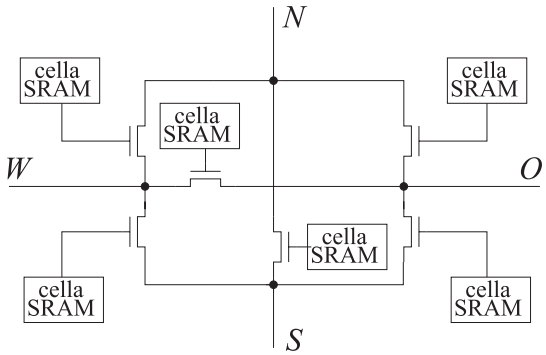

Figura 1.31: Utilizzo di celle di memoria SRAM e porte di trasmissione NMOS per realizzare interconnessioni programmabili.

dispositivo di collegamento	tecnologia	riprogrammabile?	volatile?	applicazioni
fusibile	bipolare	no	no	PAL
EEPROM	CMOS	si	no	PAL, CPLD
antifusibile	CMOS	no	no	FPGA
SRAM	CMOS	si	si	FPGA

Tabella 1.4: Caratteristiche delle differenti tecniche di programmazione utilizzate nei dispositivi logici programmabili.

isolante, realizzando un contatto elettrico a bassa resistenza fra le due linee di interconnessione in metal2 ed in metal3.

La Figura 1.31 mostra l'utilizzo di celle di memoria SRAM per realizzare delle interconnessioni programmabili. Ad ogni cella di memoria è collegata la gate di un transistore NMOS, che opera come porta di trasmissione. Utilizzando sei locazioni di memoria ed altrettante porte di trasmissione nel circuito di figura 1.31 si possono effettuare tutti i possibili collegamenti fra i quattro terminali N,W,E ed S. Il vantaggio di disporre di un sistema riprogrammabile viene pagato con una maggiore occupazione di area e con maggiori ritardi di propagazione rispetto alla tecnologia basata su antifusibili. Le memorie SRAM sono inoltre volatili, per cui la programmazione viene "persa" dopo aver scollegato l'alimentazione alla FPGA. In questi circuiti la mappa dei collegamenti viene in pratica memorizzata in una piccola ROM, esterna alla FPGA. Nei primi istanti dopo l'applicazione della tensione di alimentazione, il contenuto della ROM viene trasferito nella SRAM interna programmando così l'FPGA.

La Tabella 1.4 riassume le caratteristiche delle differenti tecniche di programmazione utilizzate nei dispositivi logici programmabili.

1.6.2 FPGA commerciali

Come accennato in precedenza, è possibile molto spesso utilizzare indifferentemente un FPGA o un CPLD per realizzare una stessa applicazione. In alcuni casi si utilizza la stessa sigla FPGA per riferirsi indifferentemente all'una o all'altra classe di dispositivi programmabili.

Per avere un esempio dell'architettura di FPGA disponibili commercialmen-

te, la Figura 1.32 mostra lo schema della cella logica degli FPGA della famiglia XC4000 della XILINX. La cella logica è basata sull'utilizzo di alcuni multiplexer e di tre piccole memorie SRAM, ognuna dei quali è programmabile in modo tale da realizzare una qualsiasi funzione logica di quattro ingressi.

La Tabella 1.5 riporta le caratteristiche salienti di questa famiglia di FPGA .

Caratteristiche	XC4003	XC4010	XC4025	XC4085
Capacità (gate equivalenti)	3 000	10 000	25 000	85 000
Dimensioni matrice blocchi logici	10×10	20×20	32×32	56×56
Numero di flip-flop	360	1120	2560	7168
Numero di pin di I/O	80	160	256	448

Tabella 1.5: Caratteristiche degli FPGA della famiglia XILINX XC4000.

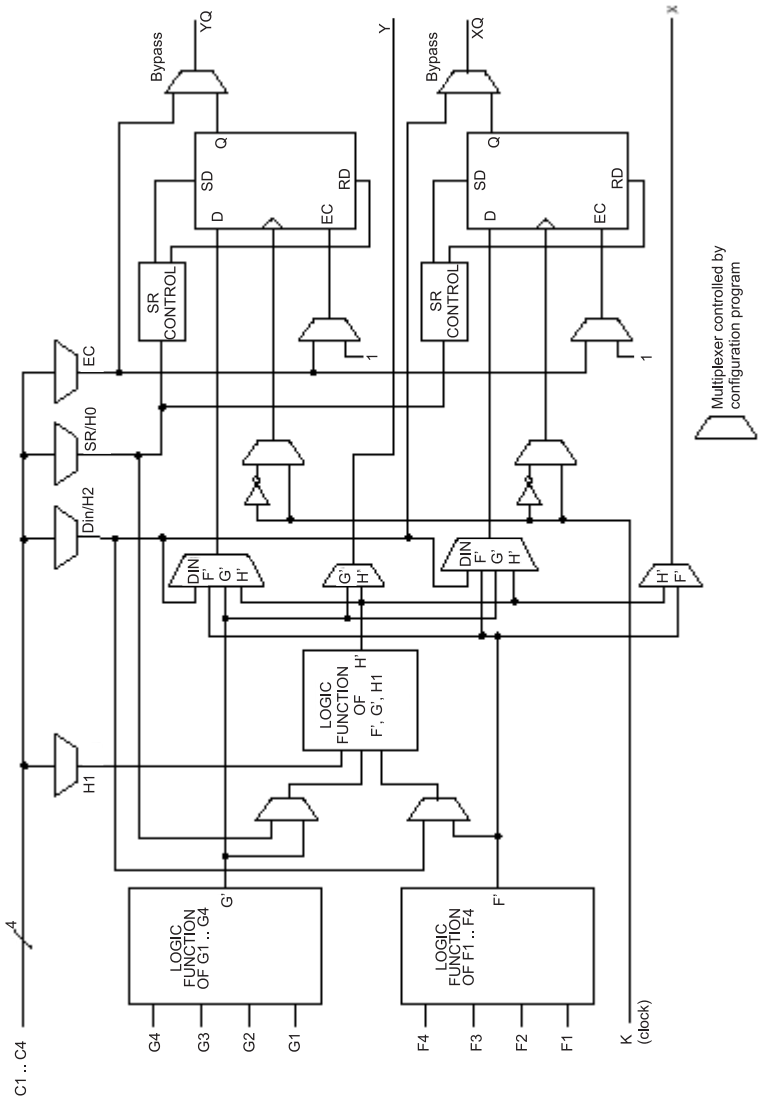

Figura 1.32: Cella logica degli FPGA della famiglia XC4000.

Bibliografia

[1] D. Pellerin, M. Holley, *"Practical design using programmable logic"*, Prentice-Hall, 1991.

[2] J. F. Wakerly, *"Digital design, Principles and practices"*, Prentice-Hall, 2nd edition, 1994.

[3] K. Skahill, *"VHDL for programmable logic"*, Addison-Wesley, 1996.

[4] M. J. S. Smith, *"Application-specific integrated circuits"*, Addison-Wesley, 1997.

[5] J. Rose, A. El Gamal, A. Sangiovanni-Vincentelli, "Architecture of field-programmable gate arrays", *Proceedings of the IEEE*, vol. 81, n. 7, pag. 1013-1028 1993.

[6] S. Trimberger, "A reprogrammable gate array and applications", *Proceedings of the IEEE*, vol. 81, n. 7, pag. 1030-1041 1993.

[7] J. Greene, E. Hamdy, S. Beal, "Antifuse field-programmable gate arrays", *Proceedings of the IEEE*, vol. 81, n. 7, pag. 1042-1056 1993.

[8] S. Brown, J. Rose, "FPGA and CPLD architectures: a tutorial", *IEEE design and test of computers*, pag. 42-57, summer 1996.

[9] Produttori di CPLD ed FPGA:
Actel: http://www.actel.com
Altera: http://www.altera.com
Atmel: http://www.atmel.com
Amd: http://www.amd.com
Xilinx: http://www.xilinx.com
Philips: http://www.cool-pld.com