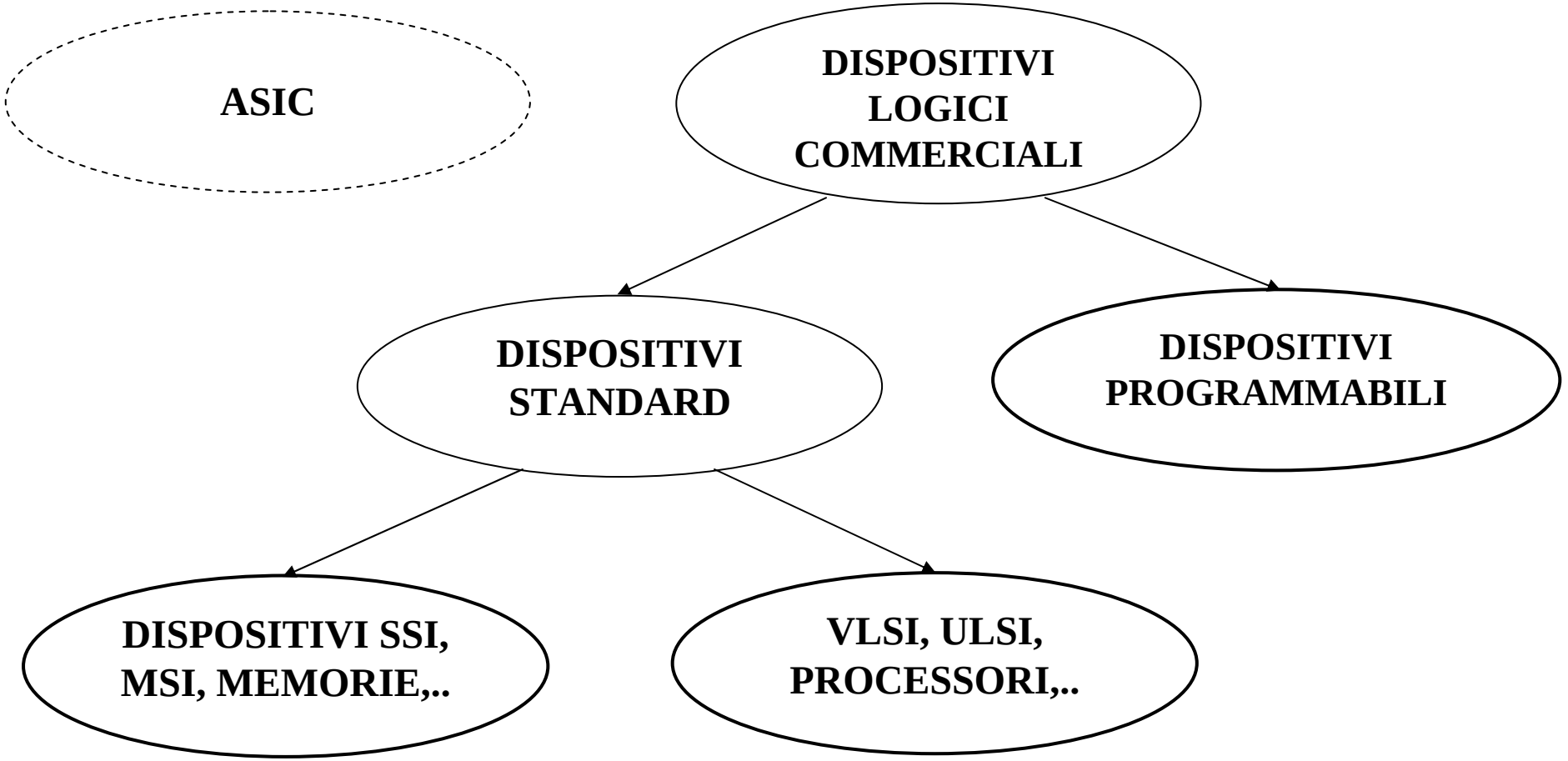
Dispositivi programmabili

Dispositivi programmabili

- **Dispositivi standard, programmabili, ASIC**



Dispositivi programmabili

- **Dispositivi logici**

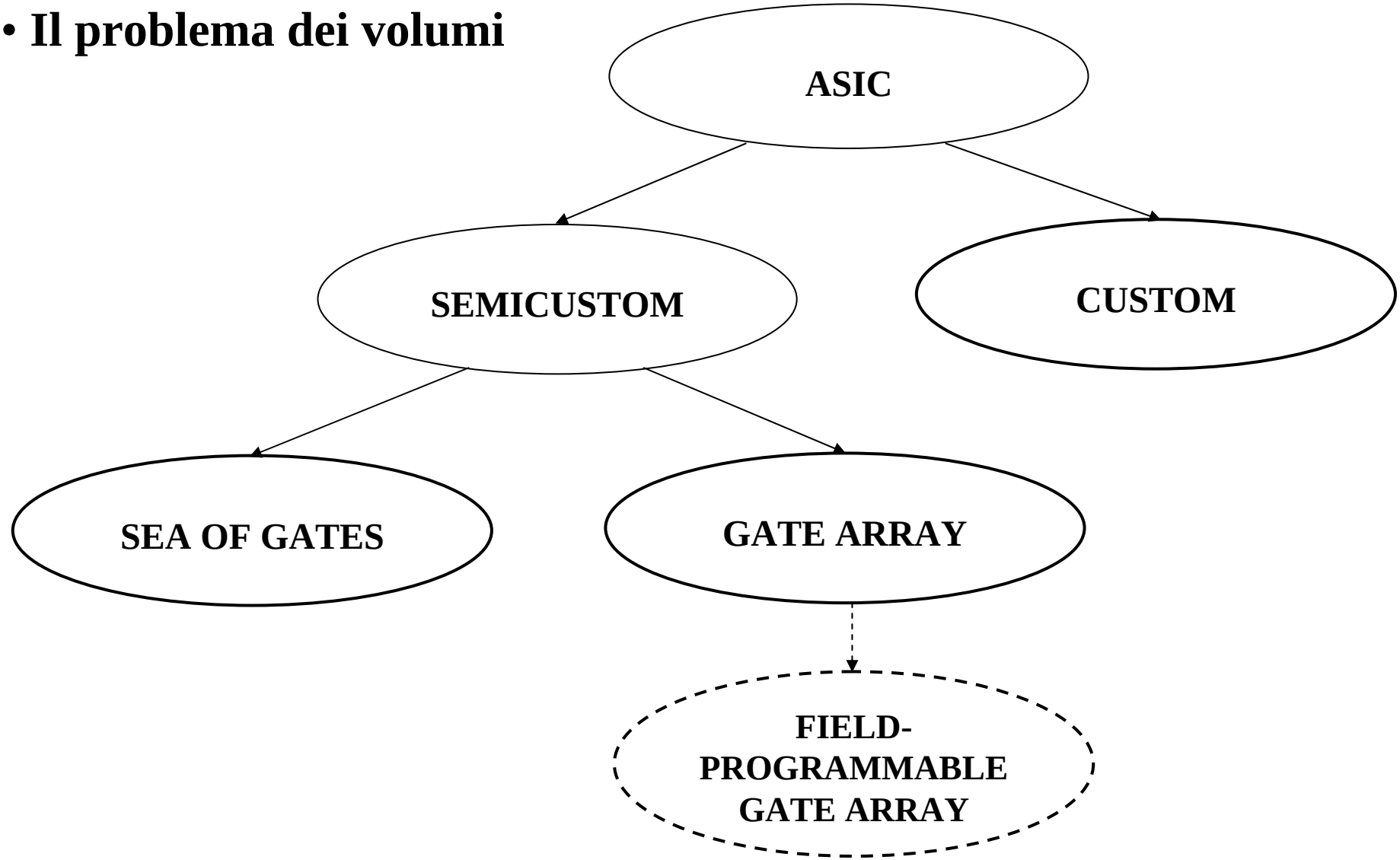
- **DISPOSITIVI LOGICI STANDARD (A LOGICA CABLATA)**
  - **Famiglie logiche (Es. 74HC, 74AS,…)**
  - **Basso costo, ampia disponibilità, elevate prestazioni**
  - **Assemblabili in schede ⟶ scarsa compattezza, elevati costi di industrializzazione (EMC,..)**

- **DISPOSITIVI LOGICI STANDARD (A LOGICA PROGRAMMABILE)**
  - **Microprocessori, microcontrollori, DSP, processori dedicati, periferiche, …**
  - **Basso costo, ampia disponibilità, elevata versatilità, basse prestazioni**
  - **Assemblabili in schede ⟶ buona compattezza, costi di industrializzazione medi**

- **DISPOSITIVI LOGICI PROGRAMMABILI**
  - **SPLD, CPLD, FPGA**
  - **Medio costo, discreta disponibilità, discreta capacità, elevate prestazioni**
  - **Assemblabili in schede ⟶ ottima compattezza, bassi costi di industrializzazione**

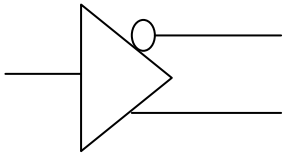
Dispositivi programmabili

- **ASIC (Application Specific Integrated Circuit)**
- **Il problema dei volumi**

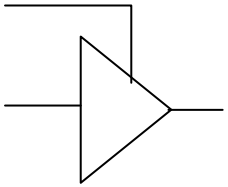


Dispositivi programmabili

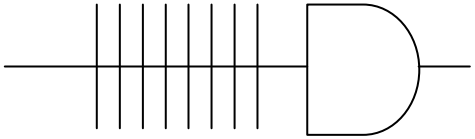
- **Dispositivi programmabili: simbologia**



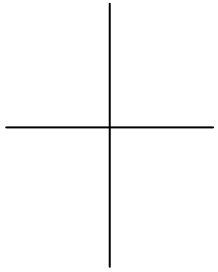
Buffer d’ingresso



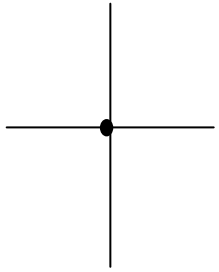
Buffer d’uscita



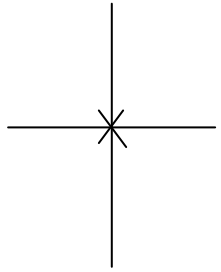
AND a 8 ingressi



Connessione aperta



Connessione chiusa



Connessione programmabile

Dispositivi programmabili

- **Dispositivi programmabili: origini e uso attuale**

- **I dispositivi programmabili nascono per:**
  - **compattare le logiche standard (SPLD)**
  - **essere utilizzati come prototipi di ASIC (FPGA)**

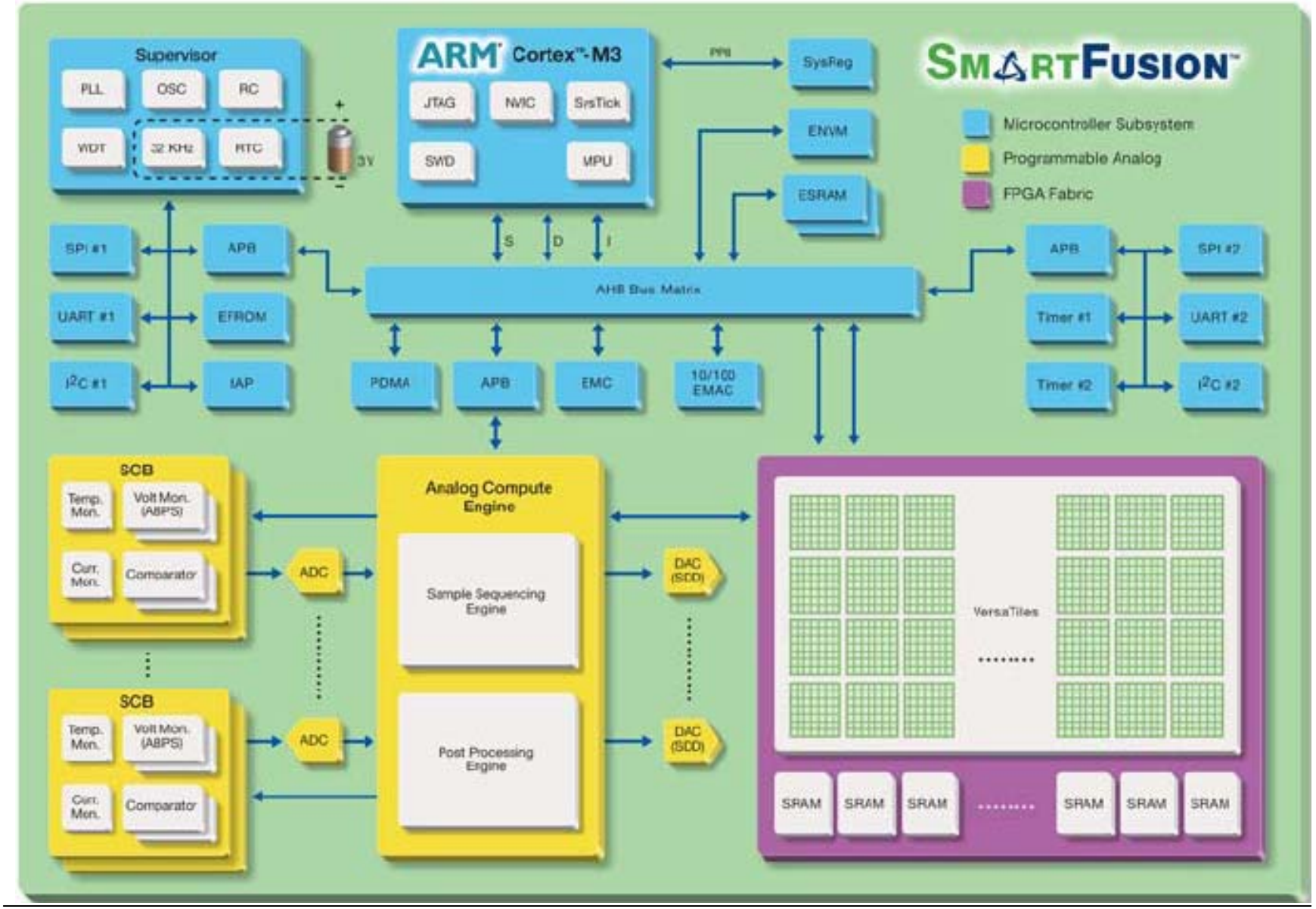
- **Nascono come dispositivi diversi**

- **SPLD**
  - **Veloci (competitive con le famiglie logiche)**
  - **Non volatili (PROM, EPROM, EEPROM, FLASH)**
  - **Evolvono nelle CPLD (usate per periferiche programmabili ad alta velocità)**

- **FPGA**
  - **Alta capacità e elevata flessibilità, anche se prototipali e volatili (RAM)**
  - **Download del programma da una memoria**
  - **Oggi utilizzate per "Soft-processors" in applicazioni DSP**
  - **Mixed signal FPGA (integrano ADC e DAC, sono System-On-a-Chip)**
  - **Possono integrare processori evoluti (hard-processors) e altri circuiti**

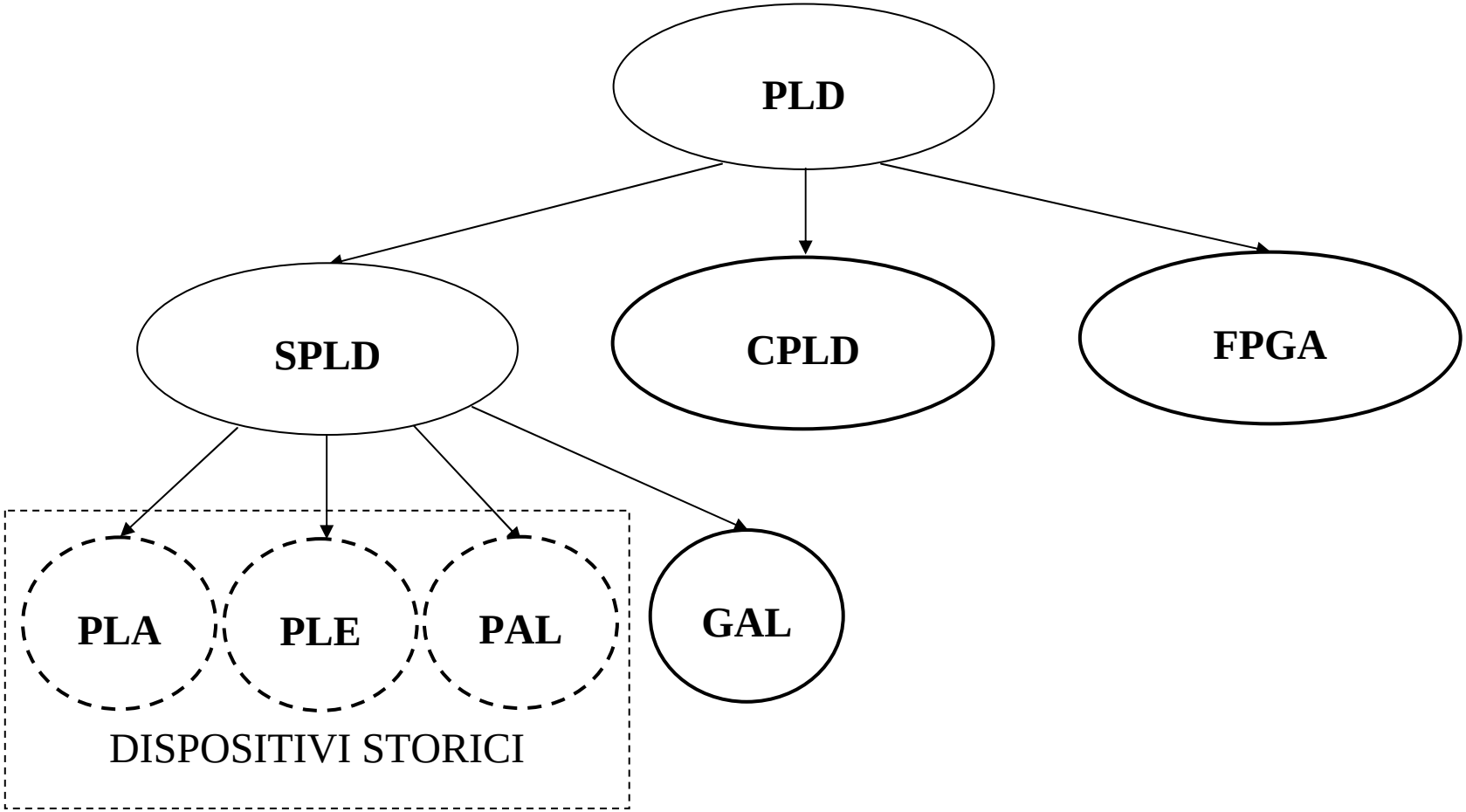
Dispositivi programmabili

- **Dispositivi programmabili: System-On-a-Chip**



Dispositivi programmabili

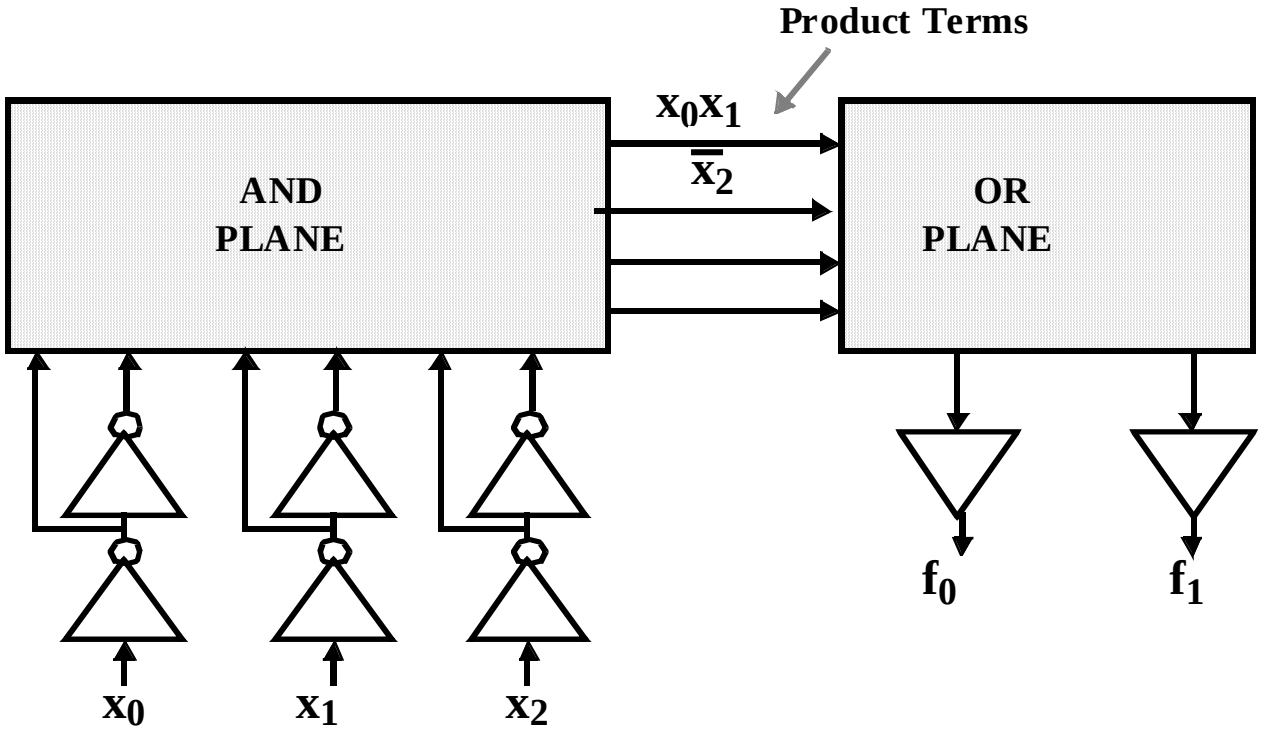
- **Dispositivi programmabili: classificazione**



Dispositivi programmabili

- **Dispositivi programmabili: SPLD (Simple PLD)**

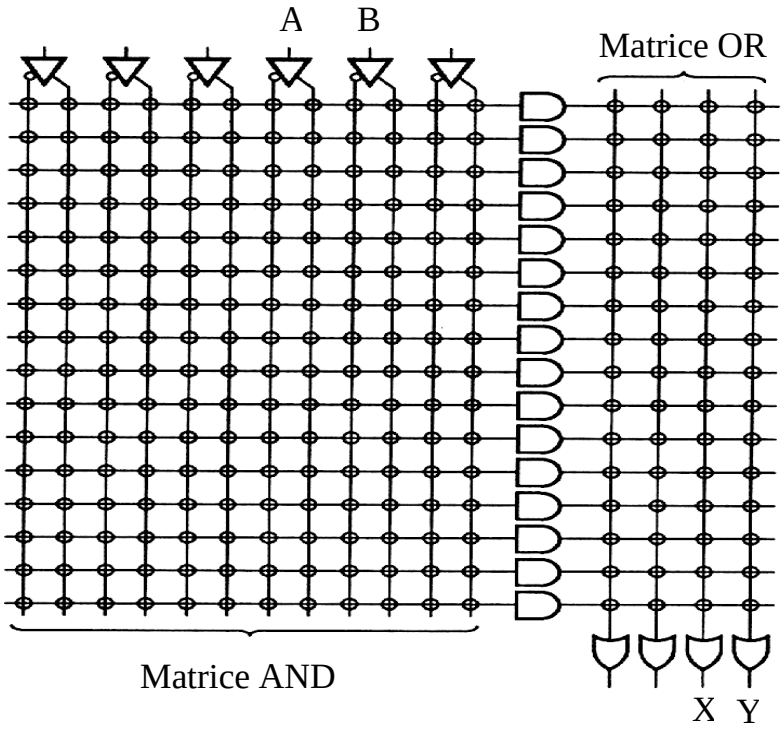
- **matrice di "AND" seguita da una matrice di "OR"**
  - **uscita = funzione descrivibile come somma di prodotti (SOP = Sum of products)**
  - **linguaggio di programmazione semplice (tipo tabella della verità)**
  - **tempo di propagazione da ingresso a uscita indipendente dal tipo di funzione**



Dispositivi programmabili

- **SPLD: Programmable Logic Array (PLA)**

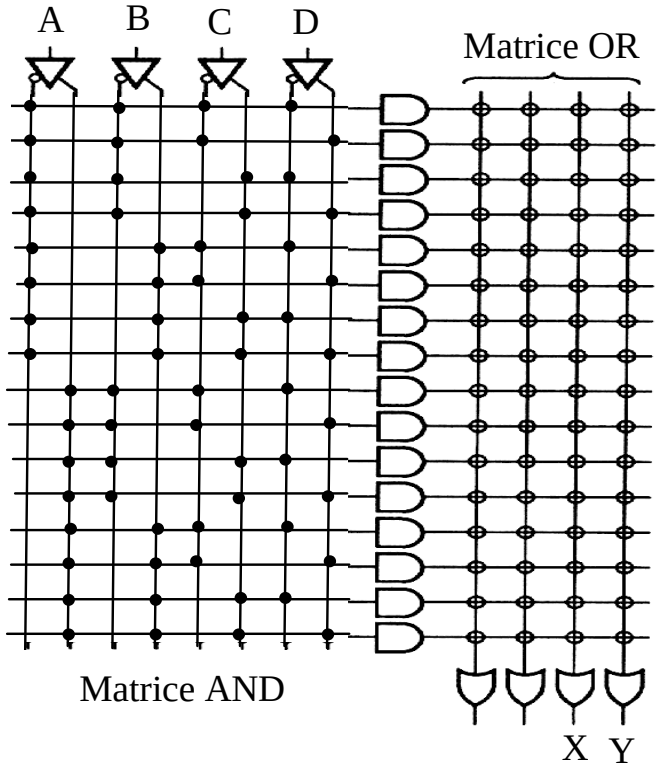
- **PLA (N ingressi, K uscite, ciascuna intesa come Sum-Of-Products**
  - **Matrice di M porte AND a 2N ingressi completamente programmabile**
  - **Matrice di K porte OR a M ingressi completamente programmabili**
  - **K funzioni logiche a N ingressi indipendenti costituite da M (max) minterm**
  - **M indica la capacità della PLA, Numero "fuses" = 2N·M + M·K = (2N+K)M**



Architettura PLA

Dispositivi programmabili

- **SPLD: Programmable Logic Element (PLE)**
- **PLE (N ingressi, K uscite, ciascuna intesa come Sum-Of-Products**
  - **Matrice di 2^N porte AND a 2N ingressi fissa (SOP "canonica")**
  - **Matrice di K porte OR a 2^N ingressi completamente programmabili**
  - **Architettura delle funzioni realizzate mediante Look-up Table, ossia memorie configurate in lettura (indirizzi = ingressi, dati = uscite)**
  - **Numero "fuses" = $2^N \cdot K$ (K funzioni logiche indipendenti del tutto generali)**

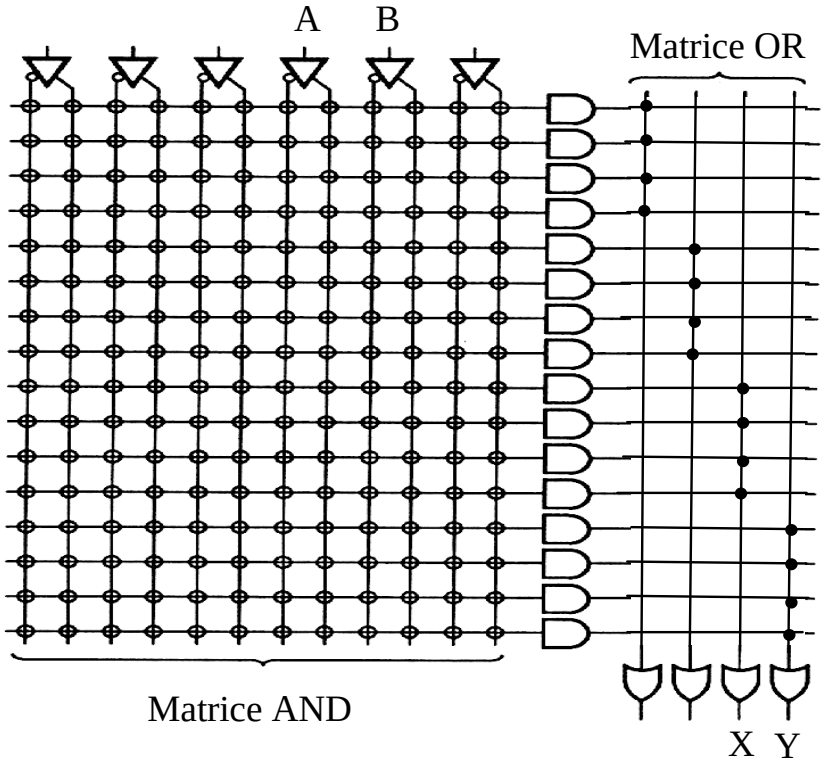


Architettura PLE

Dispositivi programmabili

- **SPLD: Programmable Array Logic (PAL)**

- **PAL (N ingressi, K uscite, ciascuna intesa come Sum-Of-Products**
  - **Matrice di M porte AND a 2N ingressi completamente programmabile**
  - **Matrice di K porte OR a L ingressi fissa (L<<M)**
  - **K funzioni logiche a N ingressi indipendenti costituite da L (max) minterm**
  - **Numero "fuses" = 2N·M**

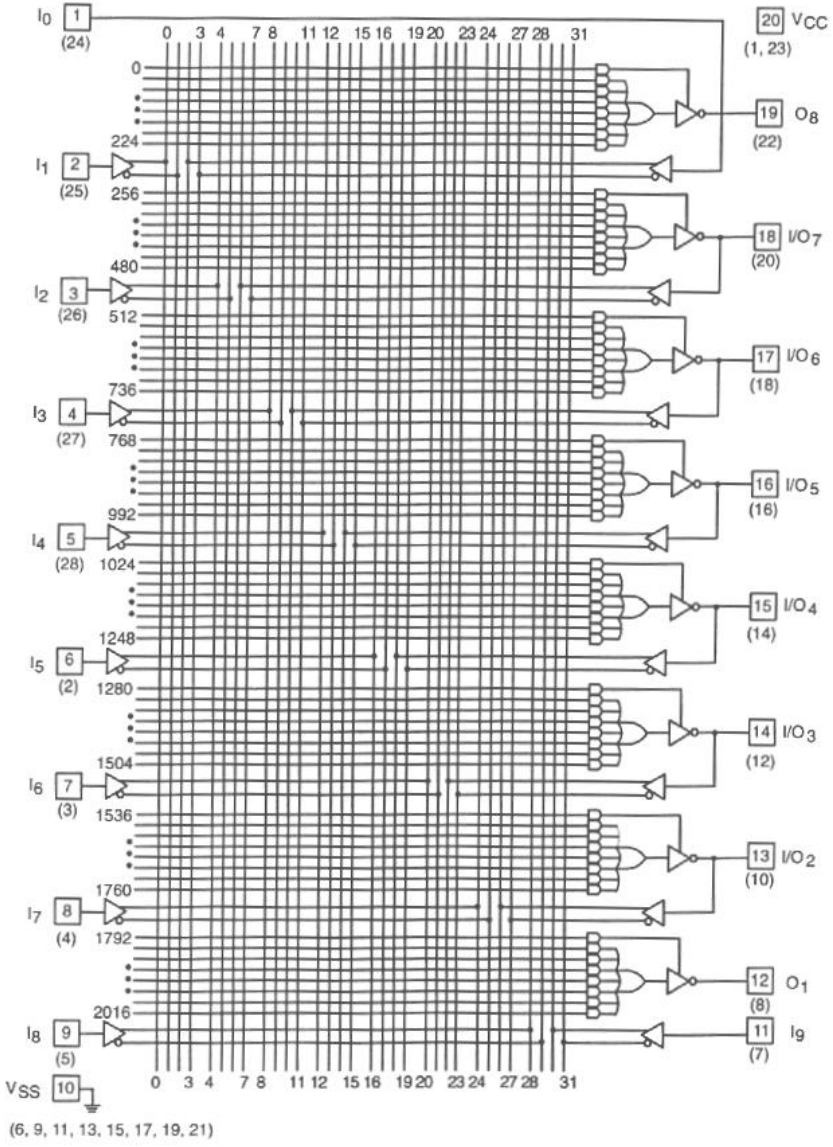


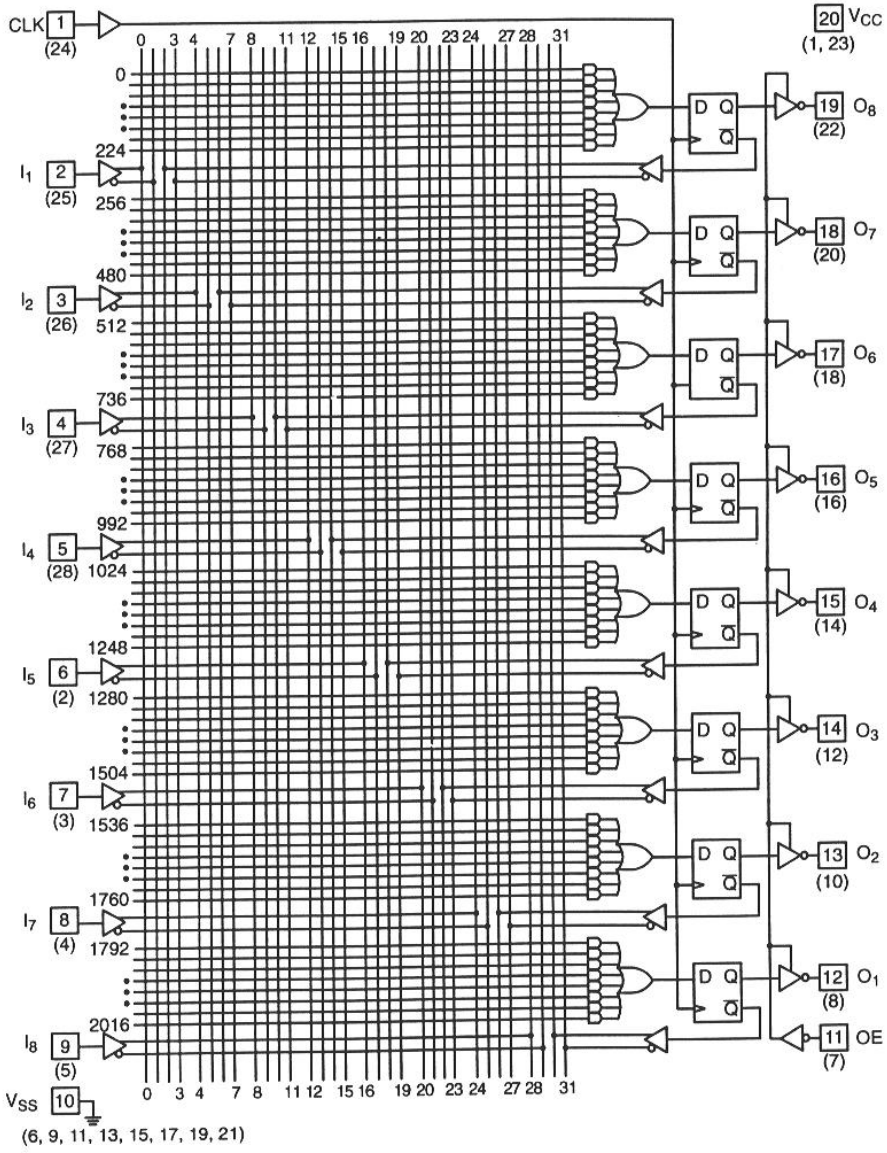
Architettura PAL

Tipo PLD	Fuses	Limiti
PLA	(2N+K)M	Dipende da M
PLE	$2^N K$	Ottima per pochi ingressi
PAL	2NM	Un minterm per ogni uscita

Dispositivi programmabili

- **SPLD: PAL 16L8 (left) 16R8 (right)**



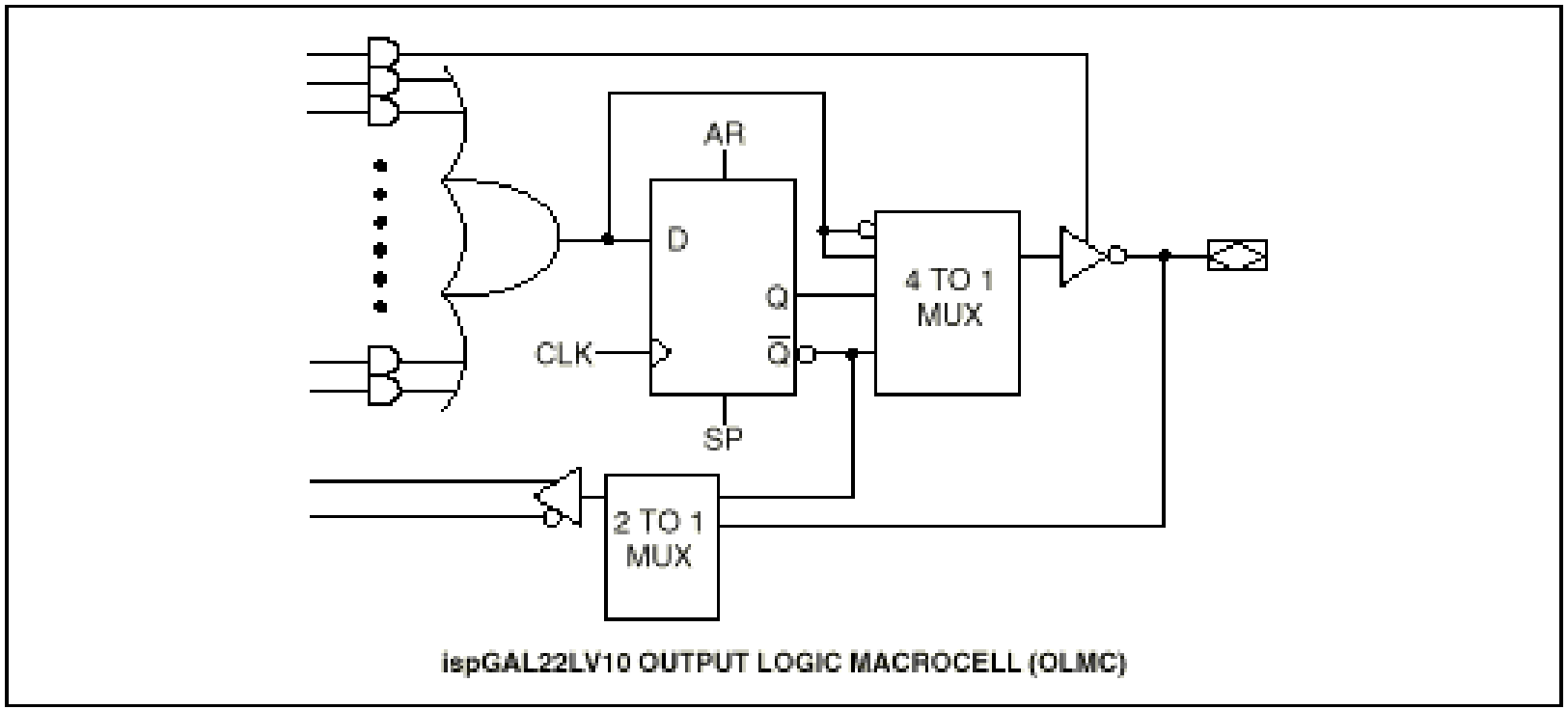


Dispositivi programmabili

- **Limiti delle SPLD**

- **Consumi, non riprogrammabile, scarsa flessibilità**
  - **in uno stesso dispositivo posso avere tutte funzioni simili (Esempio: attive basse, senza flip-flop come nella 16L8) invece le funzioni sono diverse tra loro**

- **Soluzione: mantenere l'architettura PAL ma personalizzare ogni uscita grazie all'introduzione delle macrocelle di uscita**

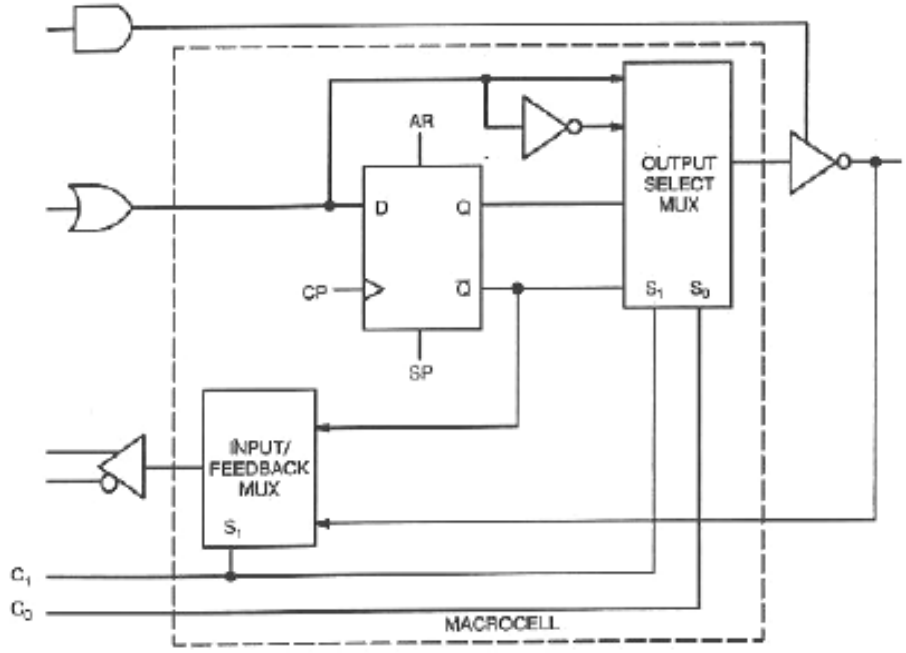


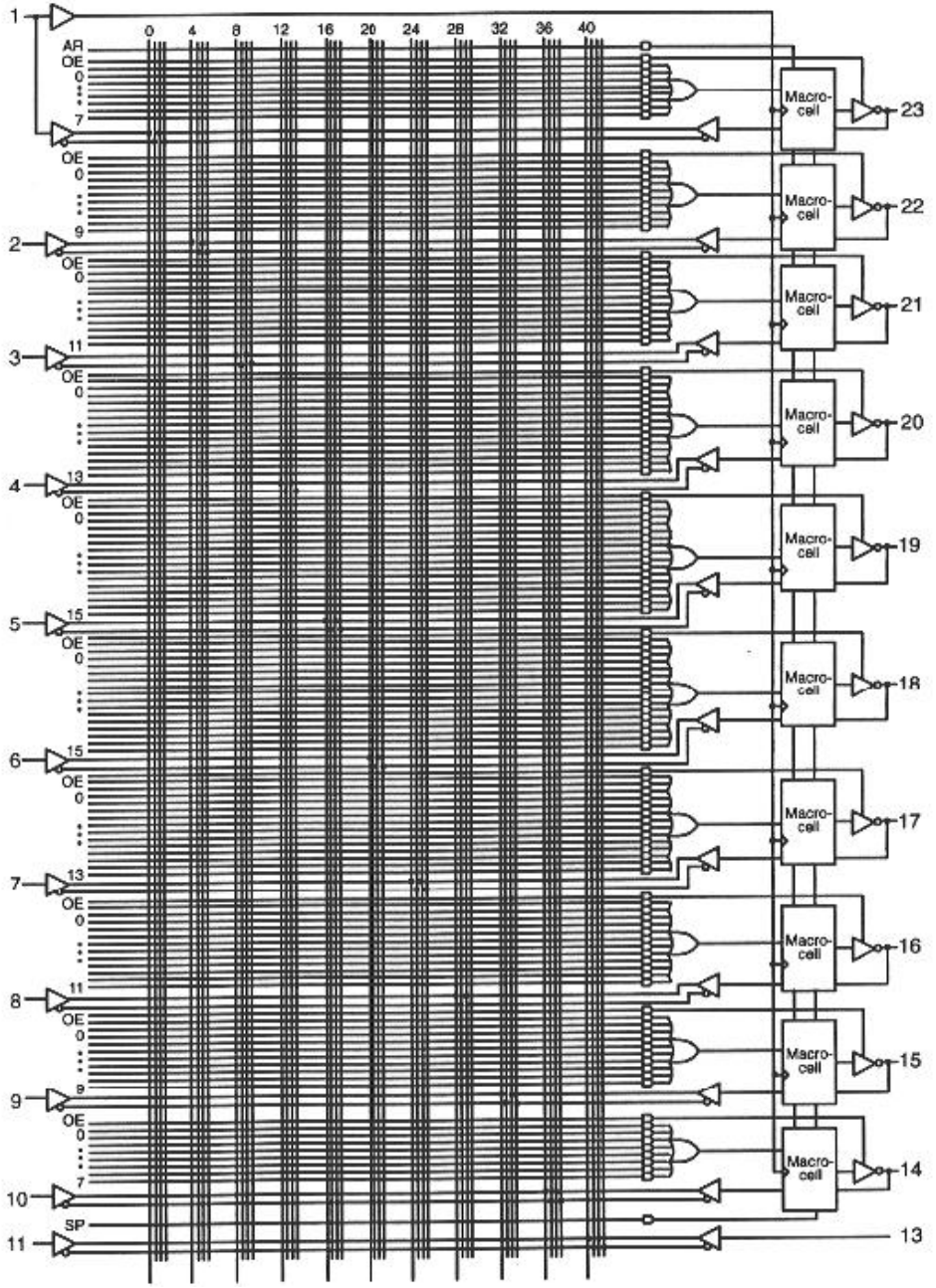
ispGAL22LV10 OUTPUT LOGIC MACROCELL (OLMC)

Dispositivi programmabili

- **Generic Logic Array (GAL)**
  Es. GAL22V10

- **Tecnologia CMOS EEPROM**
- **Ideale per macchine sincrone**
- **FAN-IN porte OR variabile**
- **Modello temporale “ideale”**

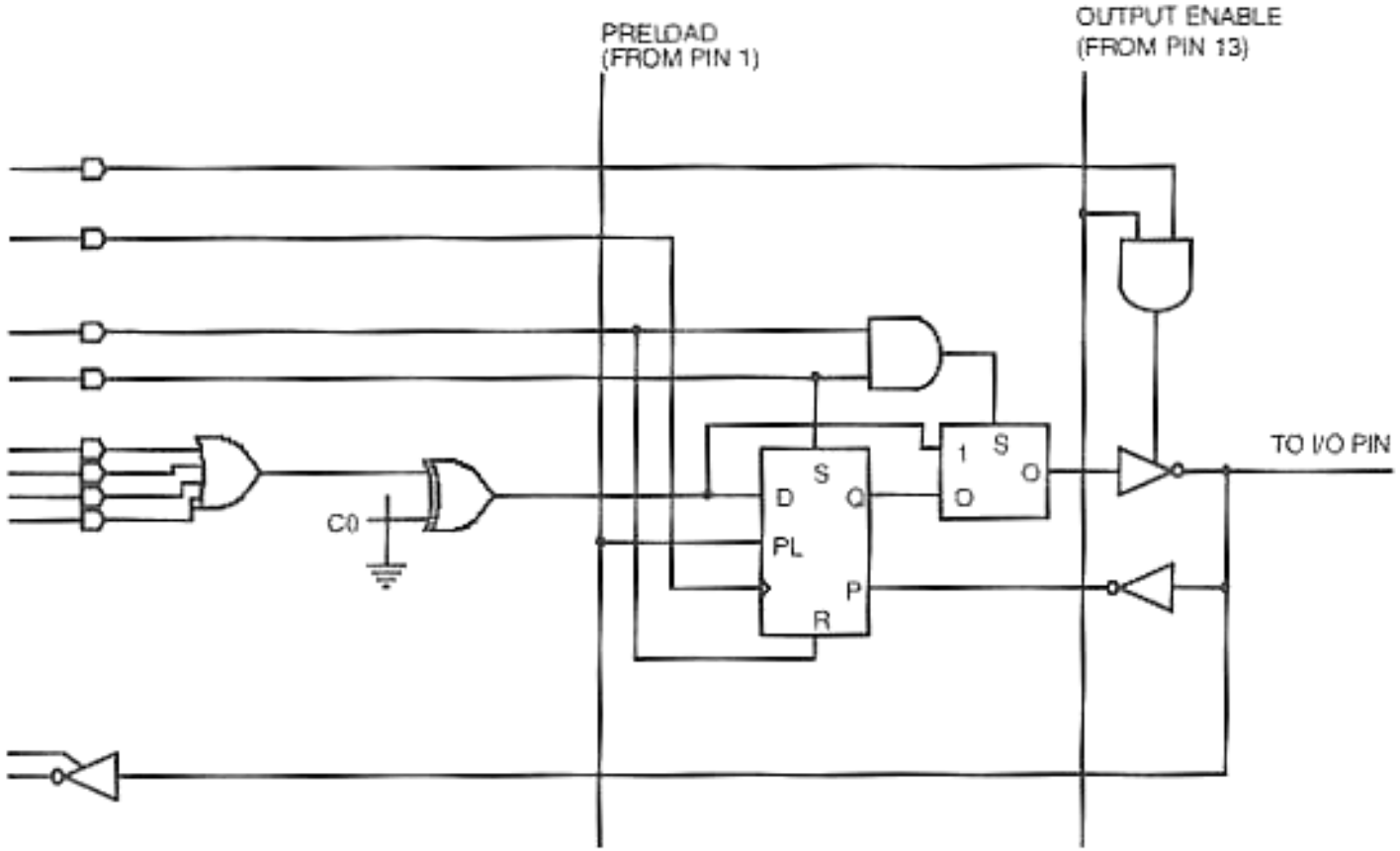




Dispositivi programmabili

- **GAL22RA10, macrocelle a confronto**

- **Macrocella più versatile (macchine sincrone e asincrone)**
- **Poco competitiva rispetto a FPGA**



Dispositivi programmabili

- **GAL22V10, parametri temporali**

- **Pochi parametri caratteristici indipendenti dalla programmazione**
  - **tpd Ritardo tra il fronte di un ingresso e il fronte di un'uscita combinatoria**
  - **ts/th Tempo in cui i dati devono essere stabili prima/dopo il fronte del clock**
  - **tco Ritardo tra il fronte del clock e il fronte di un'uscita registrata**
  - **tco2 Ritardo tra il clock e un'uscita combinatoria che utilizza come ingresso un segnale registrato**
  - **tscs Minimo periodo di clock (Si pensa a un segnale registrato che rientra e prosegue verso un'uscita registrata 1/tscs = fmax)**

Parametri	Descrizione	Min.	Max.
t_{pd}	Ritardo di propagazione		4 ns
t_s	Tempo di set-up	2.5 ns	
t_h	Tempo di hold	0	
t_{co}	Ritardo clock-to-output		3.5 ns
t_{co2}	Clock-to-output (attraverso array)		7 ns
t_{scs}	Ritardo system ck-to system ck		5.5 ns

Dispositivi programmabili

- **PAL/GAL: tecniche di programmazione**

- **Il programma viene scritto in un file ASCII composto da due parti:**
  - **Assegnazione di nomi ai piedini**
  - **Descrizione delle equazioni logiche**

  –I compilatori (PALASM, ABEL, LC9000, OPAL,…) generano:
  - **Un file JEDEC con la mappa dei fusibili (“1”<-> fusibile interrotto)**
  - **Un file di documentazione**

- **Dato il file JEDEC è sempre possibile ricostruire il file sorgente a meno delle assegnazioni dei simboli.**

- **Uno stesso file sorgente può dare origine a più file JEDEC**

Dispositivi programmabili

- **PAL/GAL: tecniche di programmazione**
- **Il programma scritto in un file ASCII è composto da due parti:**
  - **Assegnazione di nomi ai piedini**

```
CHIP prova1 gal22V10
CK   IN1   IN2    IN3    IN4   IN5   IN6 nc   nc   nc   nc   gnd
nc   OUT1 /OUT2 /OUT3 OUT4 nc    nc   nc   nc   nc   nc   Vcc
```

  - **Descrizione delle equazioni logiche che legano le uscite agli ingressi, compresa l'abilitazione dell'eventuale flip-flop e buffer 3-state (il buffer 3-state consente la bidirezionalità del piedino)**

  - **Sintassi (Es. OPALjr): / NOT (! NOT) & AND (* AND) + OR**

OUT1 = IN1&/IN2 -- Uscita Sum of Product combinatoria attiva alta
OUT2 = /IN1&IN2 -- Uscita Sum of Product combinatoria attiva bassa
OUT3.D = IN3+IN4 -- Uscita Sum of Product registrata attiva bassa
OUT4.D = IN1+/OUT4 -- Uscita Sum of Product registrata attiva alta
OUT1.OE = Vcc -- Uscita OUT1 sempre attiva
OUT2.OE = IN5 -- Uscita OUT2 attiva se IN5=1, altrimenti in three state
AR = IN6 -- Reset Asincrono pilotato da IN6

Dispositivi programmabili

- **PAL/GAL: linguaggio booleano di programmazione**

- **Sintassi (Es. OPALjr)**
- **Esempio: realizzare una mappa di memoria e un selettore:**
  - **CSRAM, attivo basso, 2000h-7FFFh**
  - **CSEPROM, attivo basso, C000h-FFFFh**
  - **CSDISP, attivo basso, 8000h-9FFFh**
  - **OUTX, attivo alto, OUTX=IN1 se SEL1 altrimenti OUTX=IN2**
- **NOTA: la 16L8 impone tutte le uscite attive basse**

```
CHIP prova1 pal16l8
A15 A14  A13    SEL1     IN2     IN1  DIS  nc  nc  gnd
nc /OUTX /CSRAM /CSEPROM /CSDISP nc   nc   nc  nc  vcc

EQUATIONS
OUTX = /SEL1&/IN2 + SEL1&/IN1
CSRAM = /A15&/A14&A13 + /A15&A14
CSEPROM = A15&A14
CSDISP = A15&/A14&/A13
OUTX.oe = /DIS                  CSRAM.oe = Vcc
CSEPROM.oe = Vcc                CSDISP.oe = Vcc
```

Dispositivi programmabili

- **PAL/GAL: linguaggio booleano di programmazione (File JEDEC)**

```
PAL16L8,  EQN2JED - Boolean Equations to JEDEC file assembler *
QF2048*QP20*F0*  <------- codice del chip
|------------------------------------------- a14
||------------------------------------------ /a14
|||----------------------------------------- a15
||||---------------------------------------- /a15
||||  |--------------------------------------- a13
||||  ||-------------------------------------- /a13
||||  ||    |----------------------------------- sel1
||||  ||    ||---------------------------------- /sel1
||||  ||    ||     |------------------------------ /in2
||||  ||    ||     |     |------------------------- /in1
L1024||    ||     |     |     |-------------------- /dis
1111 1111 1111 1111 1111 1111 1111 1111   /csdisp.oe=Vcc
1001 1011 1111 1111 1111 1111 1111 1111*  /csdisp=/a14&a15&/a13
L1280
1111 1111 1111 1111 1111 1111 1111 1111   /cseprom.oe=Vcc
0101 1111 1111 1111 1111 1111 1111 1111*  /cseprom=a14&a15
L1536
1111 1111 1111 1111 1111 1111 1111 1111   /csram.oe=Vcc
1010 0111 1111 1111 1111 1111 1111 1111   /csram=/a14&/a15&a13+a14&/a15
0110 1111 1111 1111 1111 1111 1111 1111*
L1792
1111 1111 1111 1111 1111 1011 1111 1111   /outx.oe=/dis
1111 1111 1011 1011 1111 1111 1111 1111   /outx=/sel1&/in2+sel1&/in1
1111 1111 0111 1111 1011 1111 1111 1111*
C2745*
```

Dispositivi programmabili

- **PAL/GAL: Dal file JEDEC al file sorgente**

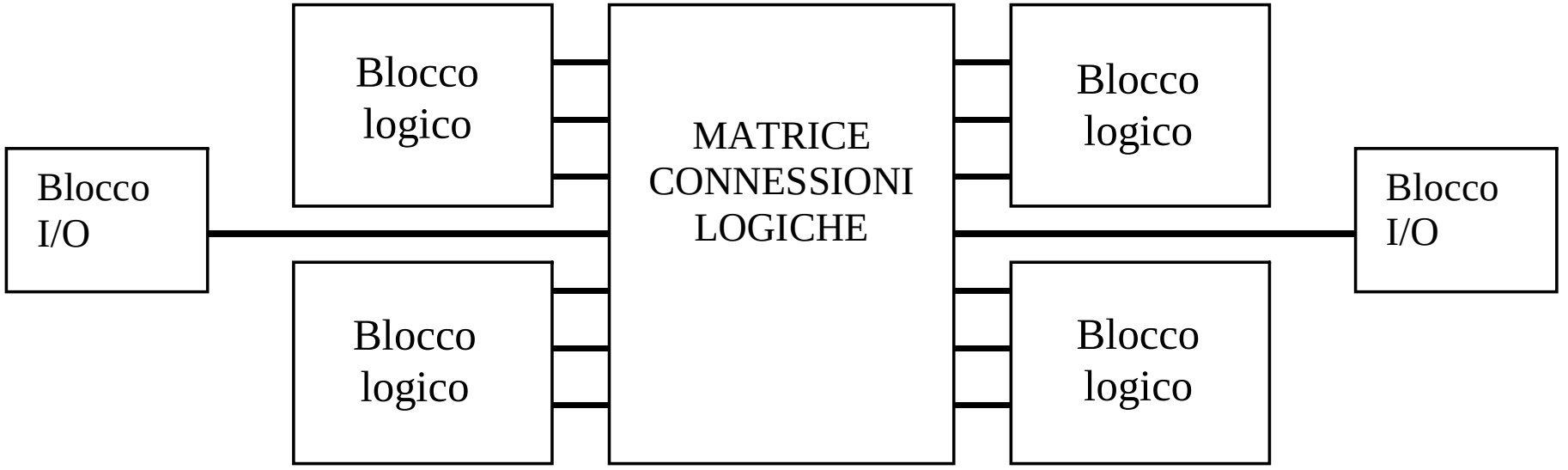
```
; JED2EQN -- JEDEC file to Boolean Equations disassembler (Version V002)
; Copyright (R) National Semiconductor Corporation 1990,1991
; Disassembled from prova1.jed. Date: 1-18-96

chip prova1 PAL16L8

i1  i2  i3  i4  i5  i6  i7  nc  nc  GND
nc  o12 o13 o14 o15 nc  nc  nc  nc  Vcc

equations

/o15 = /i2 & i1 & /i3
o15.oe = Vcc

/o14 = i2 & i1
o14.oe = Vcc

/o13 = /i2 & /i1 & i3
    + i2 & /i1
o13.oe = Vcc

/o12 = /i4 & /i5
    + i4 & /i6
o12.oe = /i7
```

Da un sorgente è possibile generare tanti file JEDEC
Da un file JEDEC è possibile generare solo un sorgente
Ovviamente il simbolico è perso

Dispositivi programmabili

- **Dalle GAL alle CPLD (Complex PLD)**

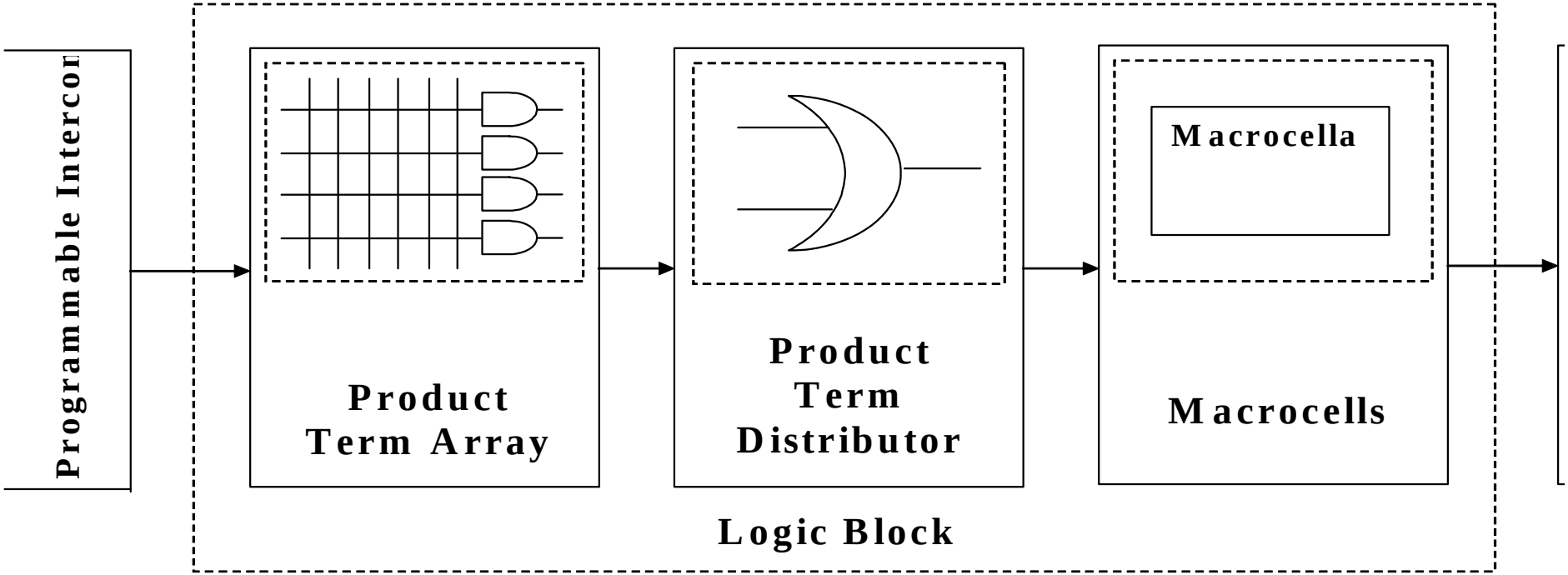
- **Integra n blocchi (blocchi logici) simili a una GAL e una matrice di interconnessione, svincolando la macrocella dall'uscita**
  - **– uscita = funzione descrivibile come connessione tra somma di prodotti con possibilità di flip-flop**
  - **– tempo di propagazione da ingresso a uscita ancora semplice da modellizzare (non dipende dal tipo di funzione ma dal numero di volte per cui si passa dalla matrice)**



Dispositivi programmabili

- **CPLD: il blocco logico**

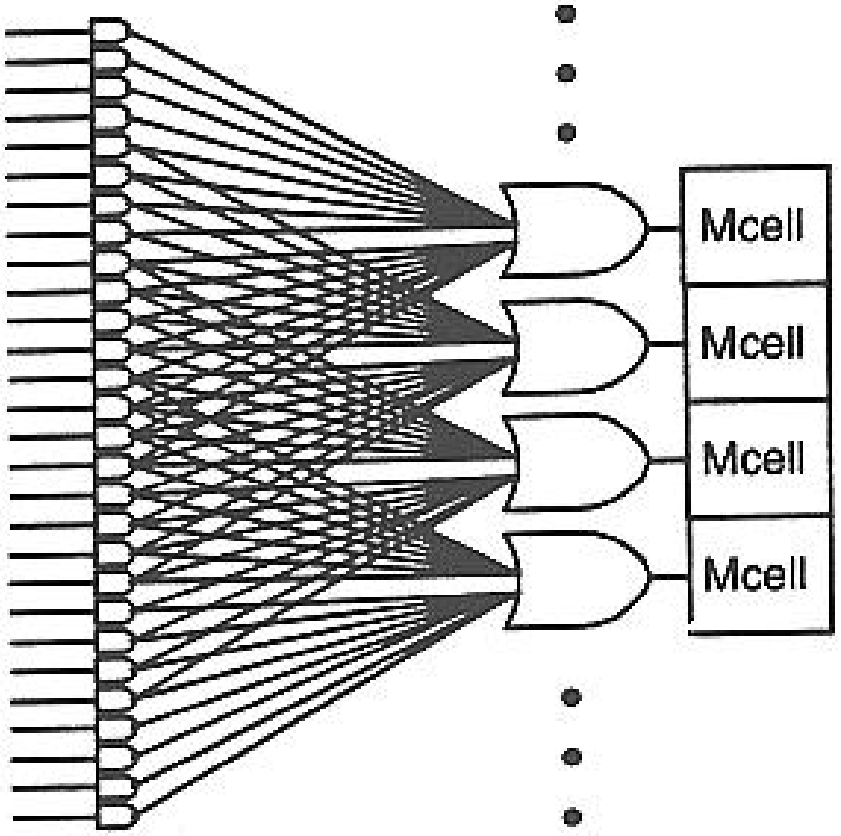
- **Product Term Array = matrice di AND**
- **Product Term Distributor = matrice di OR**
- **La macrocella può essere di I/O o “buried”**



Dispositivi programmabili

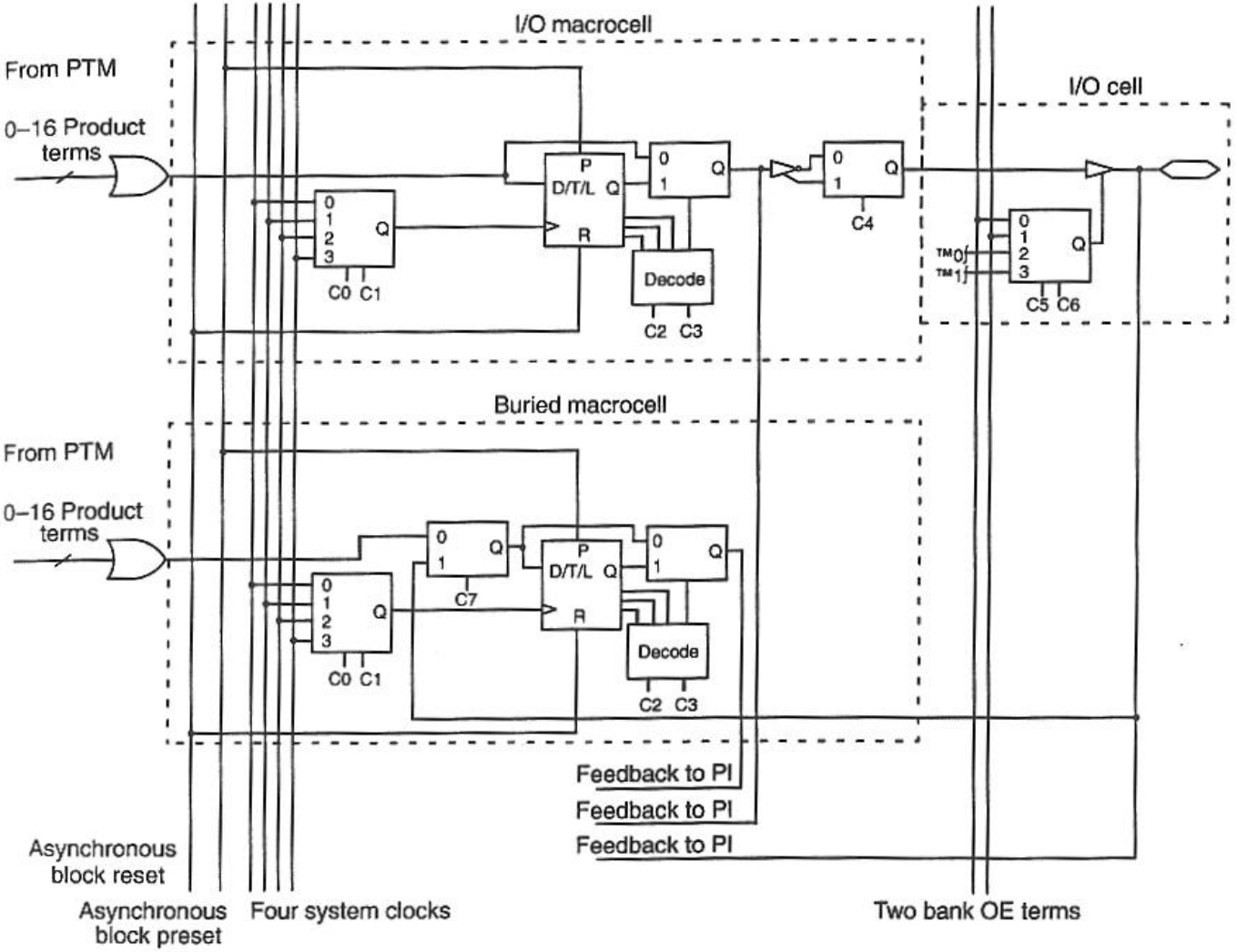
• CPLD, P-TERM DISTRIBUTOR

• **Da 0 a 16 P-term per ogni macrocella (le M-cell centrali sono avvantaggiate)**
  – **Central sharing**
  – **Peripheral steering**



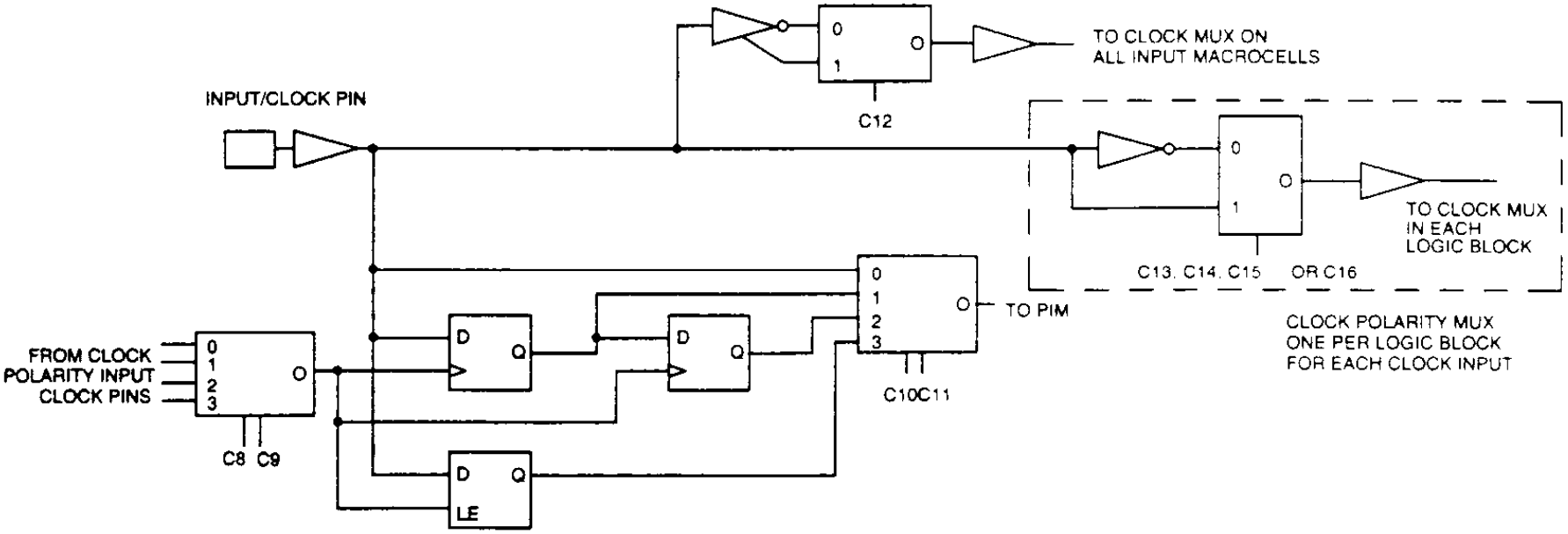
Dispositivi programmabili

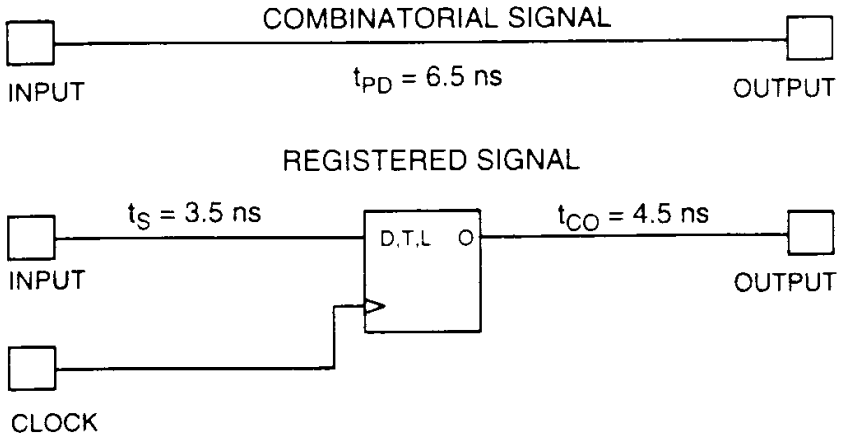
- **CPLD, macrocelle di I/O e buried**



Dispositivi programmabili

• CPLD, macrocella d'ingresso, modello di timing

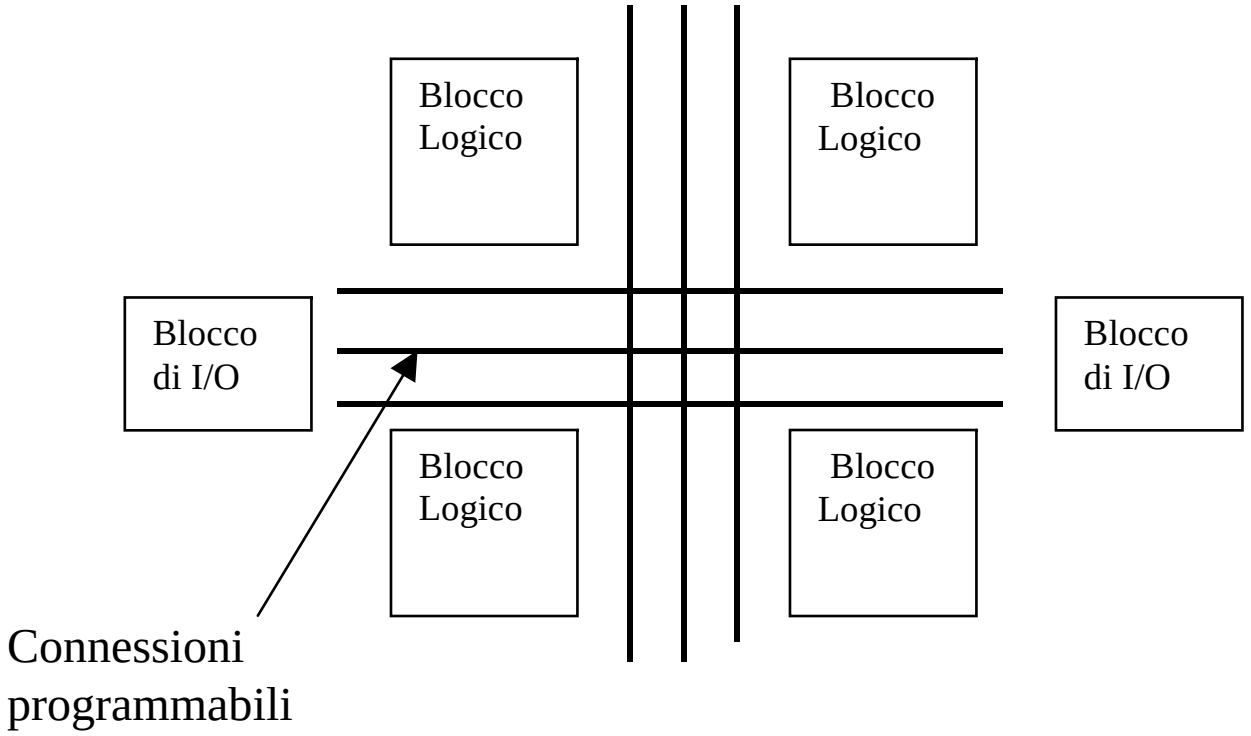




Dispositivi programmabili

- **FPGA, architettura generale**

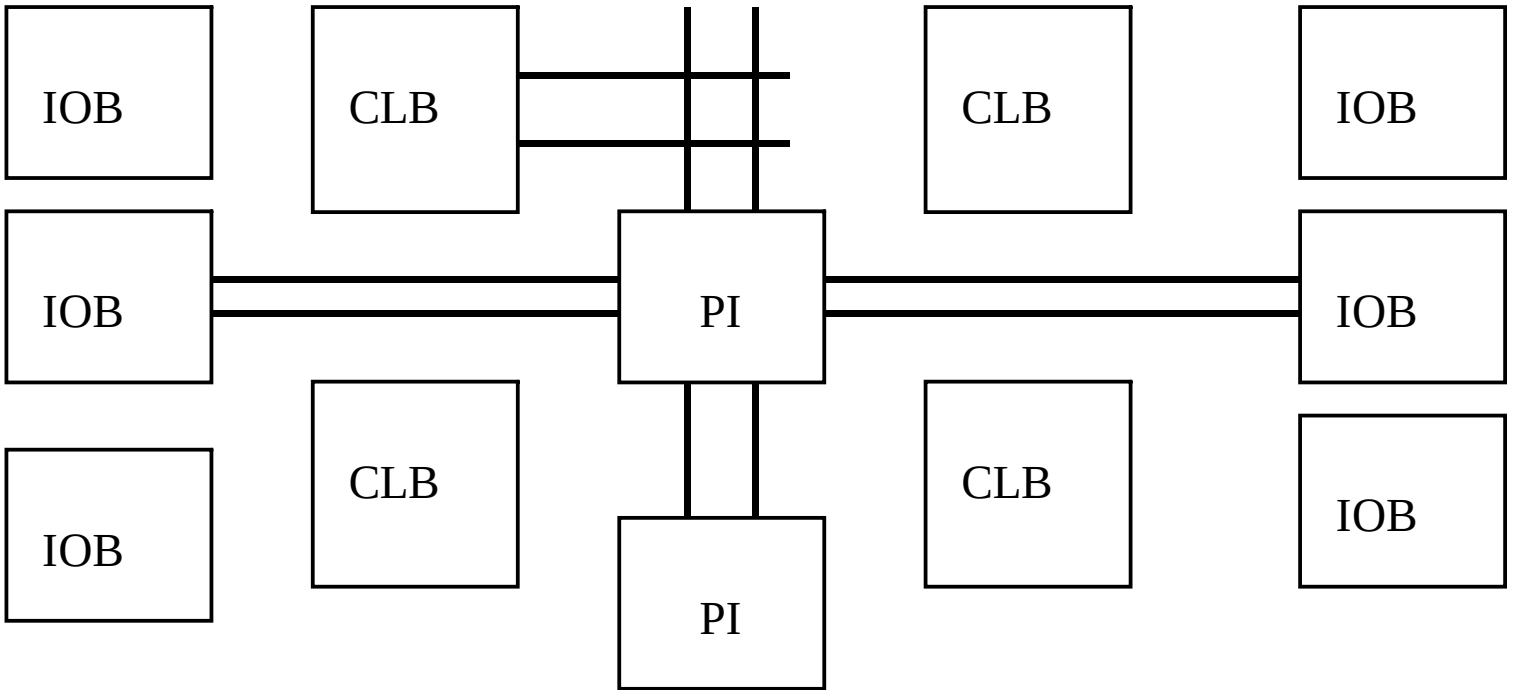
- **Architettura più complessa ma più flessibile delle CPLD**
- **Matrice di macrocelle con griglia di connessioni (prototipi di ASIC)**
  - **uscita = funzione descrivibile in modo generale**
  - **tempo di propagazione da ingresso a uscita dipendente dal tipo di funzione e dalla strategia di connessione delle sottofunzioni elementari**



Dispositivi programmabili

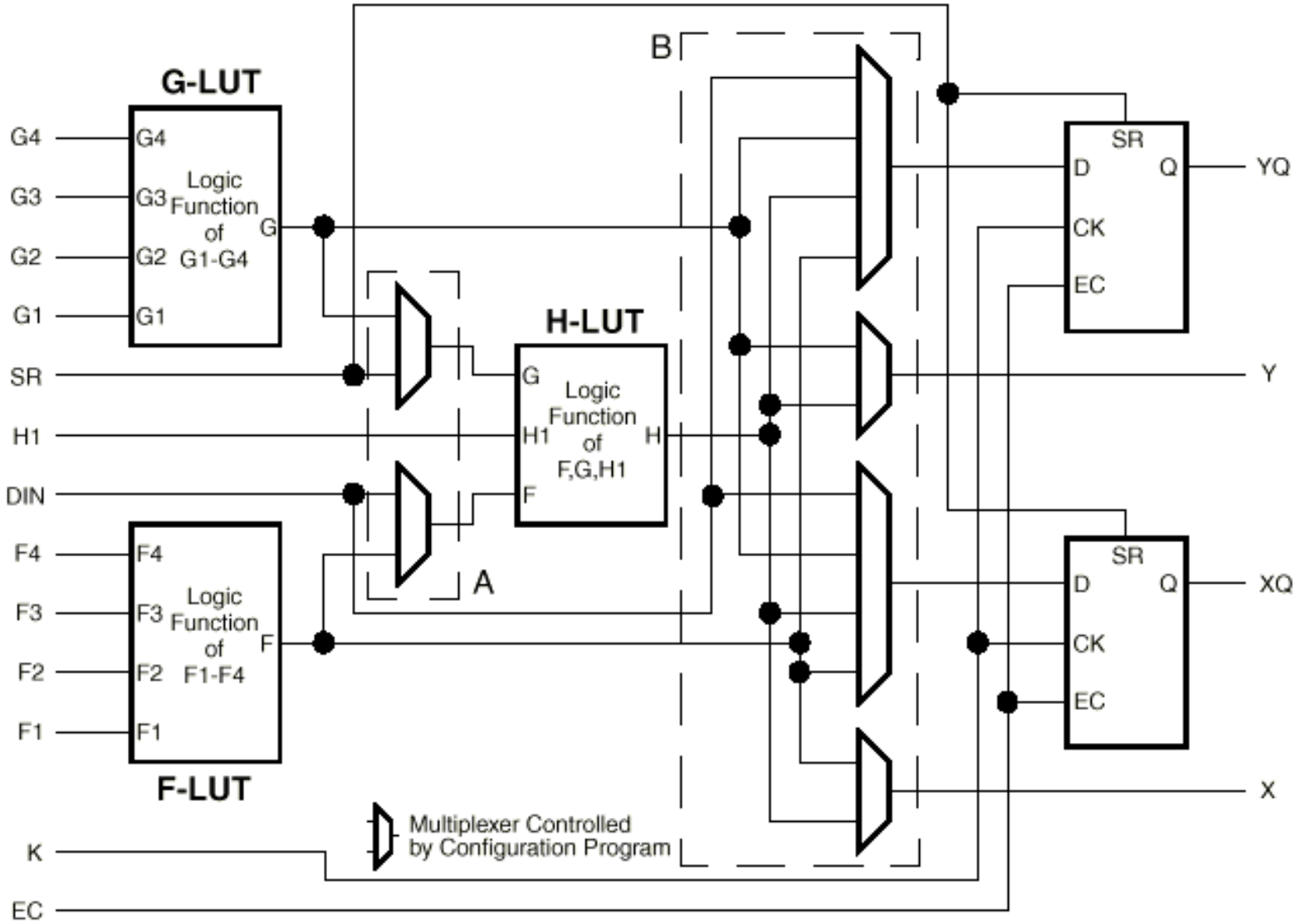
- **FPGA, architettura Xilinx SPARTAN**

- **CLB –Configurable Logic Block- blocchi a logica configurabile o macrocelle**
- **IOB –Input Output Block- blocchi di ingresso-uscita**
- **PI –Programmable Interconnect Interconnessioni programmabili**



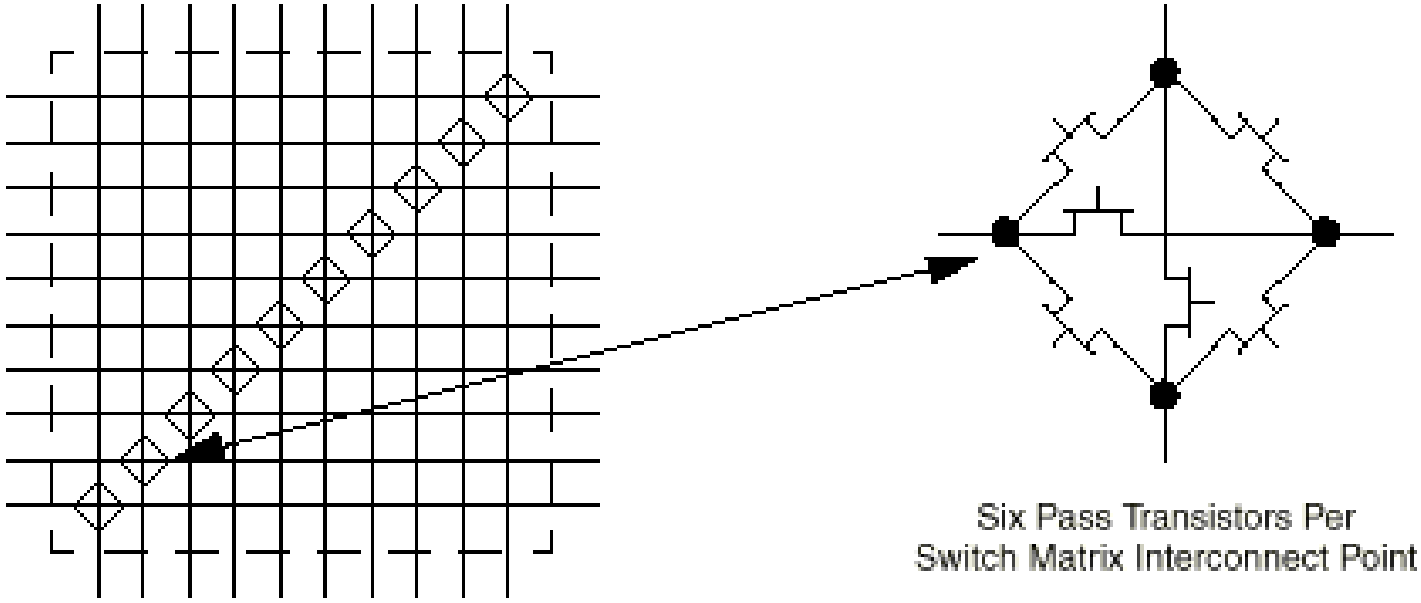
Dispositivi programmabili

- **Blocco logico di una FPGA (Xilinx Configurable Logic Block)**



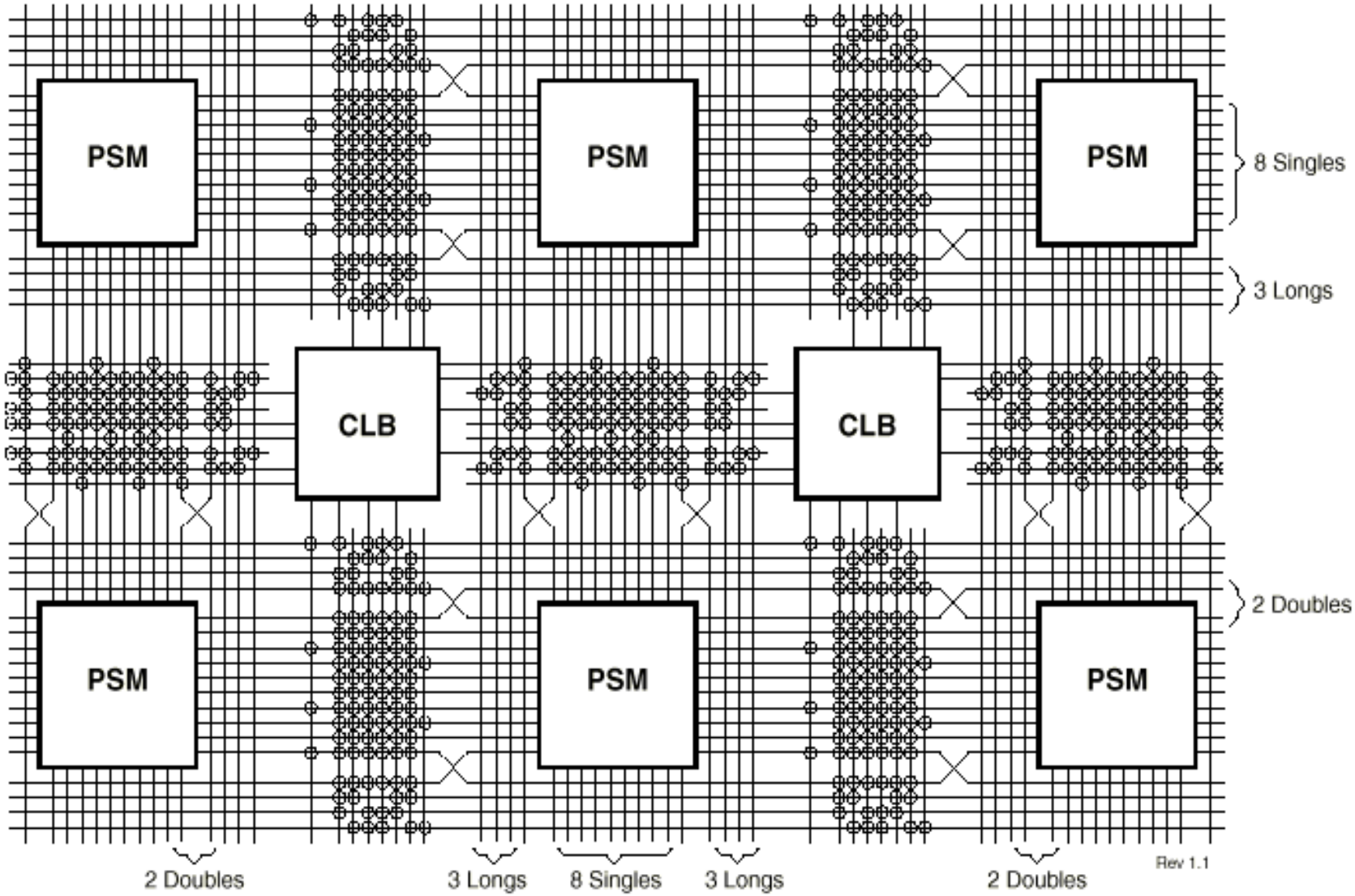
Dispositivi programmabili

- **FPGA, Interconnesioni**
- **Programmable Switch Matrix (PSM)**



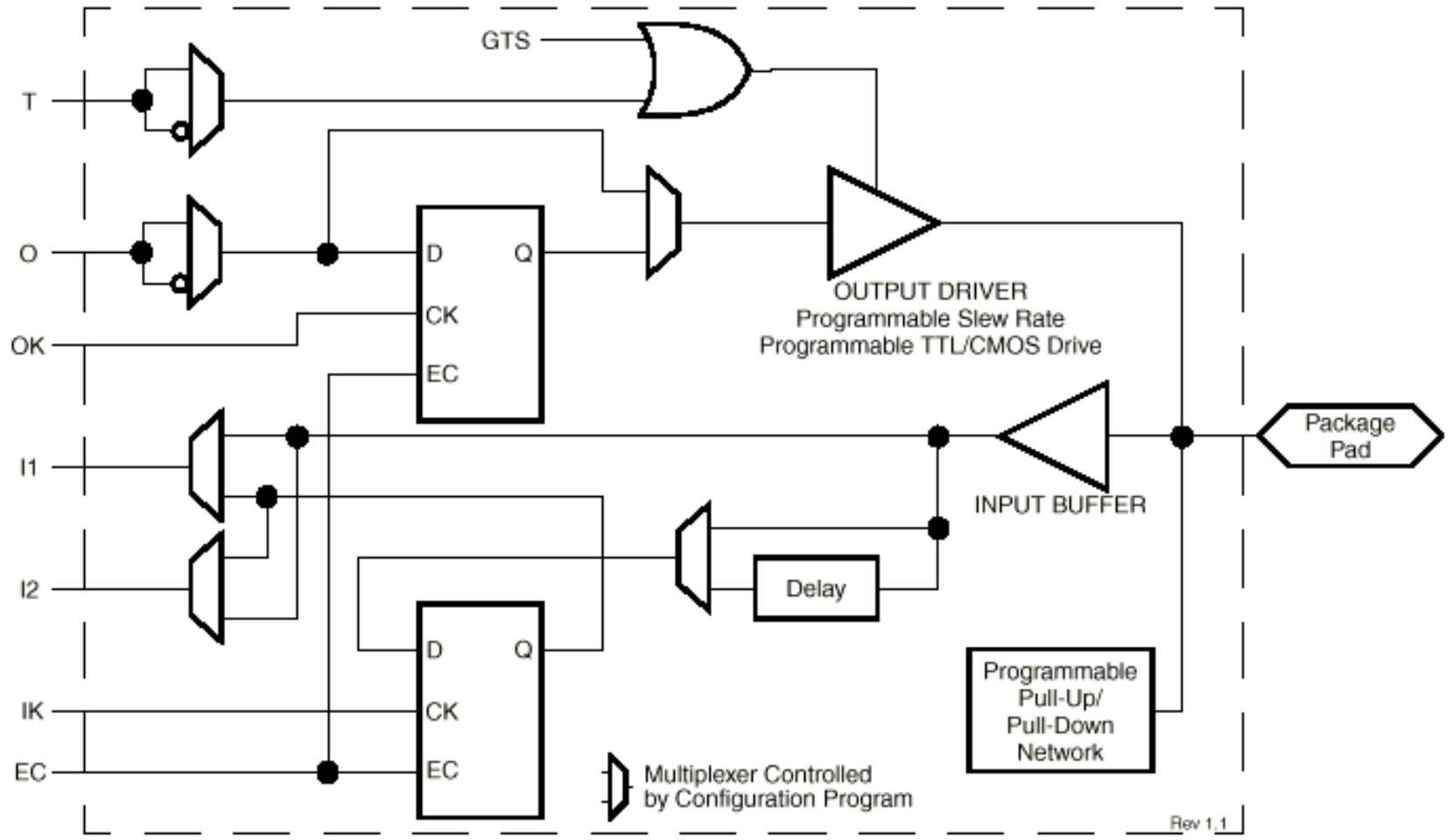
Dispositivi programmabili

- **FPGA, Interconnesioni**



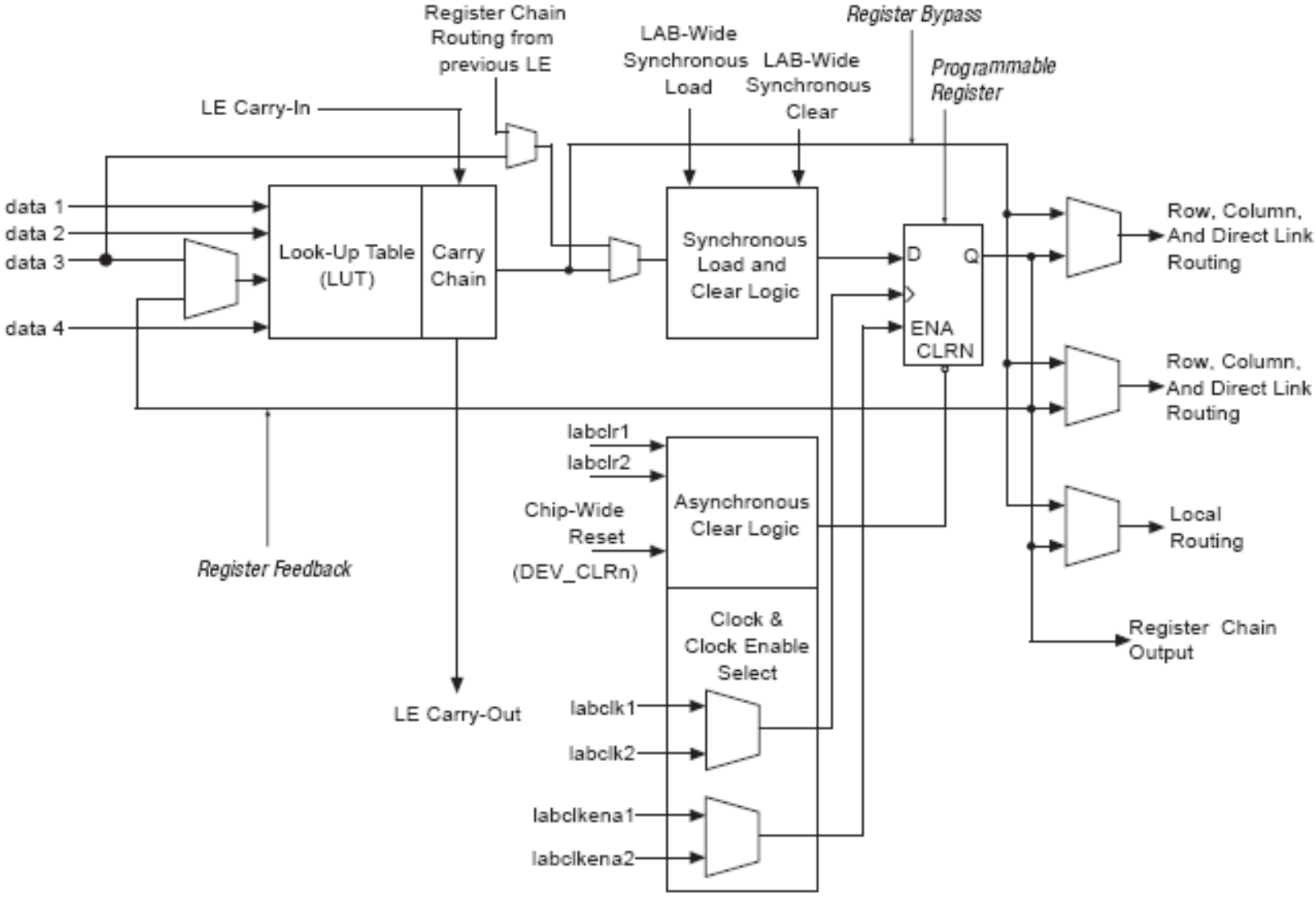
Dispositivi programmabili

- **FPGA, Blocco logico di I/O**



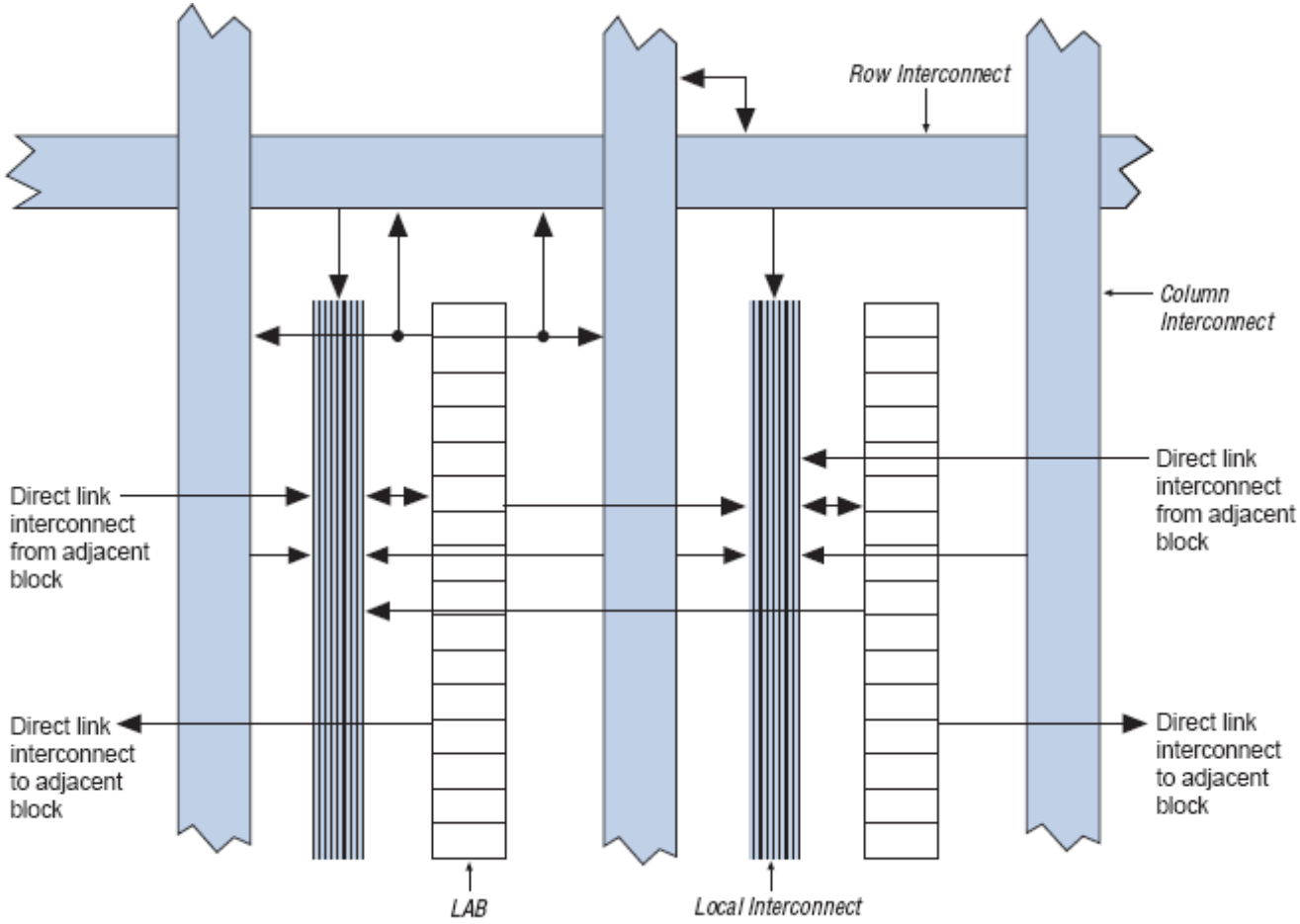
Dispositivi programmabili

- **Nell'FPGA Altera Cyclone III la più piccola unità di logica si chiama Logic Element - LE**



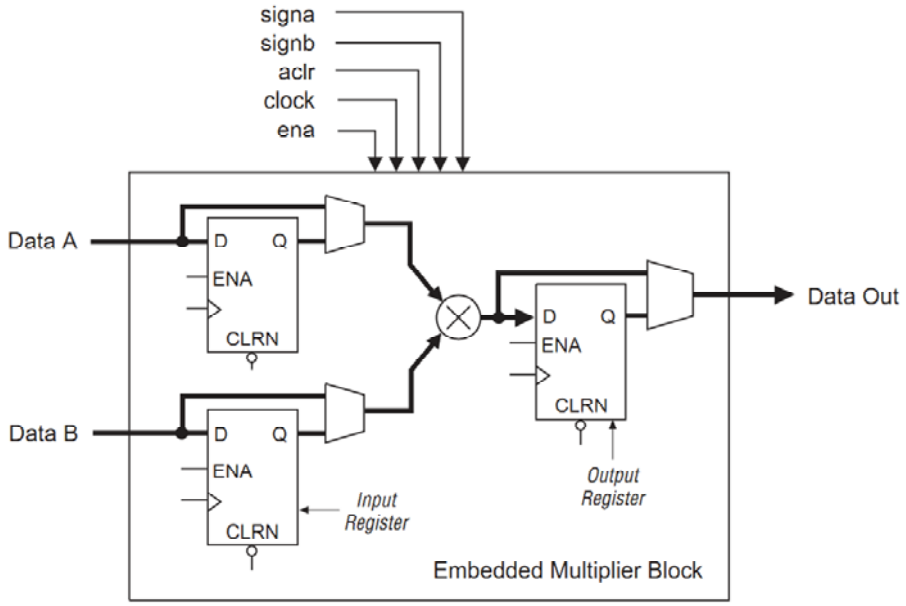
Dispositivi programmabili

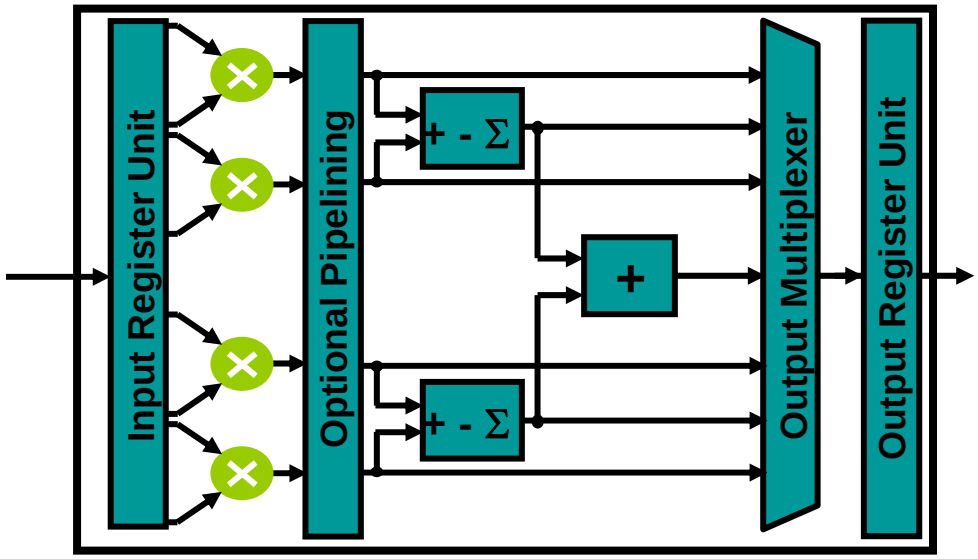
- **Nell'FPGA Altera Cyclone III gruppi di 16 LE formano un Logic Array Block (LAB)**
- **I LE all'interno di un LAB hanno dei percorsi locali di interconnessione**
- **I LE in diversi LAB comunicano tramite le interconnessioni di riga e di colonna**



Dispositivi programmabili

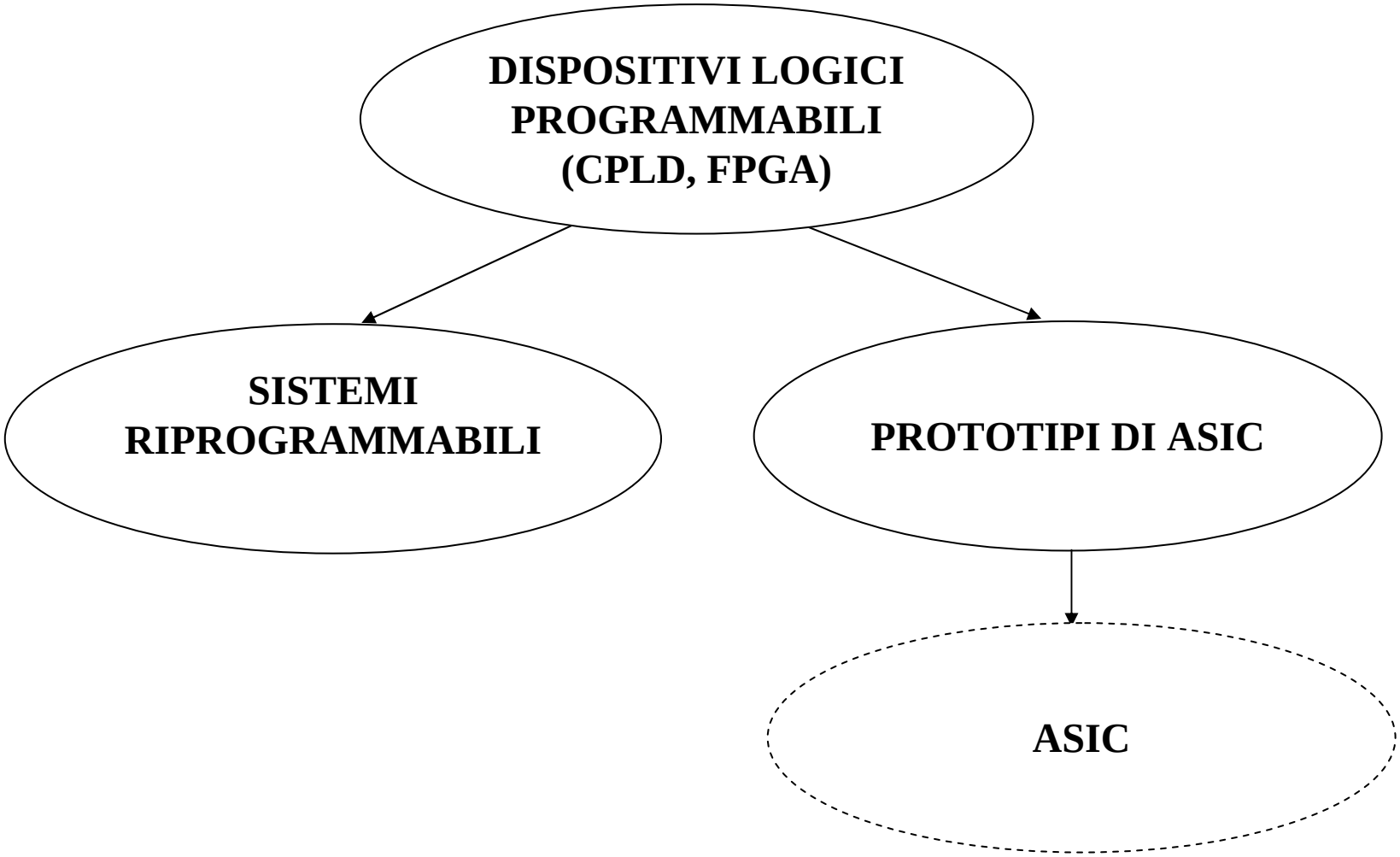
- **Moltiplicatori embedded dell'Altera Cyclone III**
- **Utilizzati per realizzare in modo più efficiente operazioni di moltiplicazione e MAC (Multiplier/Accumulator)**
- **Blocchi DSP (Digital Signal Processor)**





Dispositivi programmabili

- **FPGA e ASIC**



Dispositivi programmabili

- **ASIC**

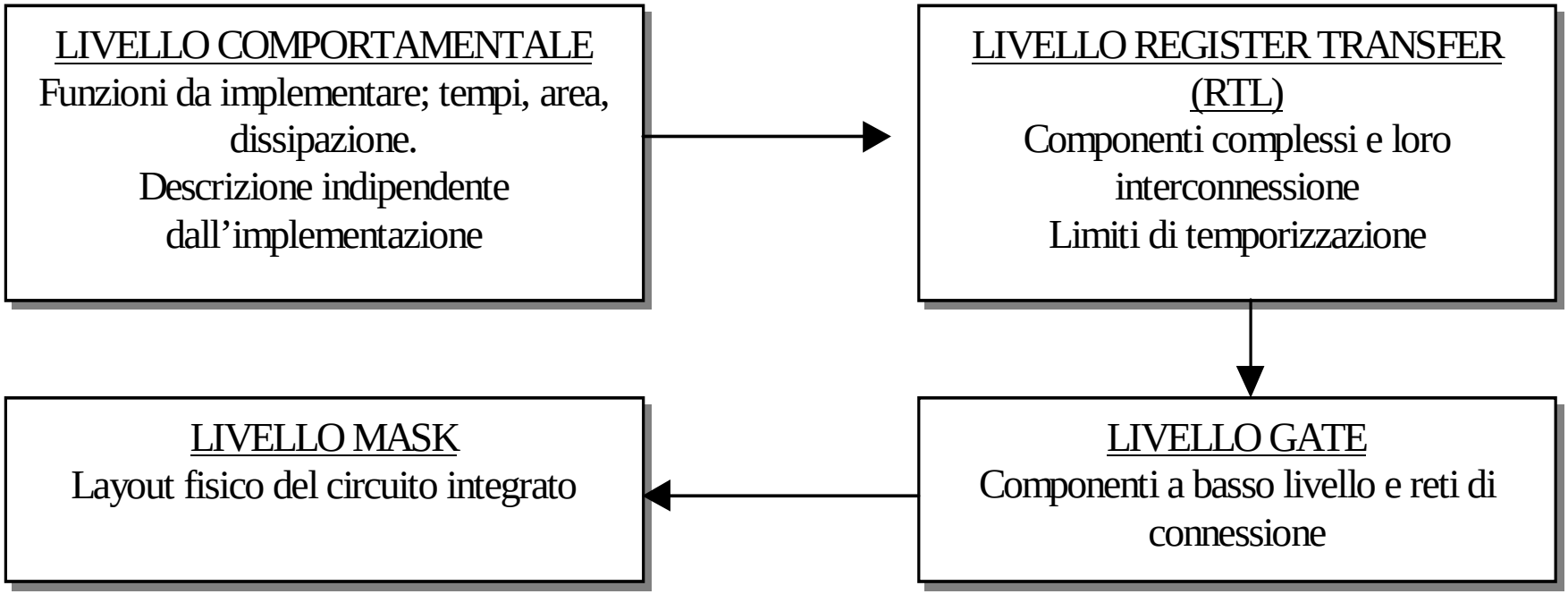
- **VANTAGGI RISPETTO A FPGA/CPLD**
  - **Riduzione costi diretti (l'ASIC, una volta sviluppato, costa meno di una FPGA)**
  - **Migliori prestazioni (progettazione "ad hoc")**
  - **Compattezza (integra più porte e/o funzioni)**
  - **Disposto verso una tecnologia mista (analogica e digitale su uno stesso chip)**

- **SVANTAGGI RISPETTO A FPGA/CPLD**
  - **Elevati costi di sviluppo (>>1 M£)**
  - **Elevati tempi di sviluppo (>>3 mesi)**
  - **Poco adatto ai medio-bassi volumi**
  - **Minore flessibilità (dispositivo intrinsecamente non riprogrammabile)**

Dispositivi programmabili

- **ASIC, FPGA e CPLD: metodologie di sviluppo**

- **Livello di descrizione**



Dispositivi programmabili

- **FPGA e ASIC**

- **Il progetto viene descritto a livello comportamentale mediante linguaggi procedurali orientati alla descrizione di circuiti (Hardware Description Language)**
  - **VHDL (Very High Speed Integration Description Language) Std. IEEE 1164 (1987)**
  - **Per applicazioni speciali si usano diversi formalismi (special Purpose Formalism)**

- **La sintesi ad alto livello si occupa dell'allocazione funzionale**
  (Se ci sono più operazioni di somma si prevede un sommatore se tali operazioni possono essere sequenziate o un numero di sommatori sufficiente a gestire il parallelismo richiesto)

- **La sintesi ad alto livello si occupa dell'implementazione funzionale**
  (Ad un sommatore viene associata un'architettura ripple-carry piuttosto che un carry look-ahead)

Dispositivi programmabili

- **FPGA e ASIC**

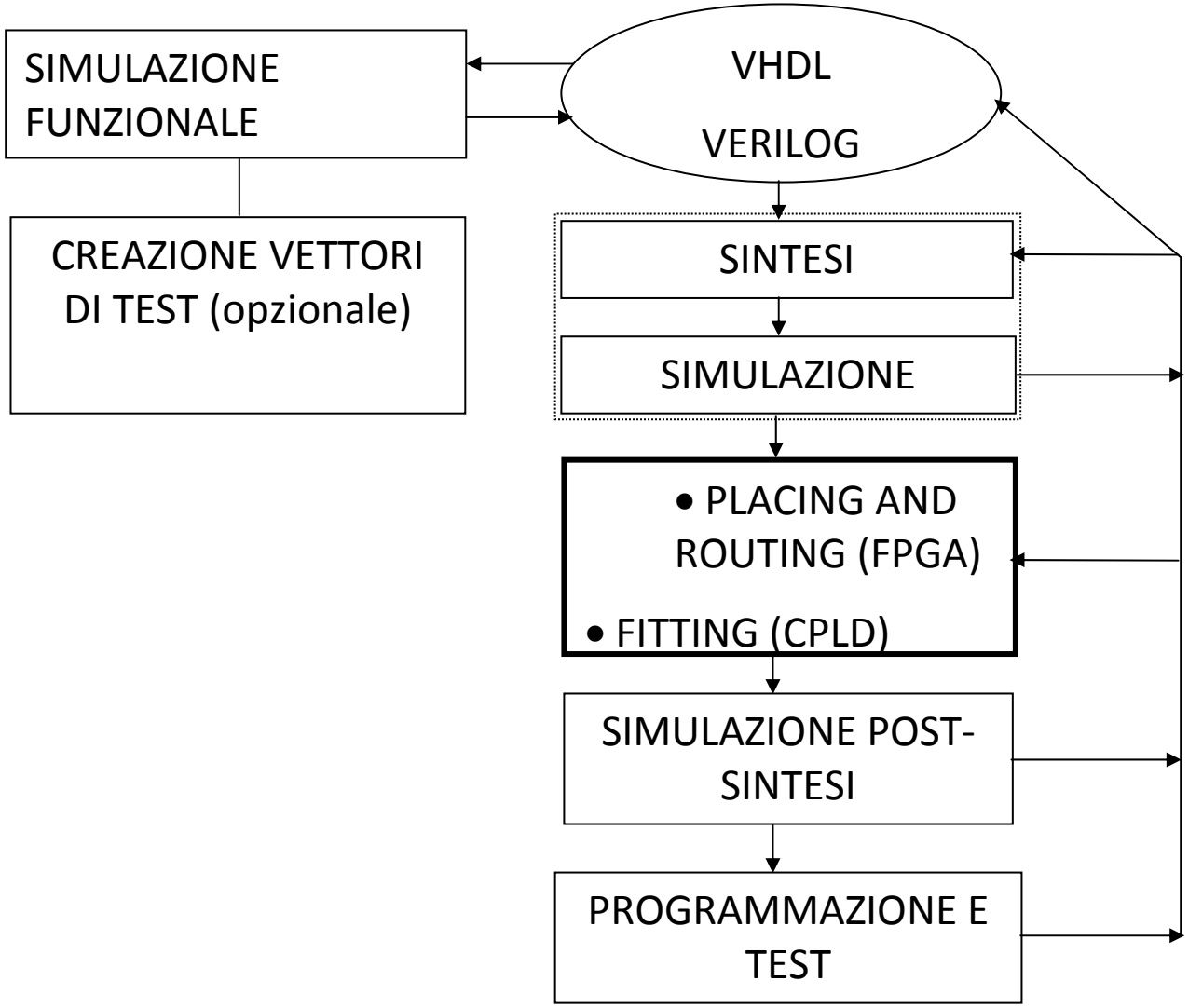
- **Sintesi logica**
  - **Si genera la descrizione a livello “gate” partendo dalla descrizione “RTL” (librerie di “strutture a gate”)**

- **Verifica del progetto**
  - **Design Verification (simulazione ad alto livello o pre-sintesi)**
  - **Implementation Verification (simulazione a livello gate o post-sintesi –PLD- o a livello Layout –ASIC-)**
  - **Manufactoring Verification (verifica del prototipo)**

- **Sintesi fisica**
  - **Insieme di processi tecnologici per la realizzazione fisica del circuito integrato.**
  - **La produzione delle maschere per la realizzazione su silicio del IC richiede sofisticate tecniche per minimizzare l’area di estensione, i ritardi di propagazione e la dissipazione di potenza.**

Dispositivi programmabili

- **Flusso di progetto per FPGA e ASIC**



Dispositivi programmabili

- **Linguaggi per CPLD e FPGA**

- **Linguaggi Booleani**
  (usati per i primi dispositivi logici programmabili –PAL, GAL- adatti a descrivere funzioni in forme SOP –Sum of Product-)
  - **Circuiti combinatori** **Tabelle della verità** **Forme SOP**
  - **Circuiti sequenziali** **Tabelle stato attuale-stato prossimo** **Forme SOP**

- **Linguaggi grafici**
  (si usano componenti delle famiglia 74XX definendo i collegamenti. Di ogni componente esiste uno o più moduli di libreria ottimizzati in area o tempo)
  - **I componenti della 74XX possono non essere i migliori a descrivere i circuiti**
  - **Un linguaggio testuale meglio si presta a strutture di simulazione**

- **Linguaggi testuali (Es. VERILOG)**
  (funzionalmente adatti a ASIC e dispositivi programmabili)
  - **Necessità di uno standard**

Dispositivi programmabili

- **Storia del VHDL**

- **Programma USA chiamato VHSIC (Very High Speed Integrated Circuit) per progettazione di circuiti integrati complessi (inizio 1980)**

- **VHDL (VHSIC Hardware Description Language) proposto come standard IEEE (1986) e ratificato come ANSI/IEEE Std 1076-1987 (aggiornato nel 1993)**

- **ANSI/IEEE Std 1164 (1992) -> multi-value logic**
  sono previsti 9 valori logici per il tipo bit e per i tipi derivati:
  "0", "1", "Z"-alta impedenza-, "X"-impredicibile-, "W"-impredicibile alta impedenza-, "U"-non inizializzato-, "L"-pull down-, "H"-pull up-, "-"-indifferente-

- **VHDL, rispetto ai metodi tradizionali, mostra i seguenti vantaggi:**
  - **Migliore documentazione**
  - **Potenza nella simulazione**
  - **Migliore descrizione del processo di sintesi**

Dispositivi programmabili

- **VHDL, modalità di descrizione**

- **Molteplici modalità di descrizione (Es. comparatore a due bit)**
  - **A) e B) descrizioni strutturali (schema funzionale, linguaggio booleano)**
  - **C) e D) descrizioni comportamentali (generica, solo per process)**

A) netlist

```
U1: xor2 port map(a(0), b(0), x(0));
U2: xor2 port map(a(1), b(1), x(1));
U3: nor2 port map(x(0), x(1), aeqb);
```

B) Boolean equations

```
aeqb <= (a(0) XOR b(0)) NOR
    (a(1) XOR b(1));
```

C) Concurrent statements

```
aeqb <= “1” when a=b else “0”;
```

D) Sequential statements

```
If a=b then aeqb <= “1”;
else aeqb <= “0”;
endif
```