

# Dispositivi Programmabili

# Dispositivi programmabili

- Sono dispositivi hardware (chip) che mettono a disposizione elementi logici più o meno complessi che possono essere opportunamente interconnessi secondo diverse configurazioni in funzione delle specifiche di progetto
- Dispongono di
  - Componenti logici (porte logiche, Flip-Flop, Buffer)
  - Linee di connessione
  - Sistemi di interconnessione (Multiplexer, connessioni)
  - Porte di I/O
- Tipologie di Circuiti Programmabili
  - PLA, PAL, ROM.
  - CPLD
  - FPGA

# Dispositivi Programmabili

- I diversi dispositivi possono essere classificati in base a diversi aspetti:
  - Modalità di programmazione
    - programmabili a maschera (MPGA)
    - programmabili una volta (Fuse o Antifuse)
    - riprogrammabili (EEPROM, SRAM)
    - riconfigurabili (SRAM)
  - Connessioni
    - Globali
    - Locali e distribuite

# Modalità di programmazione

## – FUSE

- Le connessioni tra linee sono inizialmente tutte attive
- In fase di programmazione si disattivano permanentemente le connessioni inutili

## – ANTIFUSE

- Le connessioni tra linee sono inizialmente tutte inattive
- In fase di programmazione si attivano permanentemente le connessioni utili

## – EEPROM

- le connessioni inizialmente sono tutte inattive
- In fase di programmazione si possono attivare o disattivare elettricamente in modo non distruttivo
- Lo stato viene mantenuto anche in assenza di alimentazione

# Modalità di programmazione

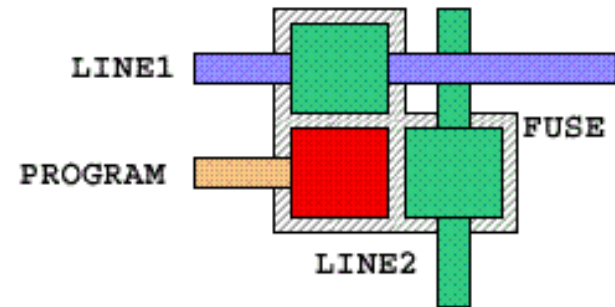
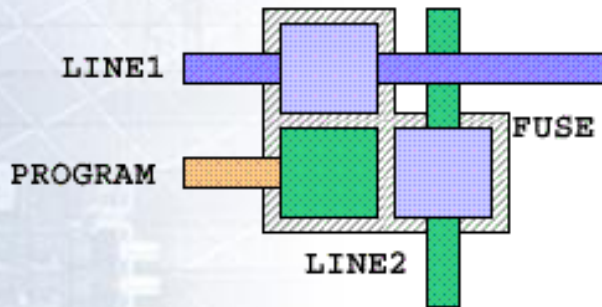
## – SRAM

le connessioni inizialmente sono tutte inattive

- In fase di programmazione si possono attivare o disattivare elettricamente in modo non distruttivo
- Lo stato NON viene mantenuto in assenza di alimentazione
- Maggiore velocità di programmazione rispetto la tecnologia EEPROM
- In base alla tecnologia la programmazione può avvenire:
  - Durante la fase non operativa del dispositivo (riprogrammabile)
  - Durante la fase operativa del dispositivo (ricongfigurabile)
    - Si interviene separatamente su varie parti del dispositivo

# Fuse

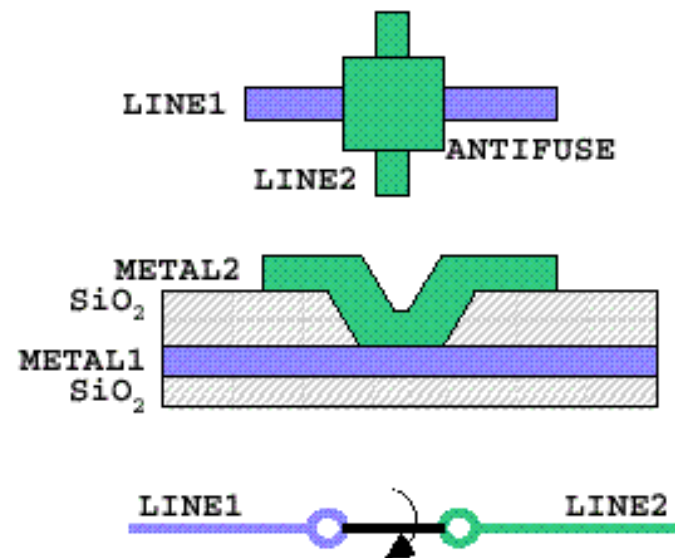
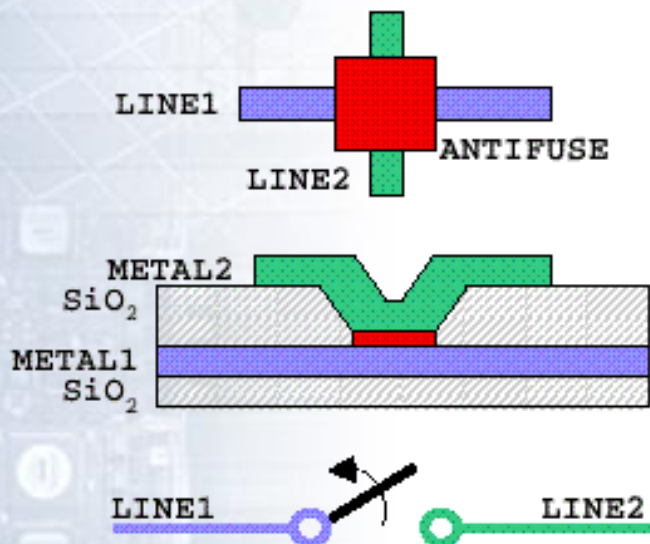
- Le linee del dispositivo sono in origine tutte connesse
  - La programmazione consiste nel “BRUCIARE” (fuse) alcune connessioni in modo tale da mantenere solo quelle necessarie
  - La programmazione avviene mediante una tensione più elevata di quella di normale funzionamento





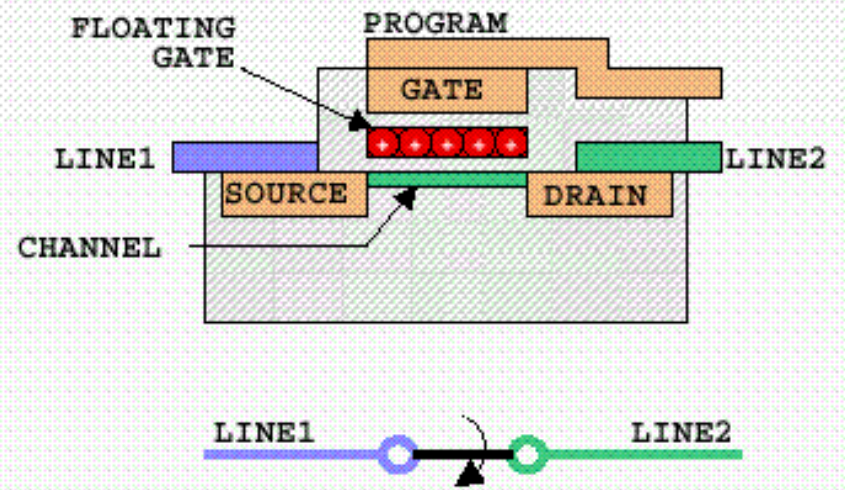
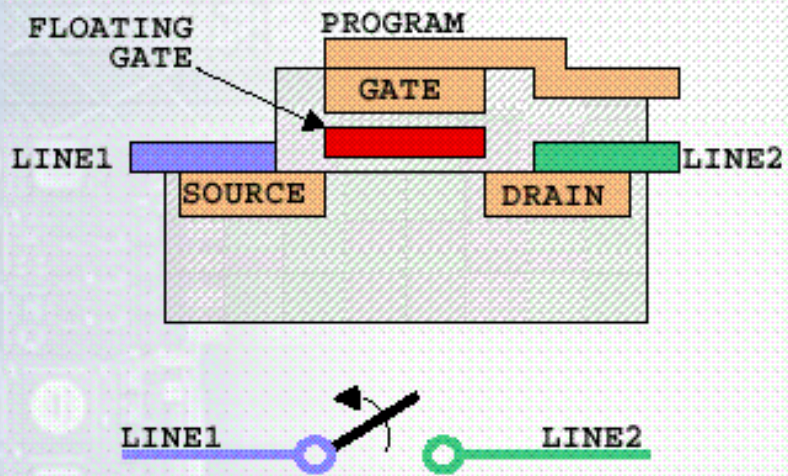
# Antifuse

- Le linee del dispositivo sono in origine tutte disconnesse
  - La programmazione consiste nel “**CREARE**” (antifuse) le connessioni necessarie
  - La programmazione avviene mediante una tensione più elevata di quella di normale funzionamento



# EEPROM

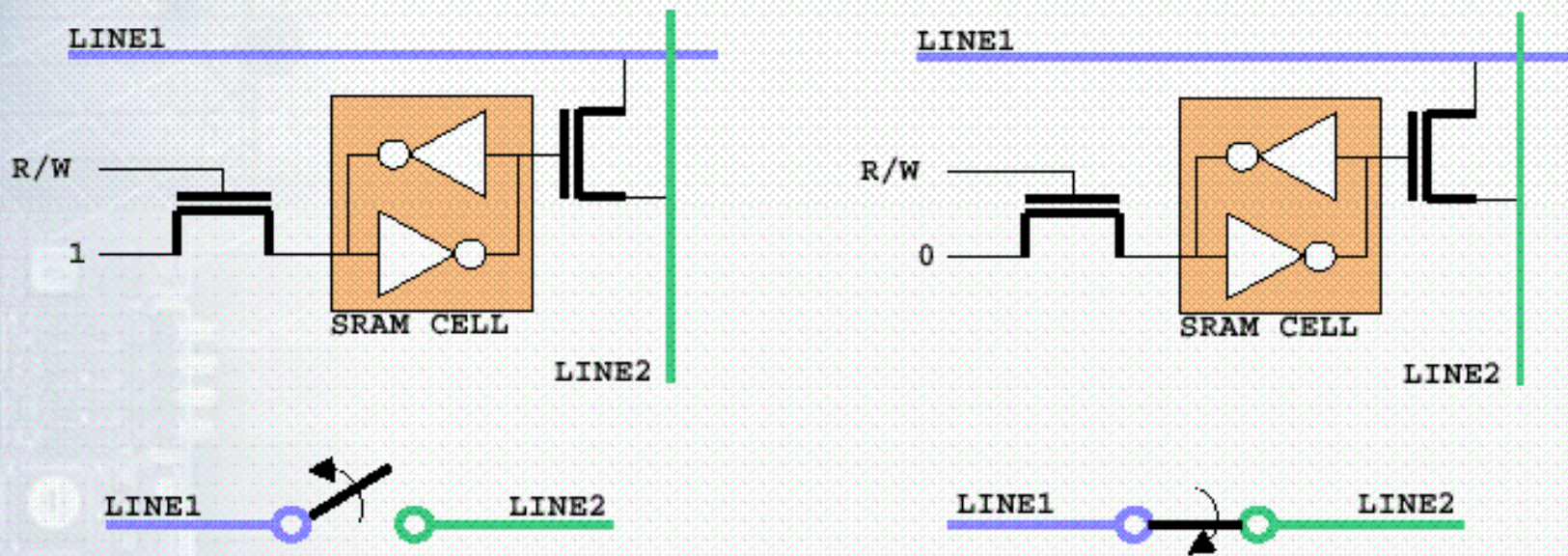
- Le linee del dispositivo sono in origine tutte disconnesse
  - La programmazione consiste nel “**DEPOSITARE**” una carica sul gate flottante del transistor in modo da mandarlo in conduzione
  - La cancellazione può avvenire elettricamente o tramite esposizione a raggi UV





# SRAM (RAM statica)

- Le linee del dispositivo sono in origine tutte disconnesse
  - La programmazione consiste nel “**MEMORIZZARE**” un valore logico (‘0’ o ‘1’) in una cella di RAM Statica



# Connessioni

- Le connessioni globali sono caratteristiche dei
  - Dispositivi logici a due livelli:
    - PAL, PLA, ROM
  - CPLD (Complex Programmable Logic Device)
- Le connessioni locali sono caratteristiche degli
  - FPGA (Field Programmable Gate Array)

# Logiche programmabili a 2 livelli

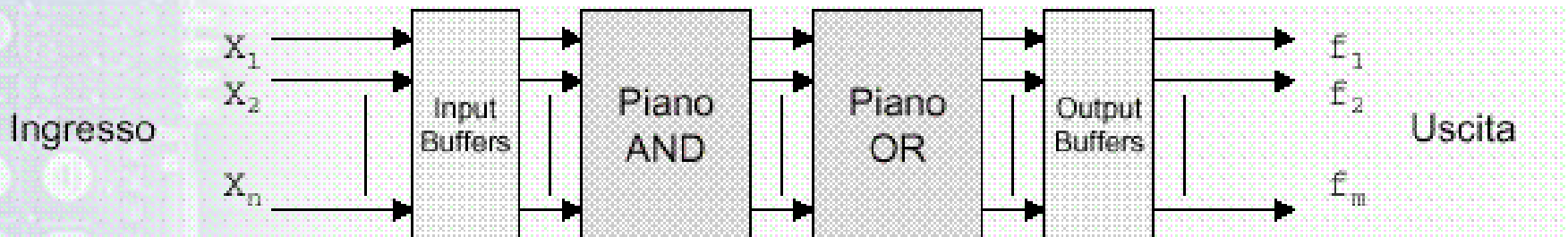
- Sono usate per realizzare funzioni logiche a due livelli

**NOTA1:** qualunque funzione combinatoria puo' essere espressa come somma di termini minimi

- **NOTA2:** si possono realizzare funzioni a piu' livelli sfruttando la retroazione

- Dispongono di:

- Un numero di ingressi fissato (Buffer di Ingresso)
- Un piano di AND (per realizzare i termini minimi)
- Un piano di OR (per realizzare le somme)
- Un numero di uscite fissato (Buffer di Uscita)

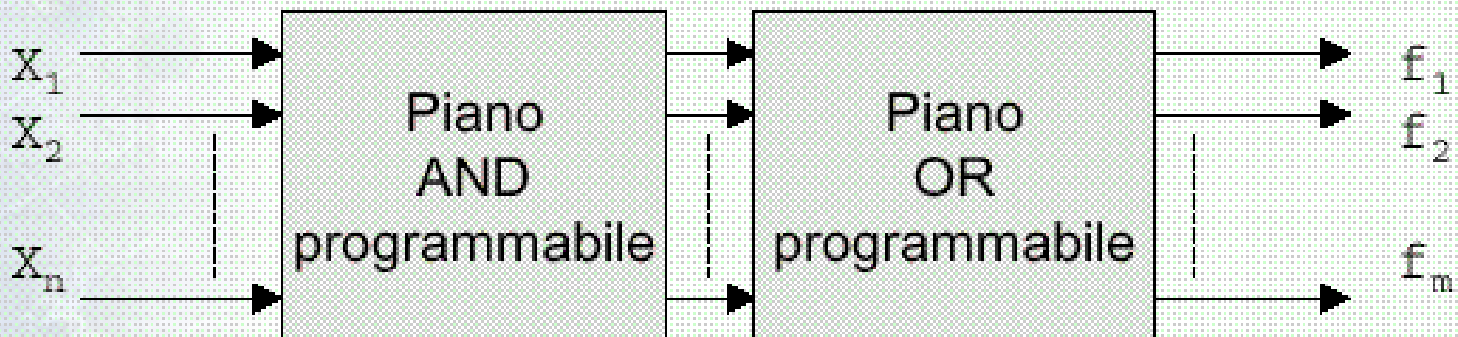


# Logiche programmabili a 2 livelli

- Vi sono tre tipi principali
  - PLA (Programmable Logic Array)
    - Piano AND programmabile
      - Implementa solo i termini minimi necessari
    - Piano OR programmabile
  - PAL (Programmable Array Logic)
    - Piano AND programmabile
    - Piano OR fissato
      - Impone un vincolo sul numero di termini minimi che la funzione contiene
  - ROM (Read Only Memory)
    - Piano AND fissato
      - Implementa tutti i possibili termini minimi (DECODER)
    - Piano OR programmabile

# Programmable Logic Array (PLA)

- Consente di realizzare qualunque funzione logica
- Questa e' espressa in somme di implicantsi

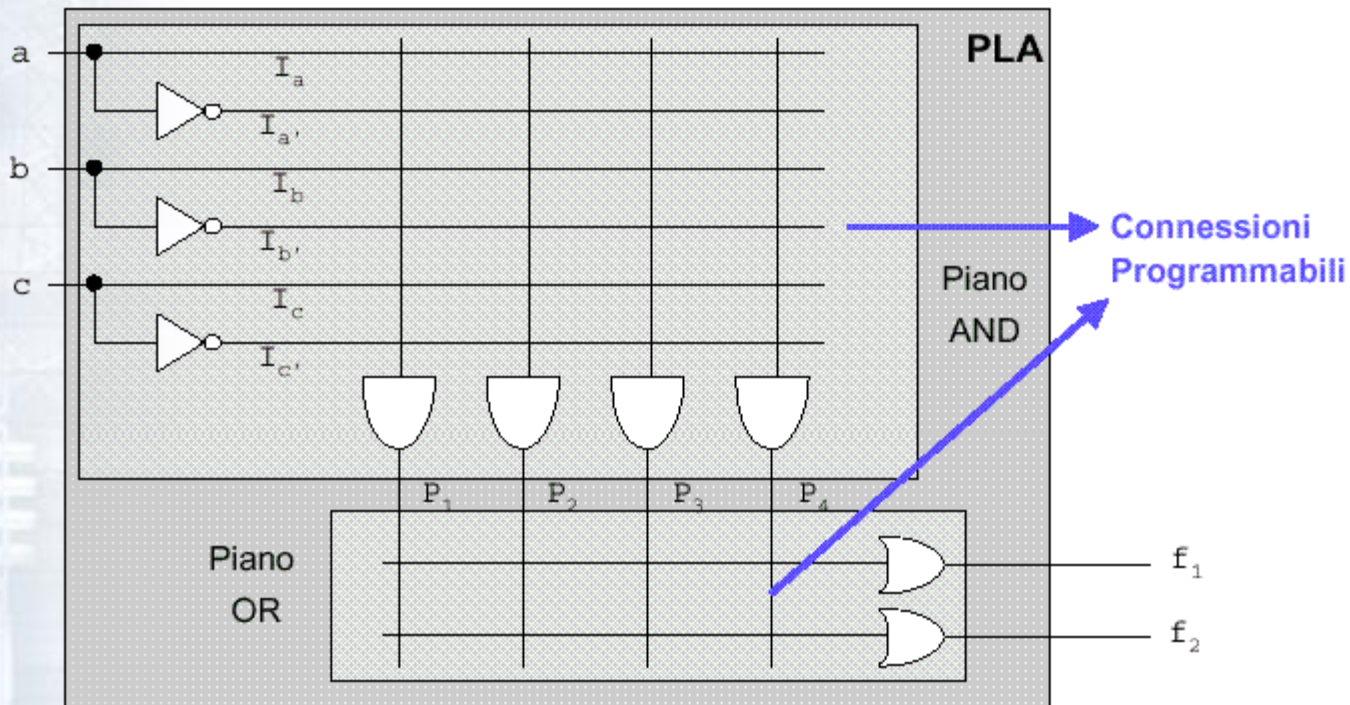




# Programmable Logic Array (PLA)

## ■ Schema logico di una PLA

- Esempio con 3 ingressi e due uscite (non programmata)



# Programmable Logic Array (PLA)

## □ Esempio 1:

- Realizzazione delle funzioni:

$$f_1 = ab + ac' + a'b'c$$

$$f_2 = ab + ac + a'b'c$$

- Prodotti:

$$P_1 = ab$$

$$P_2 = ac$$

$$P_3 = ac'$$

$$P_4 = a'b'c$$

- Somme:

$$f_1 = P_1 + P_3 + P_4$$

$$f_2 = P_1 + P_2 + P_4$$

Formato PLA:

11- 10

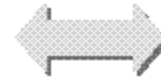
1-0 10

001 10

11- 01

1-1 01

001 01



# Programmable Logic Array (PLA)

Formato PLA:

11- 10

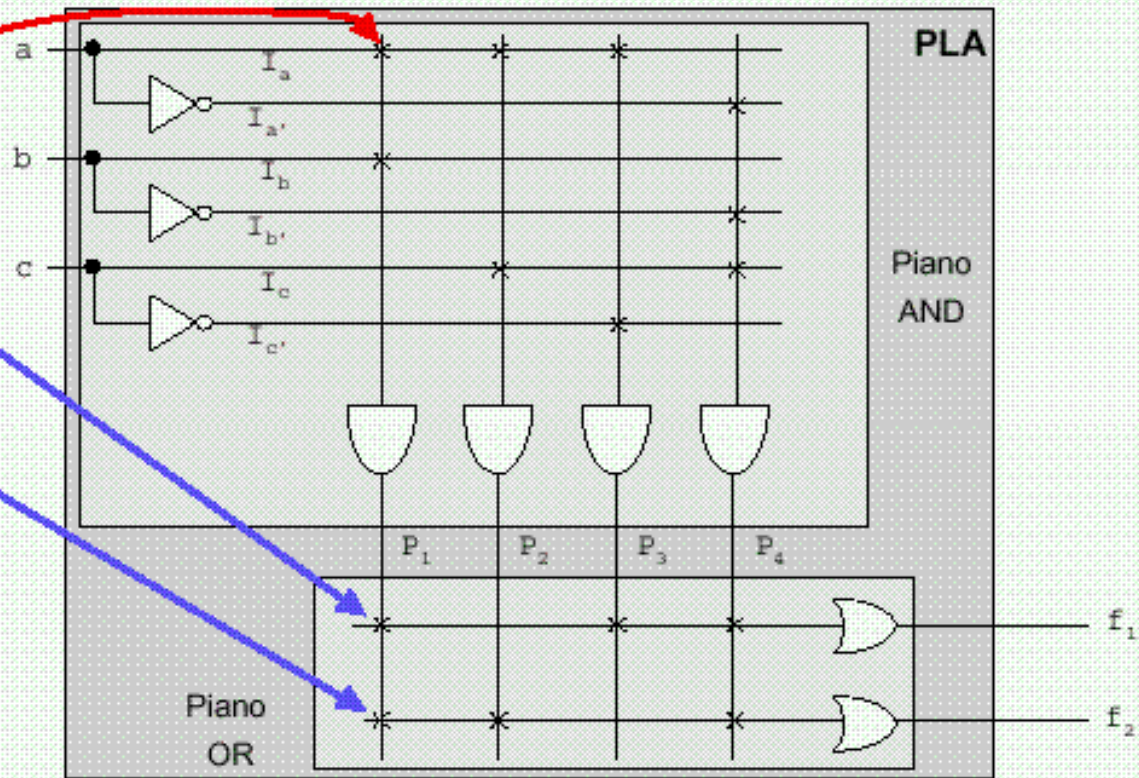
1-0 10

001 10

11- 01

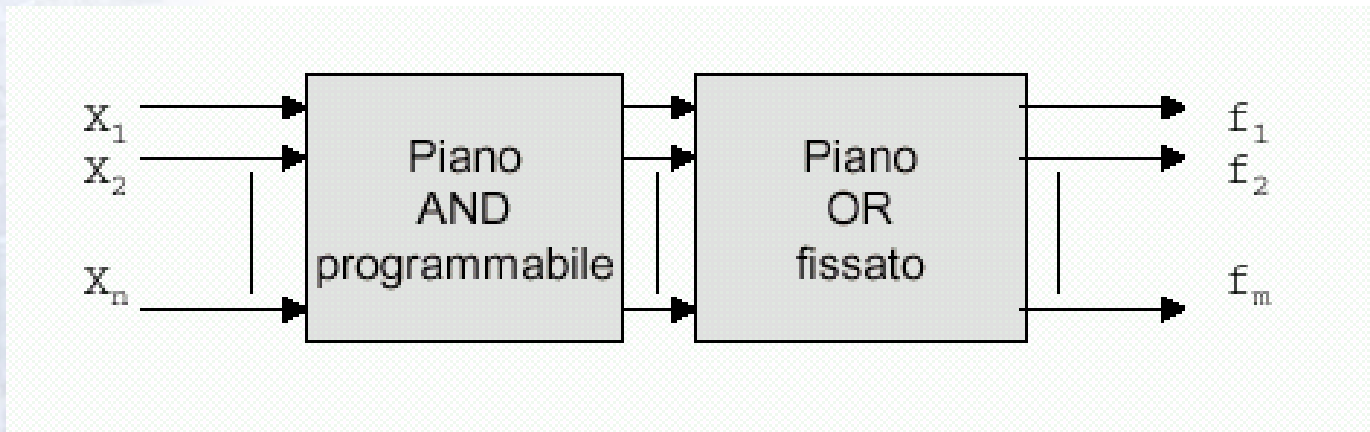
1-1 01

001 01



# Programmable Array Logic (PAL)

- Piano di AND programmabile e piano OR fissato
- Consente di implementare somme di prodotti
  - Vi può essere un limite sul numero massimo di prodotti che possono concorrere nella realizzazione di una funzione

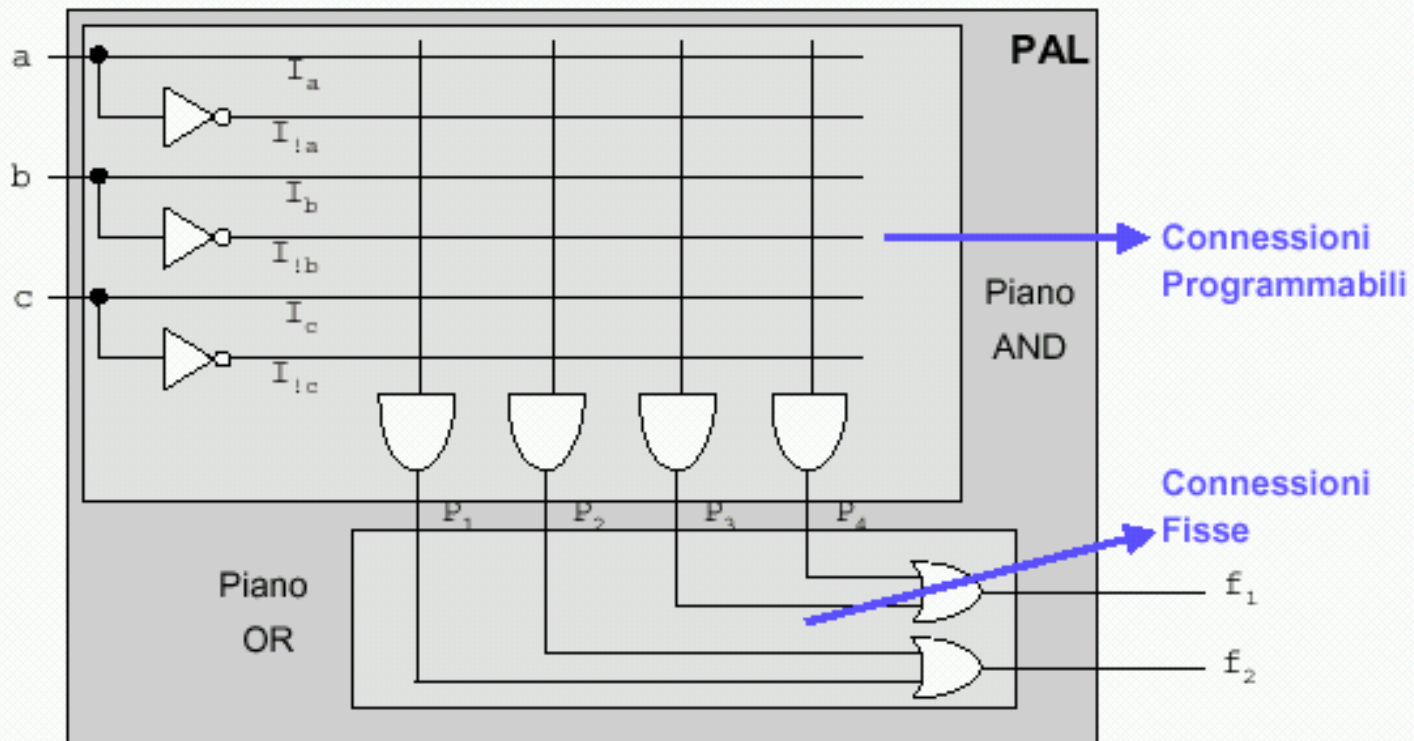




# Programmable Array Logic (PAL)

## ■ Schema logico di una PAL

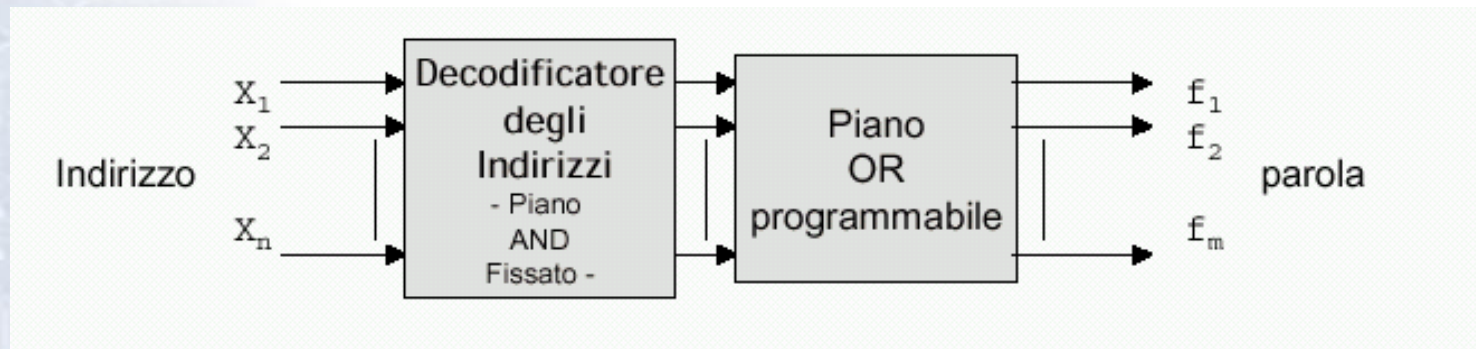
- Esempio di PAL a 3 ingressi e 2 uscite (non programmata)





# Read Only Memory (ROM)

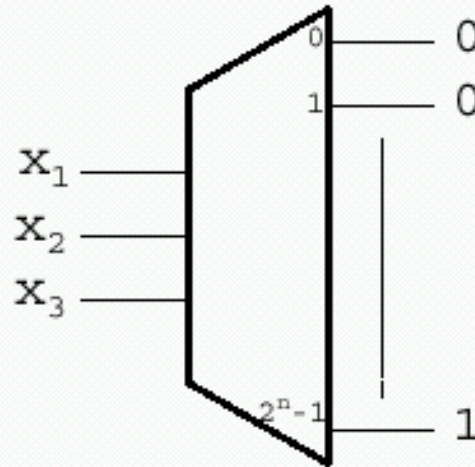
- Può essere realizzata con un piano di AND fisso e completo e con un piano di OR programmabile
- In pratica implementa  $m$  funzioni a  $n$  ingressi
- ad una configurazione d'ingresso (INDIRIZZO) viene associata una configurazione d'uscita (PAROLA)
- Il piano AND agisce da DECODIFICATORE degli indirizzi



# Read Only Memory (ROM)

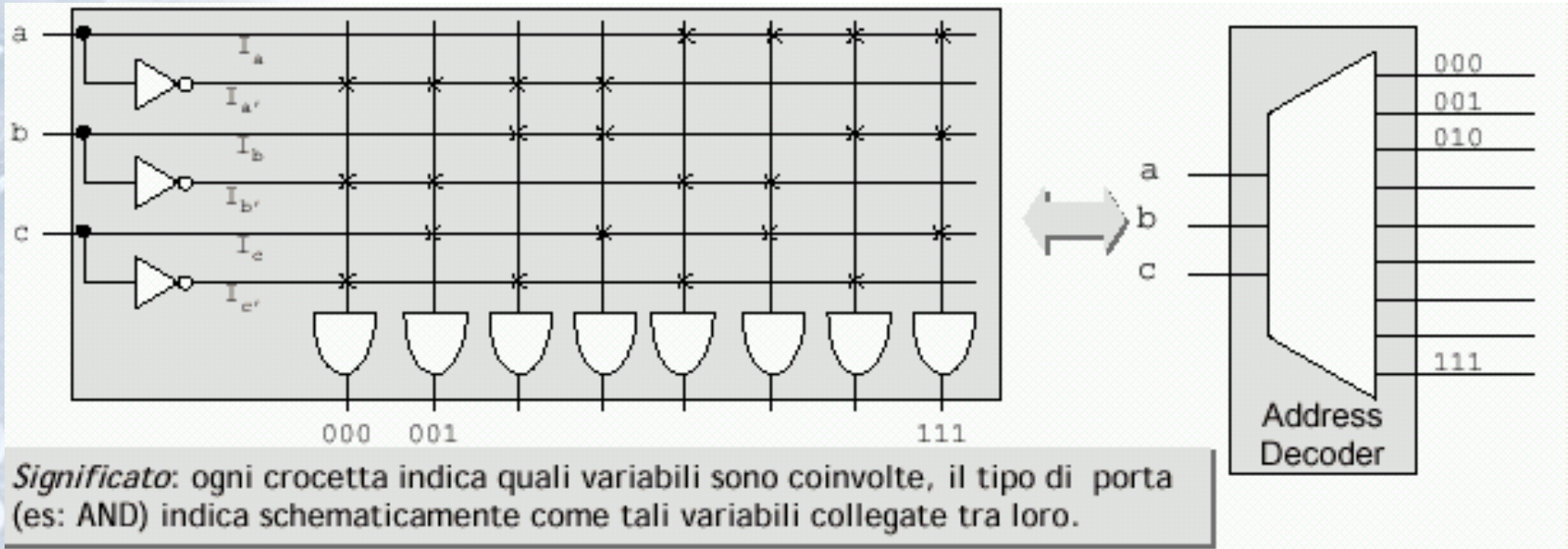
- Piano AND (decodificatore degli indirizzi)
  - realizza **tutti** i possibili termini minimi
  - per ogni configurazione d'ingresso attiva una ed una sola linea d'uscita

Esempio:  
 $x_1x_2x_3 = 111$



# Read Only Memory (ROM)

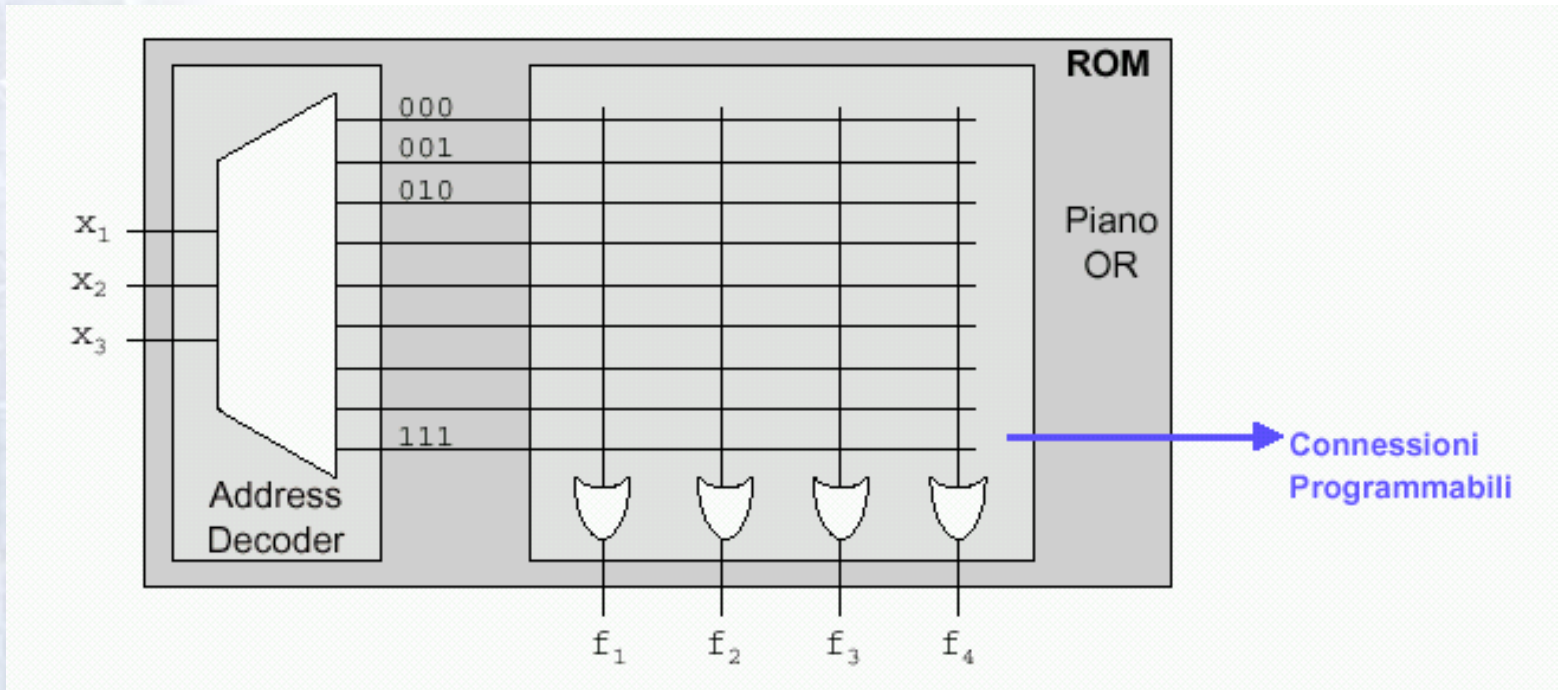
## ■ Schema logico del piano AND



# Read Only Memory (ROM)

## ■ Schema logico di una ROM

- Esempio di una ROM a 3 ingressi e 4 uscite (non programmata)



# Read Only Memory (ROM)

## ■ Esempio:

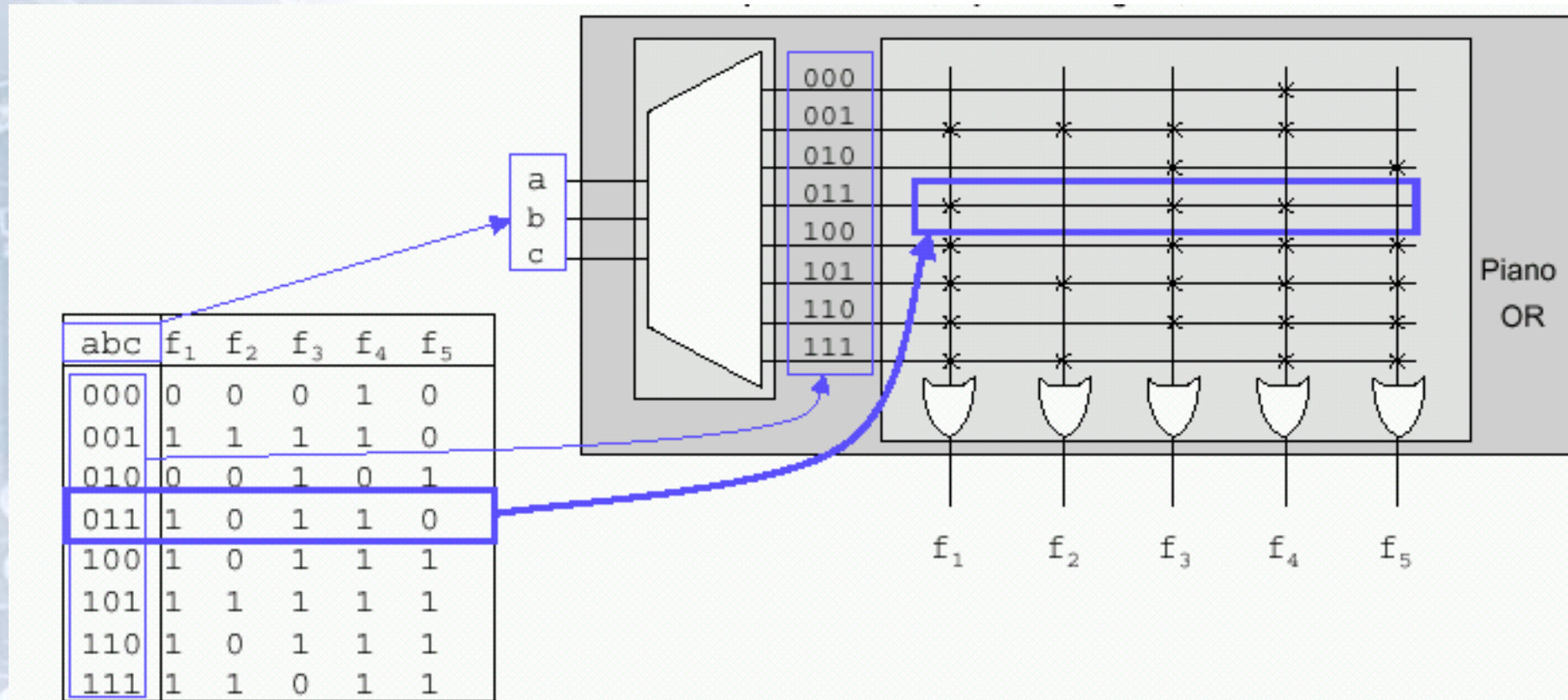
- dalla tabella di verità della funzione a più uscite

<b>abc</b>	<b>f1</b>	<b>f2</b>	<b>f3</b>	<b>f4</b>	<b>f5</b>
<b>000</b>	0	0	0	1	0
<b>001</b>	1	1	1	1	0
<b>010</b>	0	0	1	0	1
<b>011</b>	1	0	1	1	0
<b>100</b>	1	0	1	1	1
<b>101</b>	1	1	1	1	1
<b>110</b>	1	0	1	1	1
<b>111</b>	1	1	0	1	1



# Read Only Memory (ROM)

## ■ Realizzazione della funzione

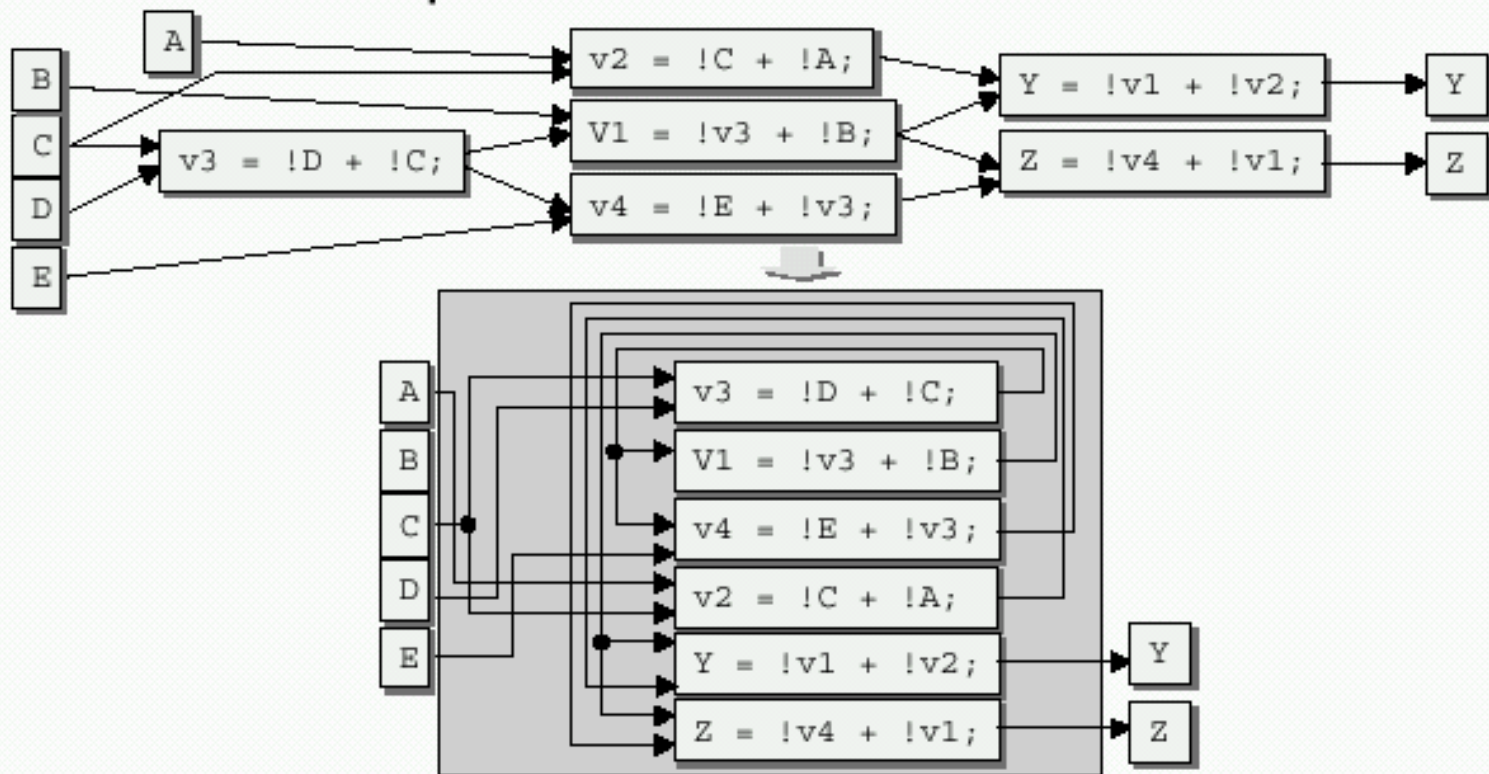


# PLA e PAL avanzate

- PLA e PAL consentono di realizzare solo reti combinatorie a due livelli
- Questo limite può essere superato
  - Introducendo una rete di reazione
    - permette di implementare reti combinatorie a più di due livelli
  - Introducendo elementi di memoria (Flip-Flop)
    - permette di implementare macchine sequenziali (sincrone)

# PLA e PAL avanzate

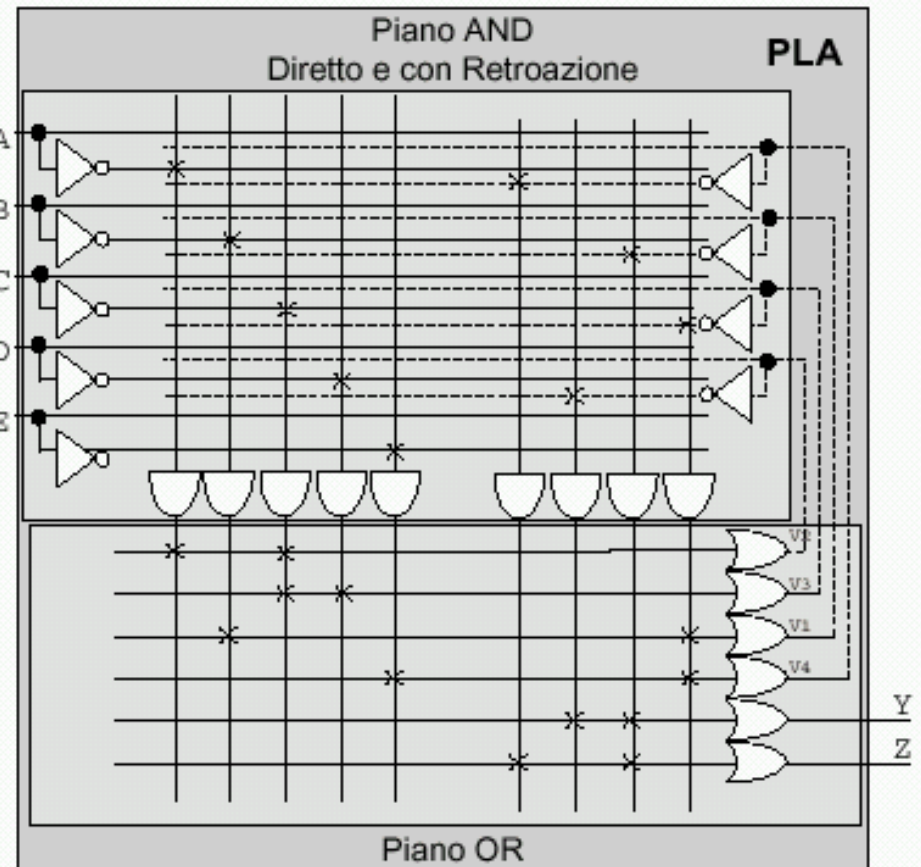
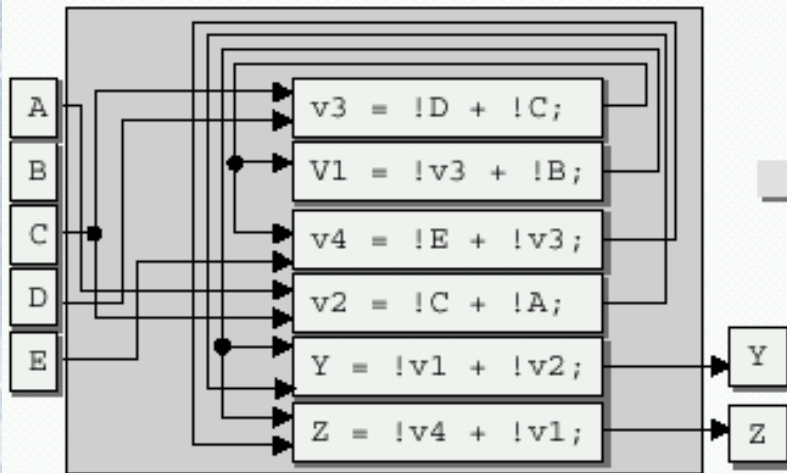
- Esempio di implementazione di una rete combinatoria a più livelli grazie alla retroazione



# PLA e PAL avanzate

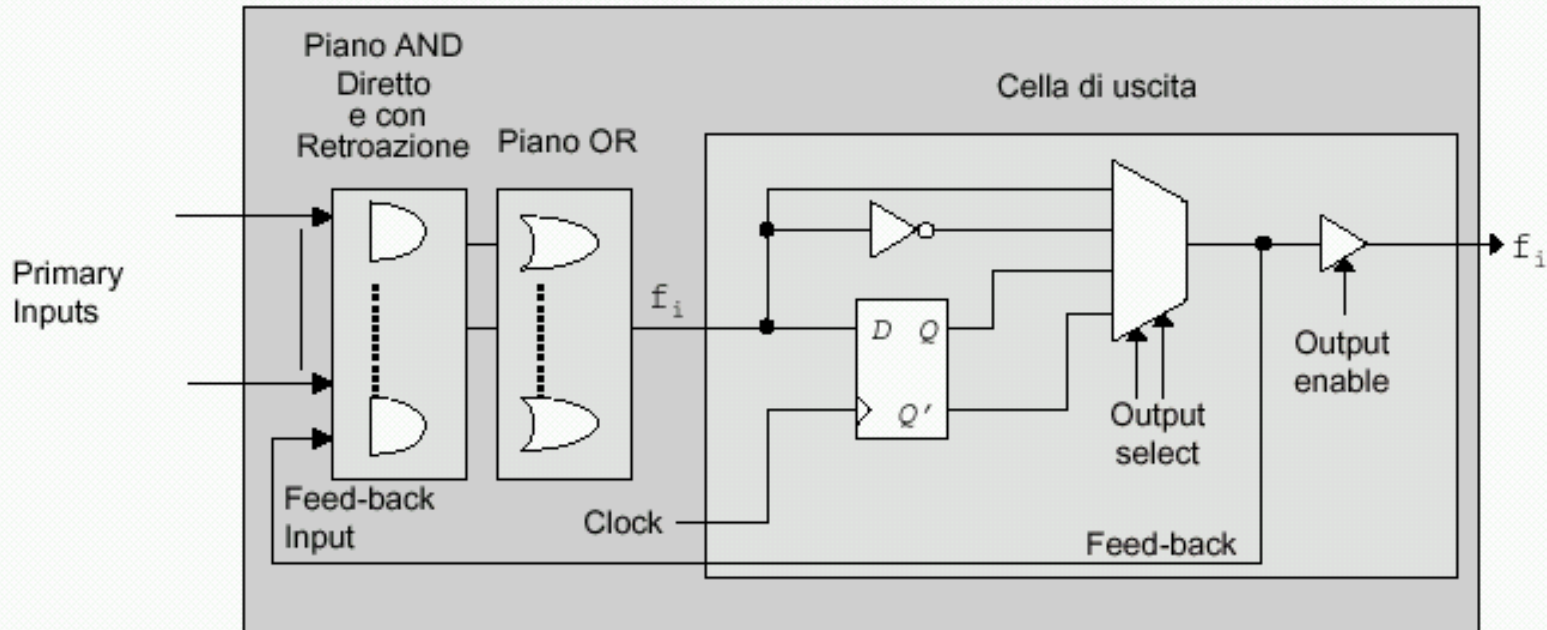
## ■ Esempio: realizzazione

□ Esempio (cont):



# PLA e PAL avanzate

- L'aggiunta di elementi di memoria in uscita possono ulteriormente ampliare le prestazioni del dispositivo

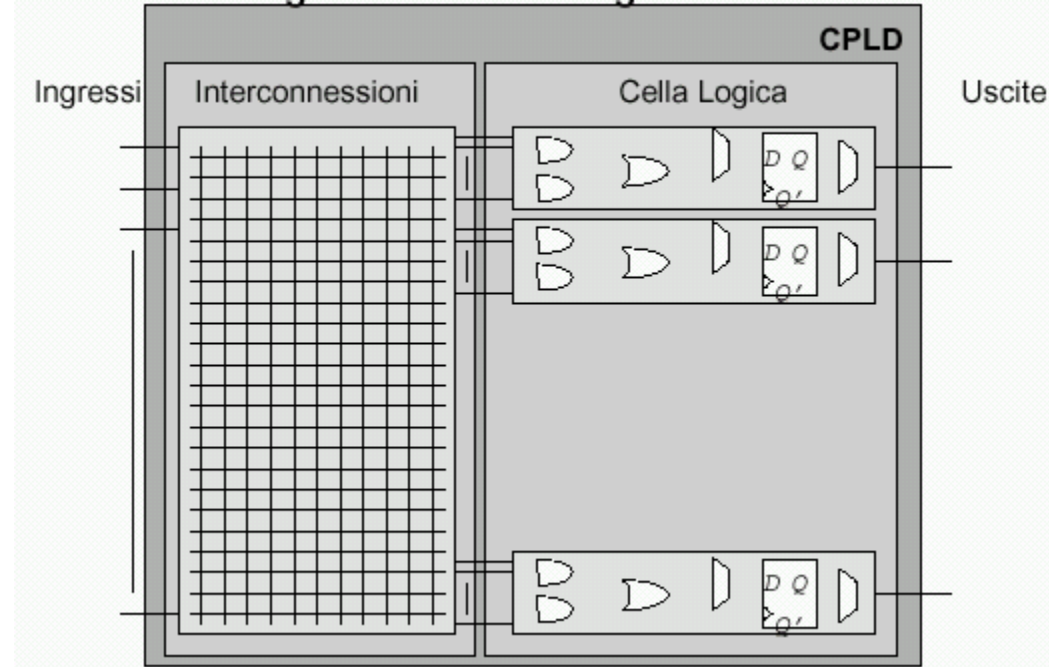




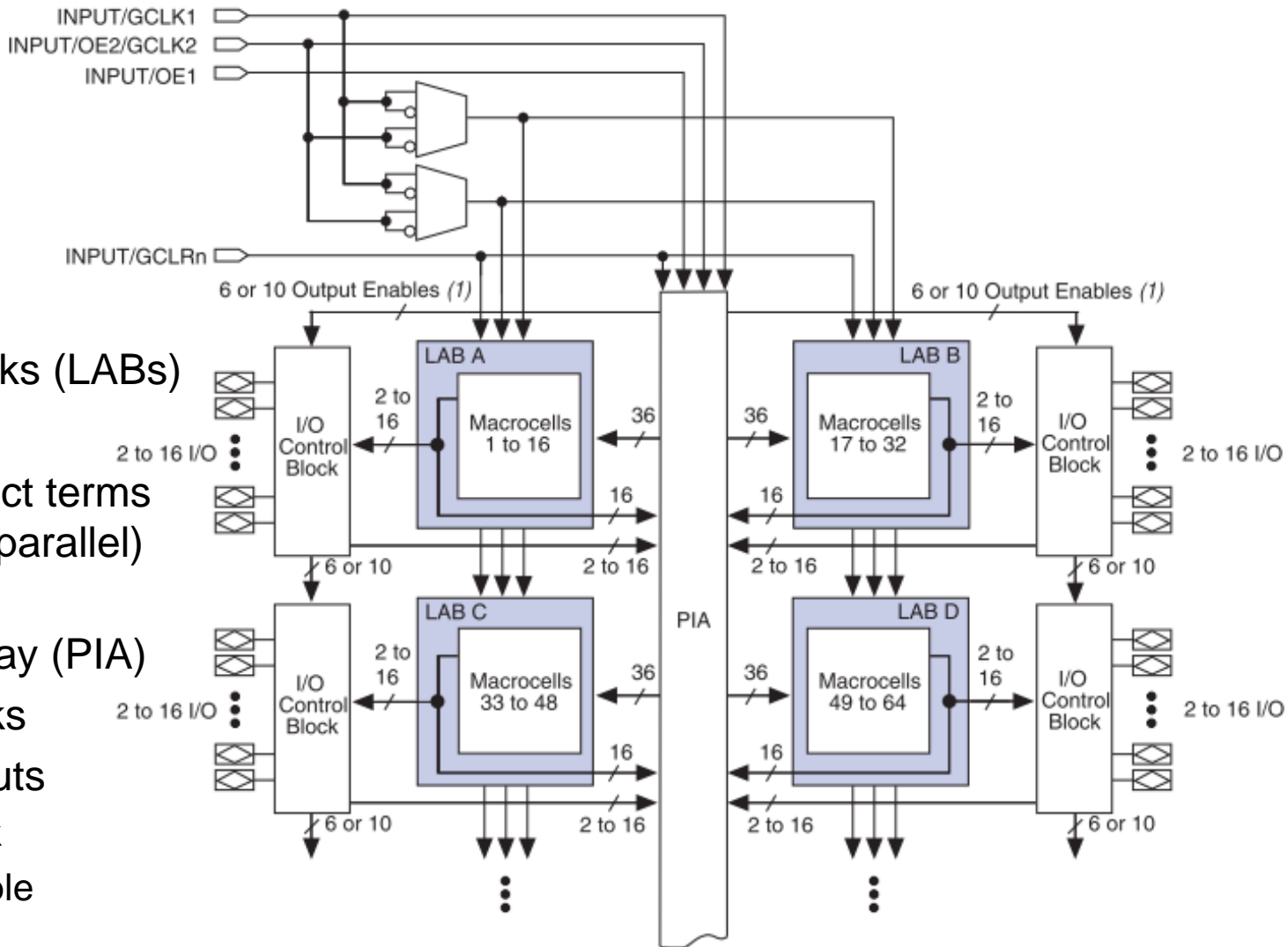
# CPLD

## Complex Programmable Logic Device

- Sono la logica evoluzione di PAL e PLA
- Sono caratterizzati da:
  - Connessioni globali
  - Logica Concentrata
- Rispetto PAL e PLA
  - Sono più complessi e hanno dimensioni maggiori
  - Consentono di ottenere prestazioni più elevate



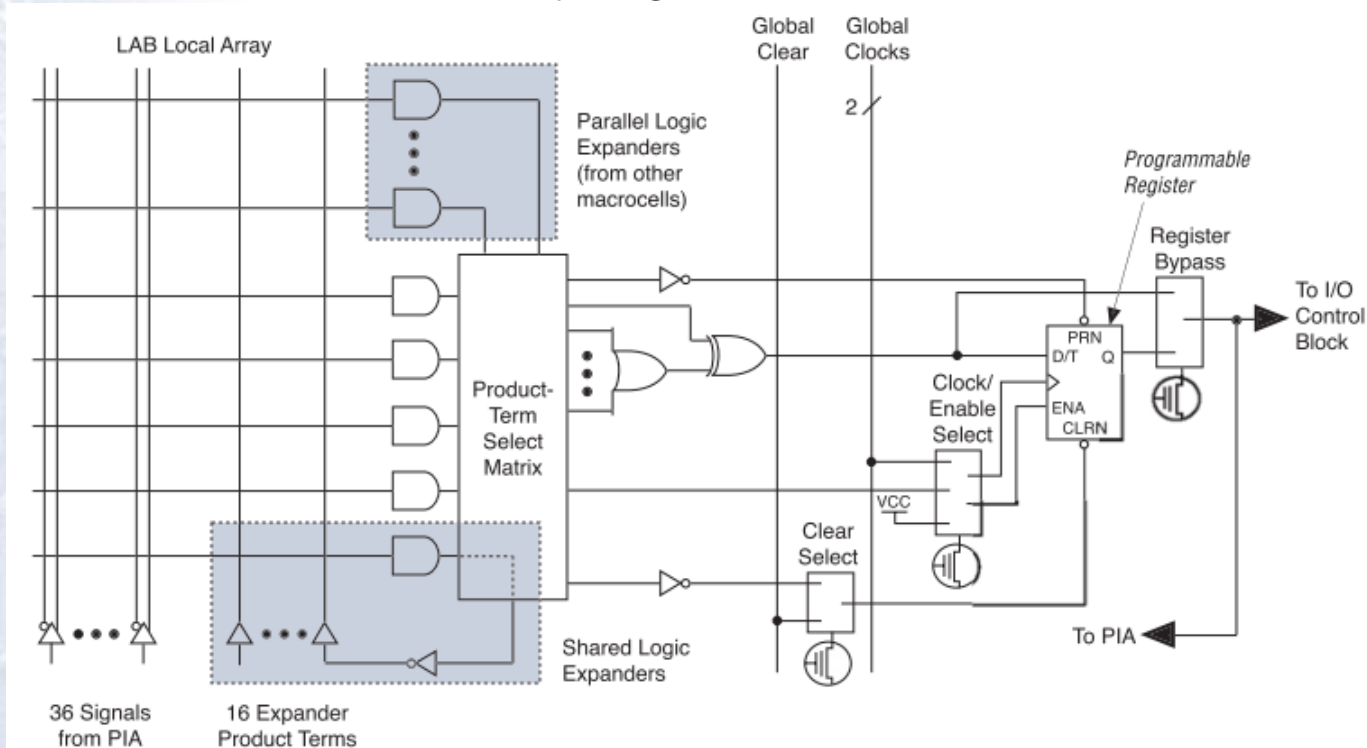
# CPLD – MAX 3000 A



- Logic array blocks (LABs)
- Macrocells
- Expander product terms (shareable and parallel)
- Programmable interconnect array (PIA)
- I/O control blocks
- 4 dedicated Inputs
  - 2 global clock
  - 1 output enable
  - 1 global clear

# Max 3000 – Logic Array Blocks

- 16 macrocells
  - logic array (5 minterms each)
  - product–term select matrix
  - programmable register
- 36 signals from the PIA and 16 from other macrocells as logic inputs
- Global controls for secondary register functions

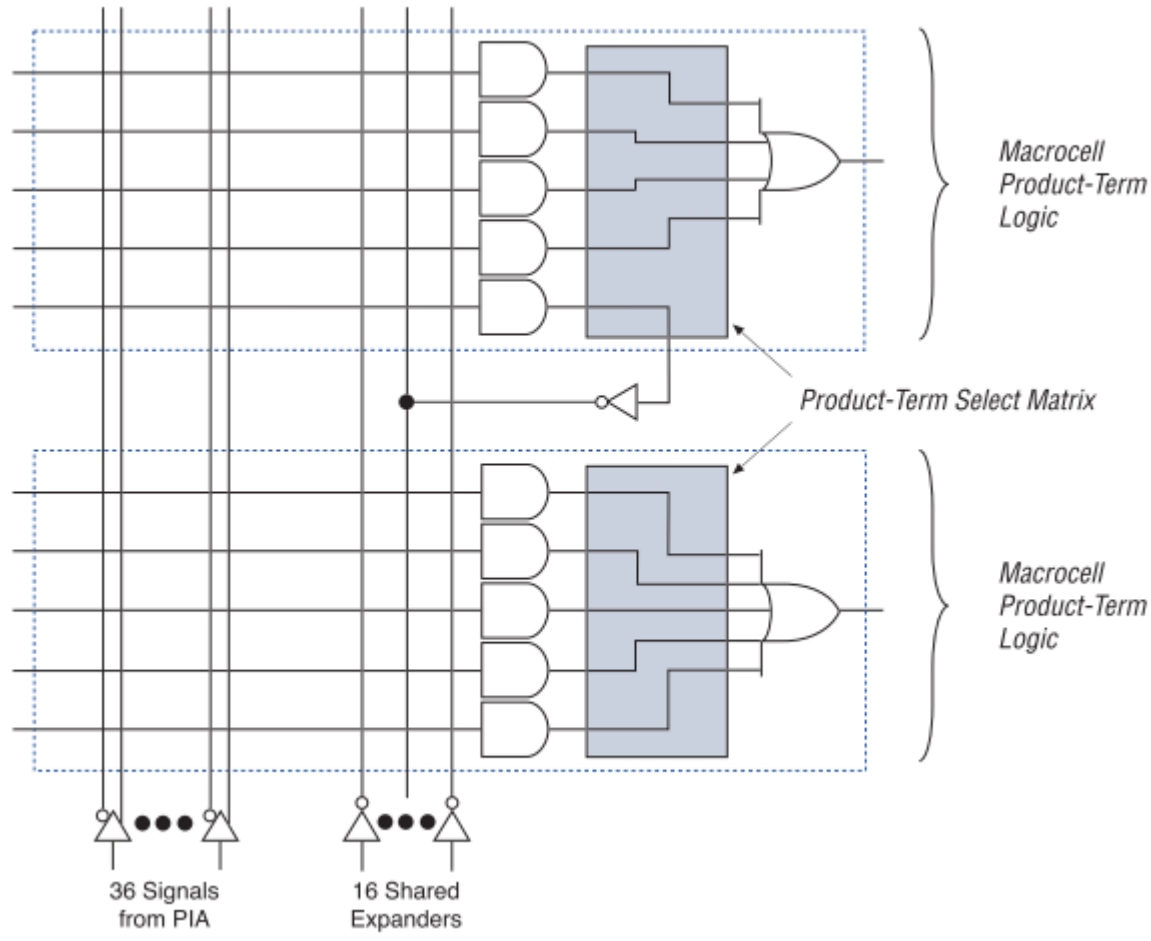


# MAX 3000 – PTSM & expansions

- Product Term Select Matrix
  - Seleziona come impiegare i vari segnali
    - Per realizzare funzioni logiche (somme di prodotti)
    - Come segnali di controllo per il FF
- Shareable logic expansion
  - Consente una retroazione
  - Il risultato può essere impiegato da altre macrocelle
  - Introduce un certo ritardo
- Parallel Logic Expansion
  - Consente di sviluppare funzioni logiche più complesse sfruttando da macrocelle adiacenti
  - Introduce un certo ritardo

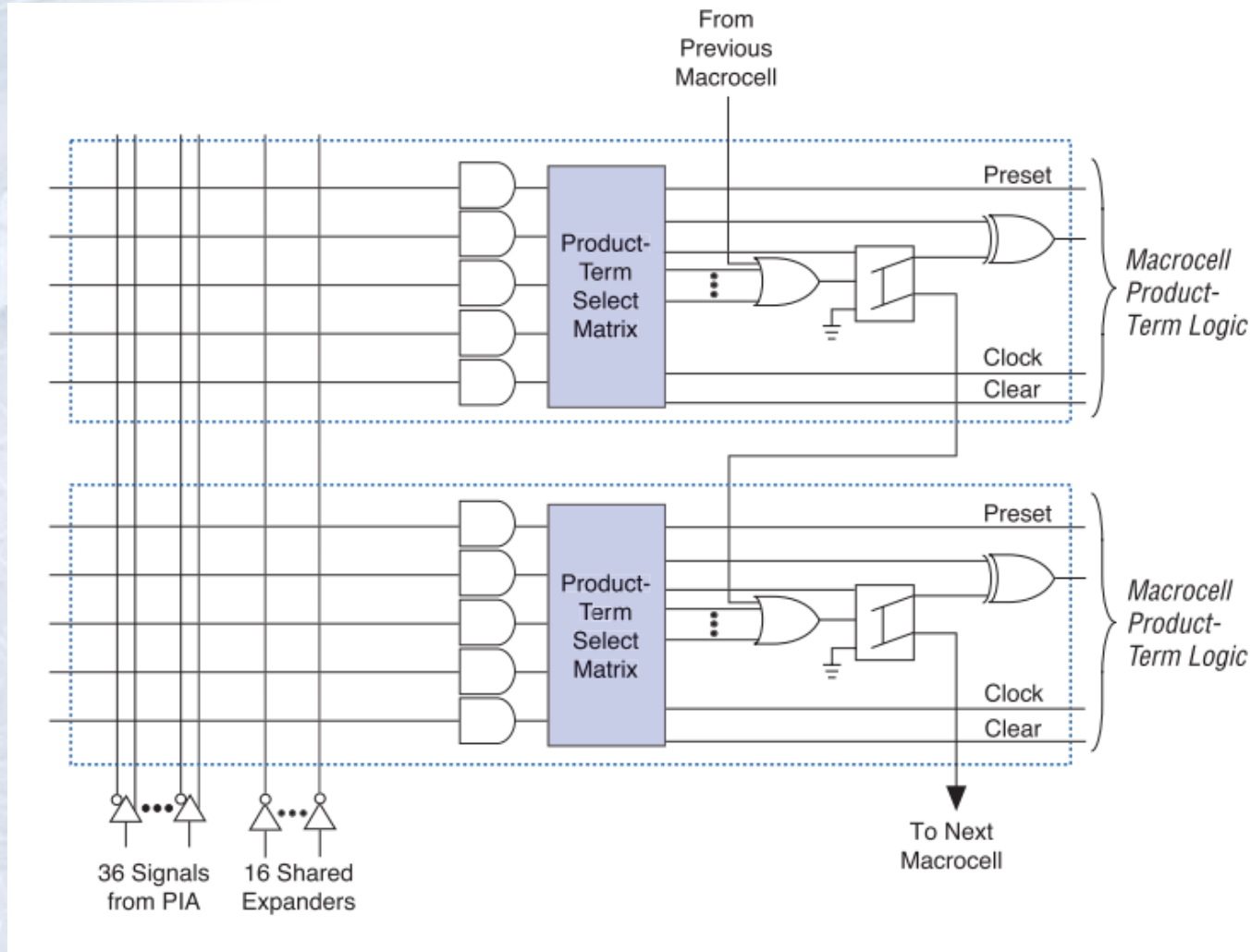
# MAX 3000 – Shareable expansions

Shareable expanders can be shared by any or all macrocells in an LAB.





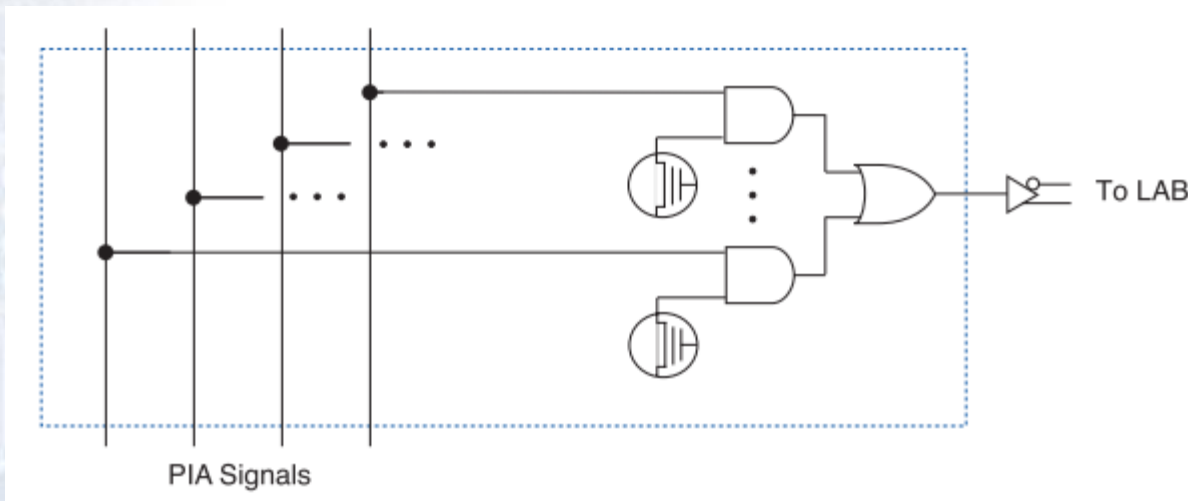
# MAX 3000 – Parallel expansions



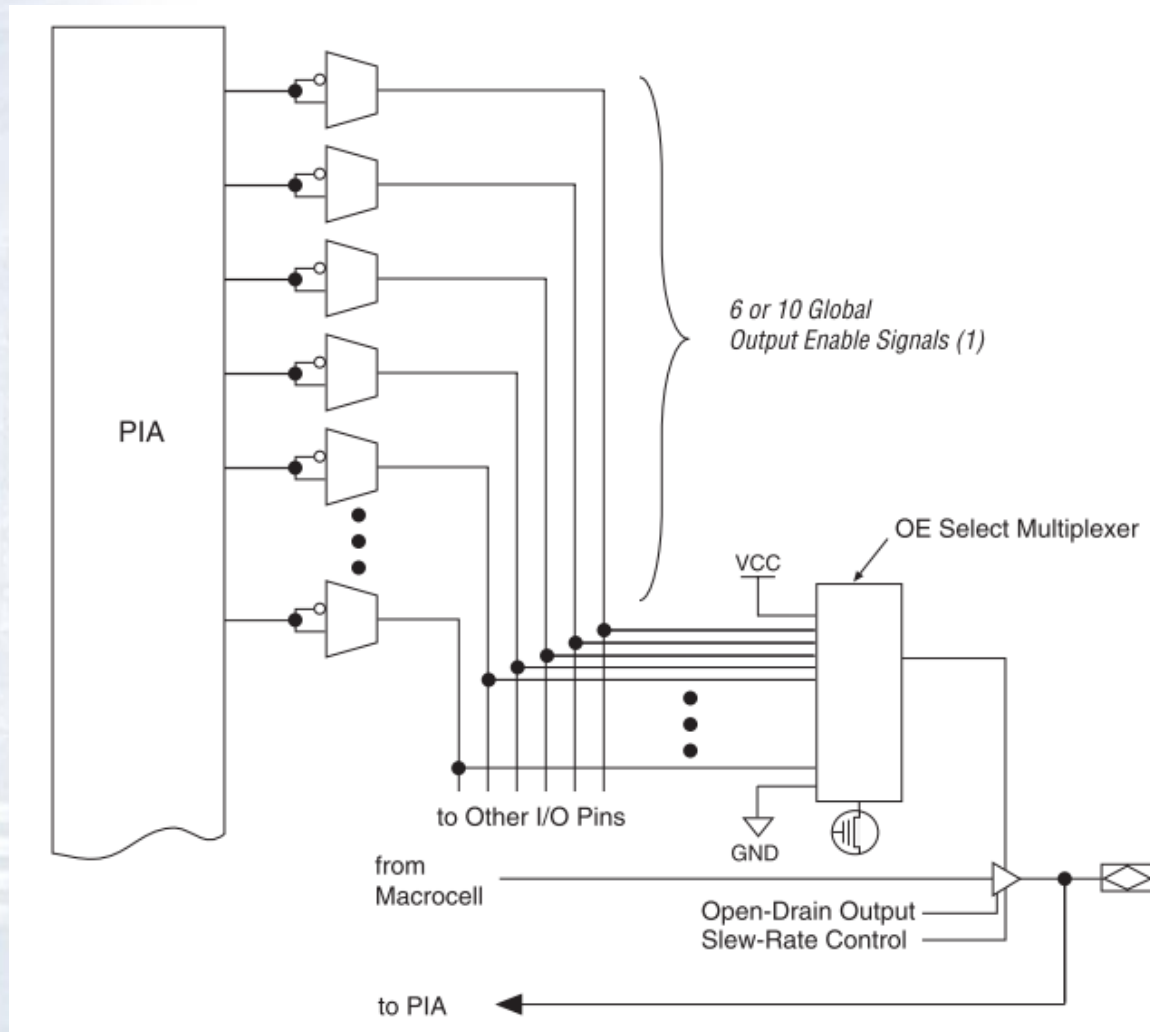
# MAX 3000 - PIA

## ■ Programmable Interconnected Array

- Il PIA è un global bus programmabile che provvede a rendere disponibili all'interno del dispositivo tutti i segnali che lo alimentano, che si possono suddividere in :
  - ingressi dedicati del MAX3000,
  - pin di I/O
  - uscite delle macrocelle
- Il ritardo di questi segnali è prevedibile



# MAX 3000 – I/O Control Blocks



# MAX 3000 – I/O Control Blocks

- I PIN di I/O possono essere configurati
  - Come , ingressi, uscite o bidirezionali
  - Possono essere posti in alta impedenza
    - Attraverso linee dedicate (dirette o negate) di OE (che derivano dal PIA)
    - Tramite altri PIN di I/O
    - Attraverso le uscite di macrocelle
    - Possono essere posti perennemente in alta impedenza
    - Possono essere posti perennemente come buffer
  - E' previsto un controllo dello slew-rate

# MAX 3000 – Product Family

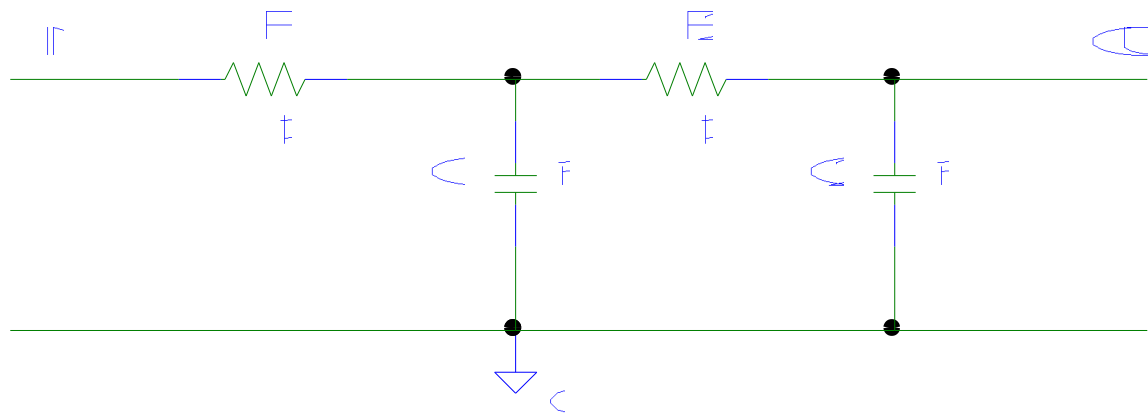
**Table 1. MAX 3000A Device Features**

Feature	EPM3032A	EPM3064A	EPM3128A	EPM3256A	EPM3512A
Usable gates	600	1,250	2,500	5,000	10,000
Macrocells	32	64	128	256	512
Logic array blocks	2	4	8	16	32
Maximum user I/O pins	34	66	98	161	208
$t_{PD}$ (ns)	4.5	4.5	5.0	7.5	7.5
$t_{SU}$ (ns)	2.9	2.8	3.3	5.2	5.6
$t_{CO1}$ (ns)	3.0	3.1	3.4	4.8	4.7
$f_{CNT}$ (MHz)	227.3	222.2	192.3	126.6	116.3



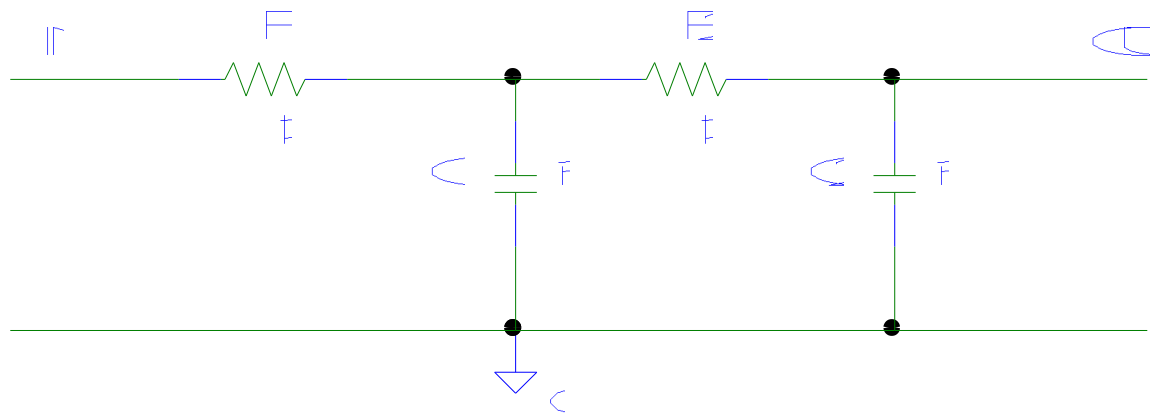
# MAX3000 - Q e A

Q: E' possibile realizzare tramite CPLD MAX3000 la seguente architettura?



# MAX 3000- Q e A

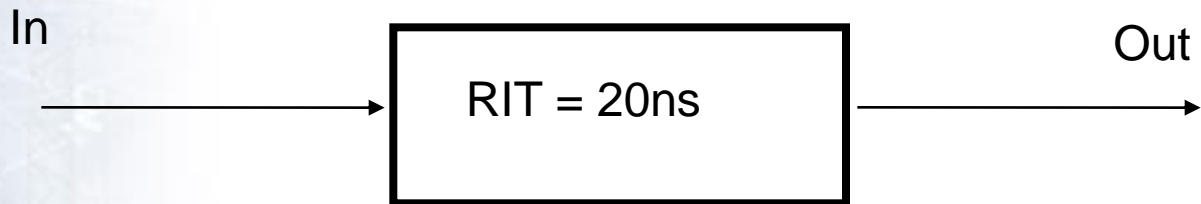
Q: E' possibile realizzare tramite CPLD MAX3000 la seguente architettura?



A: **NO** con una CPLD si possono realizzare solo circuiti **DIGITALI**

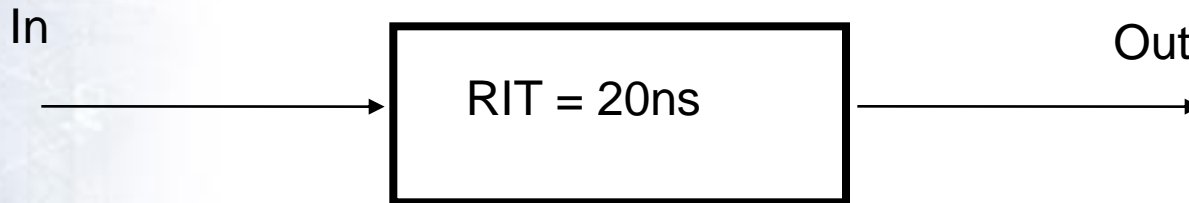
# MAX3000 - Q e A

Q: E' possibile realizzare tramite CPLD MAX3000 la seguente funzione, ossia un predeterminato RITARDO asincrono ?



# MAX3000 - Q e A

Q: E' possibile realizzare tramite CPLD MAX3000 la seguente funzione, ossia un predeterminato RITARDO asincrono ?



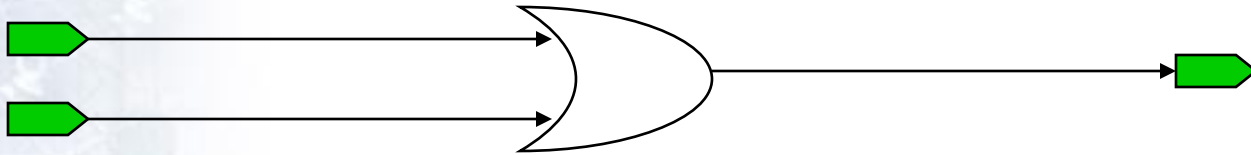
**A: NO: non vi e' alcun modo o alcun elemento che possa realizzare questa funzione.**

**Nota1:** nei circuiti digitali il ritardo e' una conseguenza (indesiderata) della struttura stessa del circuito e non un parametro da soddisfare

**Nota2:** Un elemento di ritardo e' tuttavia realizzabile in modo SINCRONO

# MAX3000- Q e A

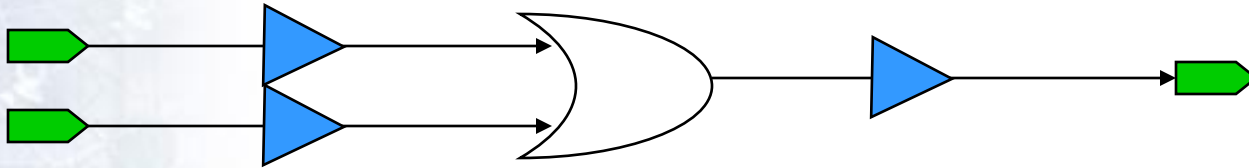
Q: E' possibile realizzare tramite CPLD MAX3000 la seguente funzione logica (gli ingressi e le uscite siano collegati direttamente ai pin di I/O del dispositivo)?





# MAX3000- Q e A

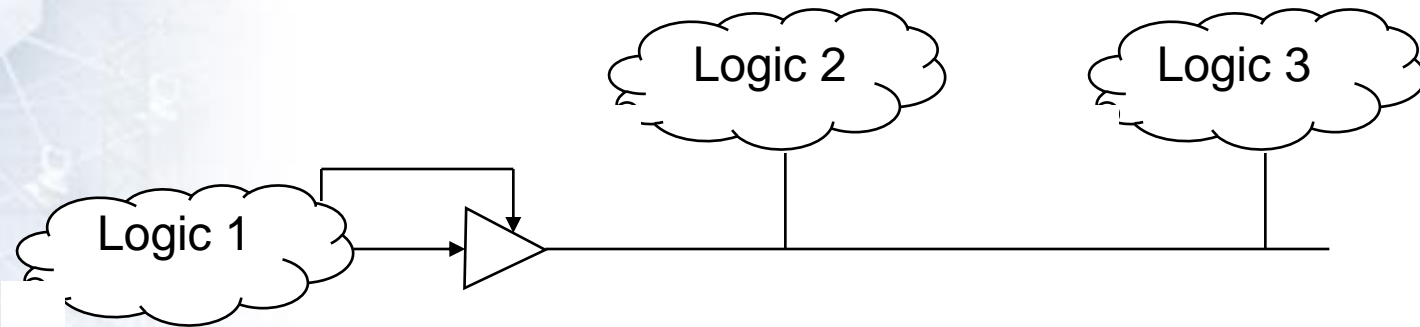
Q: E' possibile realizzare tramite CPLD MAX3000 la seguente funzione logica (gli ingressi e le uscite siano collegati direttamente ai pin di I/O del dispositivo)?



A: NO: Non esiste alcun collegamento diretto tra i pin e la logica interna, ma bisogna passare attraverso ai buffer di I/O (il tool di sviluppo tipicamente corregge automaticamente questo tipo di errore)

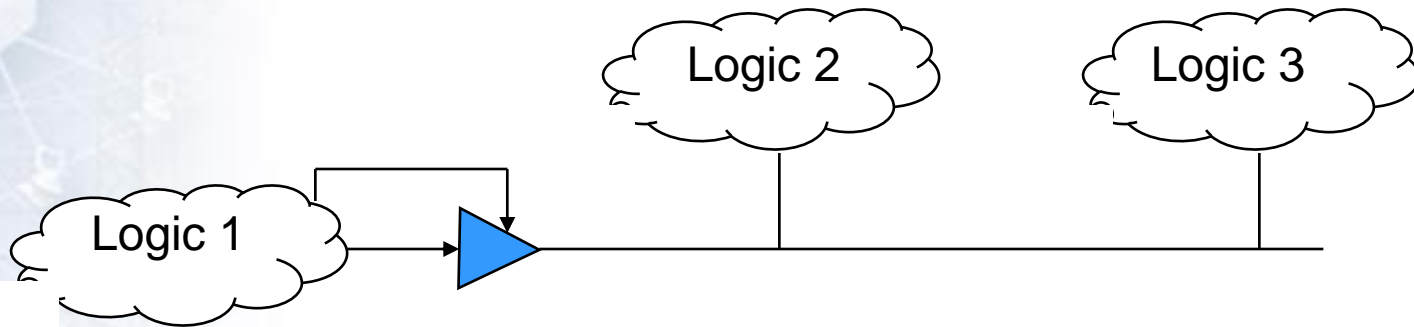
# MAX3000 - Q e A

Q: E' possibile realizzare tramite CPLD MAX3000 la seguente funzione (una logica pilota altre logiche sfruttando un bus tri-state)



# MAX3000 - Q e A

Q: E' possibile realizzare tramite CPLD MAX3000 la seguente funzione (una logica pilota altre logiche struttando un bus tri-state)

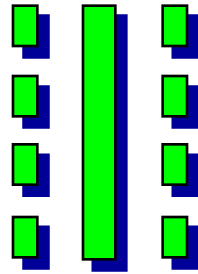


A: **NO** ! Gli unici buffer tri-state del dispositivo sono disponibili nei blocchi di I/O e pertanto non possono essere usati per pilotare logiche interne al dispositivo stesso! (un eventuale loop dal pin di uscita dovrebbe passare attraverso un buffer di ingresso che ne annullerebbe l'effetto)

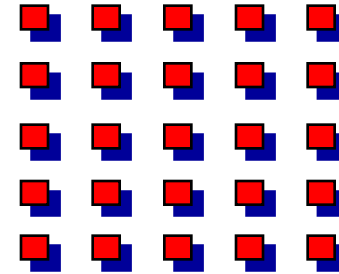
# FPGA: Introduzione

- Le FPGA (Field Programmable Gate Array) sono dispositivi programmabili costituiti da una matrice di componenti logici collegabili tra loro

**Complex Programmable Logic Device (CPLD)**



**Field-Programmable Gate Array (FPGA)**



**Architecture**

**PAL/22V10-like  
More Combinational**

**Gate array-like  
More Registers + RAM**

**Density**

**Low-to-medium  
0.5-10K logic gates**

**Medium-to-high  
1K to 3.2M system gates**

**Performance**

**Predictable timing  
Up to 250 MHz today**

**Application dependent  
Up to 200 MHz today**

**Interconnect**

**“Crossbar Switch”**

**Incremental**

# FPGA

- Le FPGA mettono a disposizione dell'utente
  - Componenti logici (CLB - Slice) costituiti da
    - logica, piccole memorie, flip-flop, buffer, multiplexer.
  - Linee di connessione
    - sia locali (corte) che distribuite (lunghe)
  - Matrici di inter-connessione
    - per collegare varie linee tra loro e da queste ai blocchi logici
  - Blocchi di I/O
    - particolari blocchi logici dedicati all'I/O provvedono Buffer, protezioni, Fan-out, resistenze di pull-up e pull-down, adattatori d'impedenza, ...
  - Blocchi particolari
    - memorie, moltiplicatori, PLL, decodificatori, ...



# FPGA

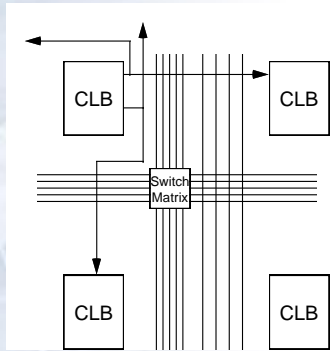
## ■ Pregi e difetti:

- Estremamente versatili
- Elevata complessità computazionale
- Più lente di CPLD e ASIC
- Costo elevato per singolo componente (ma esistono famiglie particolarmente economiche)
- Costo del prototipo ridotto
- “Time to market” molto ridotto
- Possibilità di “upgrade” del circuito (anche a distanza)
- Ottime per la realizzazione di prototipi, (ma si usano sempre di più anche negli elevati volumi di fabbricazione)
- Capacità di supportare sistemi interni (embedded systems)

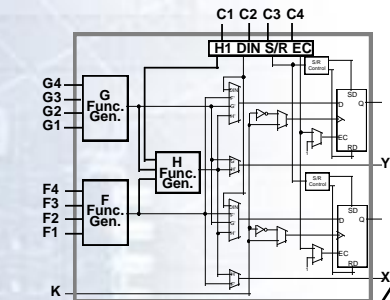
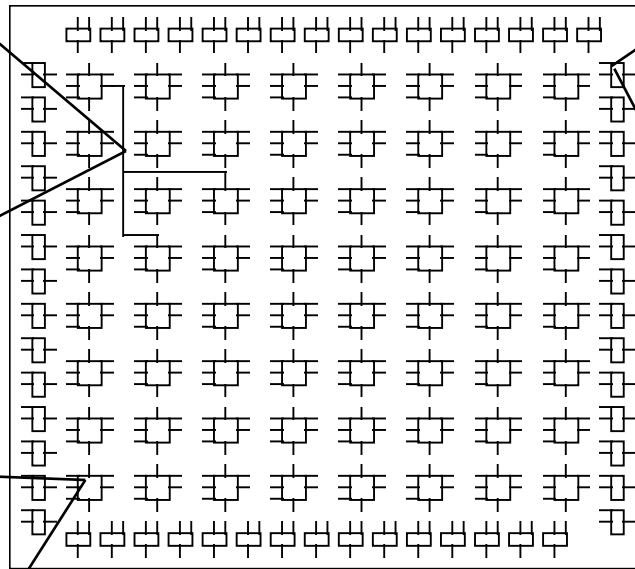
# XC4000 Architecture and Features

*Architettura Xilinx (1995)*

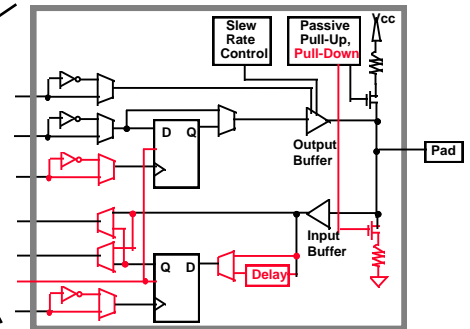
# XC4000 Architecture



**Programmable Interconnect**



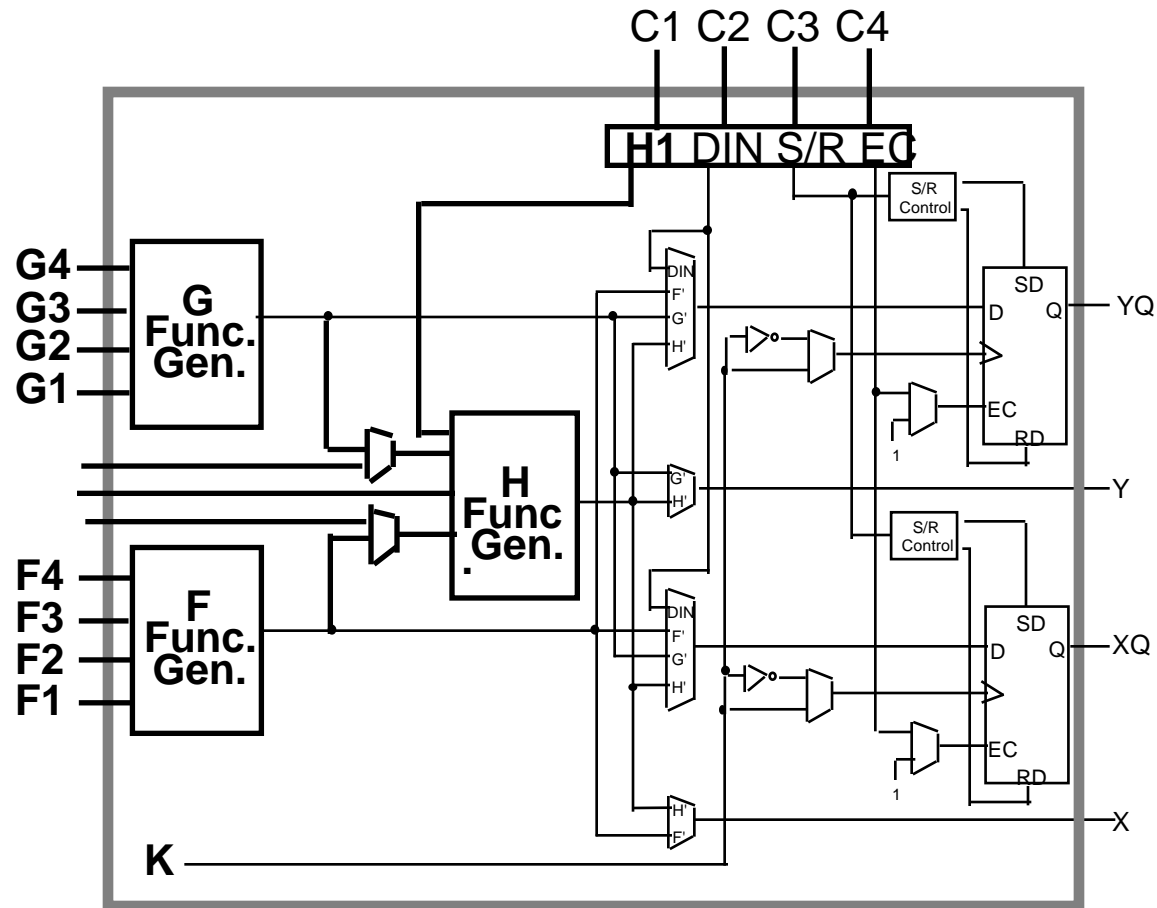
**Configurable Logic Blocks (CLBs)**



**I/O Blocks (IOBs)**

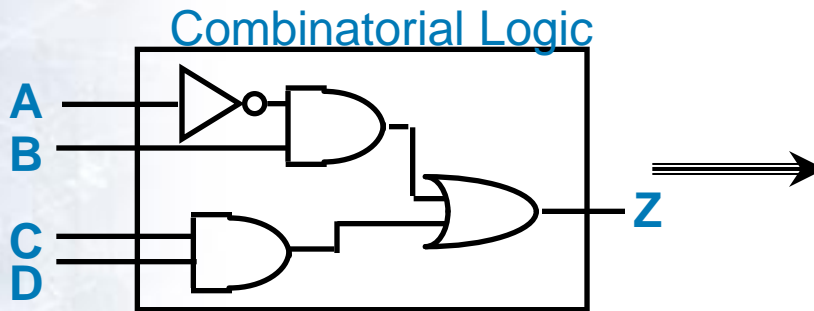
# XC4000E/X Configurable Logic Blocks

- 2 Four-input function generators (Look Up Tables)
  - 16x1 RAM or Logic function
- 2 Registers
  - Each can be configured as Flip Flop or Latch
  - Independent clock polarity
  - Synchronous and asynchronous Set/Reset



# Look Up Tables

- Combinatorial Logic is stored in 16x1 SRAM Look Up Tables (LUTs) in a CLB
- Example:

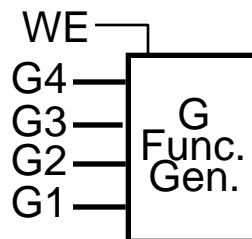


Look Up Table  
4-bit address

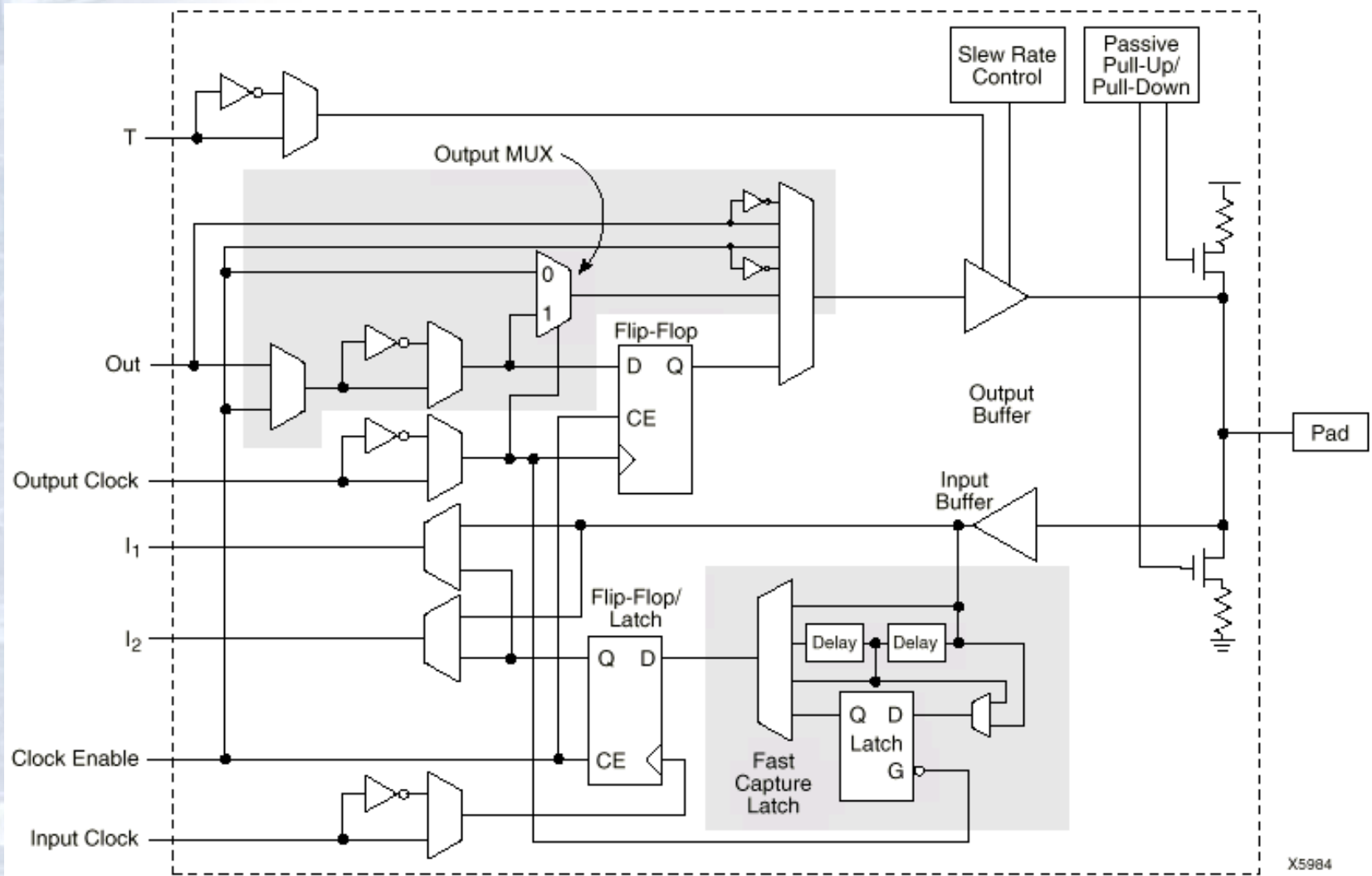
A	B	C	D	Z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

$$2^{(2^4)} = 64K !$$

- Capacity is limited by number of inputs, not complexity
- Choose to use each function generator as 4 input logic (LUT) or as high speed sync.dual port RAM



# XC4000X I/O Block Diagram



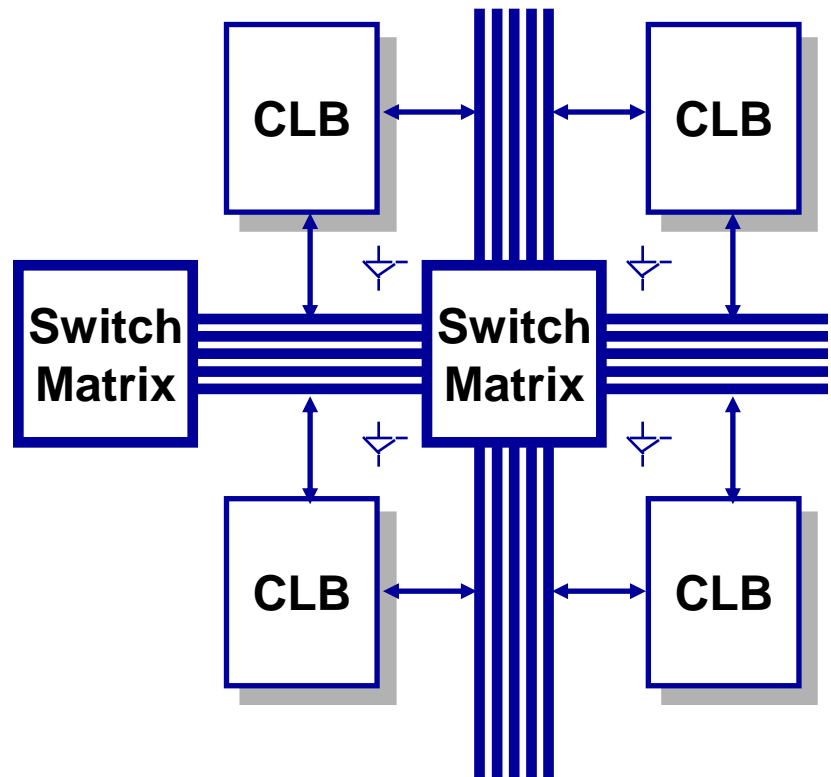
X5984

Shaded areas are not included in XC4000E family.



# Xilinx FPGA Routing

- 1) Fast Direct Interconnect - CLB to CLB
- 2) General Purpose Interconnect - Uses switch matrix
- 3) Double Lines
- 4) Long Lines  
Segmented across chip  
Global clocks, lowest skew  
2 Tri-states per CLB for busses



# What's Really In that Chip?

Programmable Interconnect Points, PIPs (White)

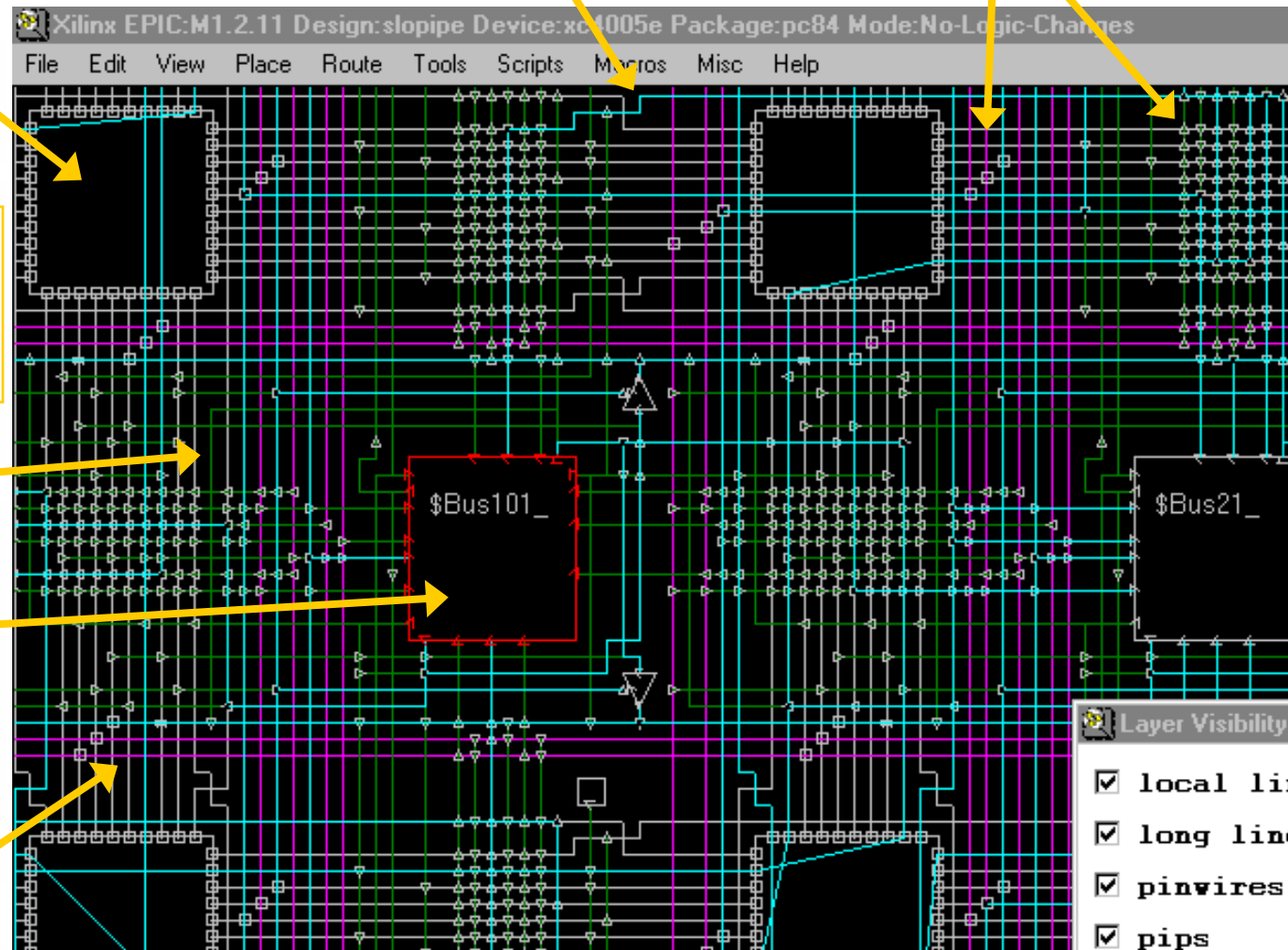
Switch Matrix

Routed Wires (Blue)

Direct Interconnect (Green)

CLB (Red)

Long Lines (Purple)



# Altera Cyclone II

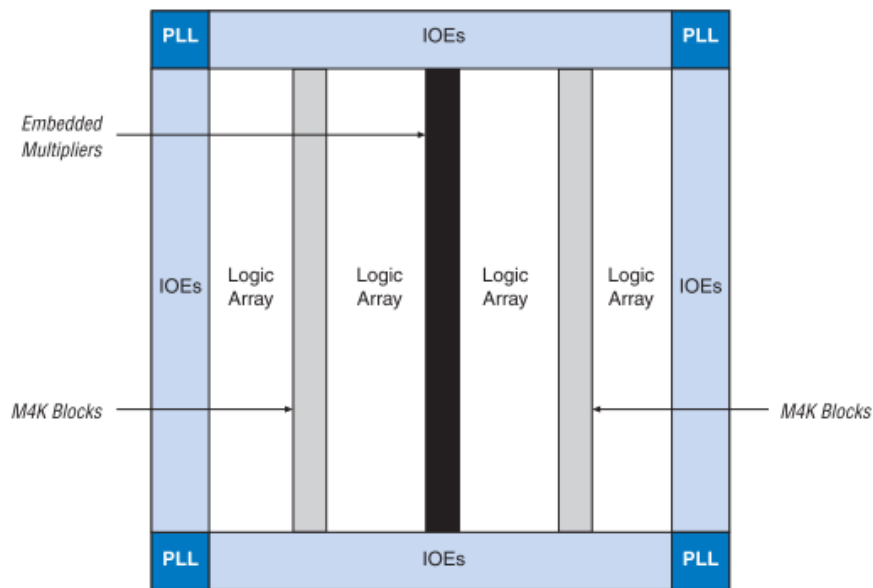
# Caratteristiche

- High-density architecture with 4,608 to 68,416 Les
- M4K embedded memory blocks
- Embedded multipliers 18x18
- Advanced I/O support
- Flexible clock management circuitry

*Table 1-1. Cyclone II FPGA Family Features*

Feature	EP2C5	EP2C8	EP2C20	EP2C35	EP2C50	EP2C70
LEs	4,608	8,256	18,752	33,216	50,528	68,416
M4K RAM blocks (4 Kbits plus 512 parity bits)	26	36	52	105	129	250
Total RAM bits	119,808	165,888	239,616	483,840	594,432	1,152,000
Embedded multipliers (1)	13	18	26	35	86	150
PLLs	2	2	4	4	4	4
Maximum user I/O pins	158	182	315	475	450	622

# Cyclone II Internals

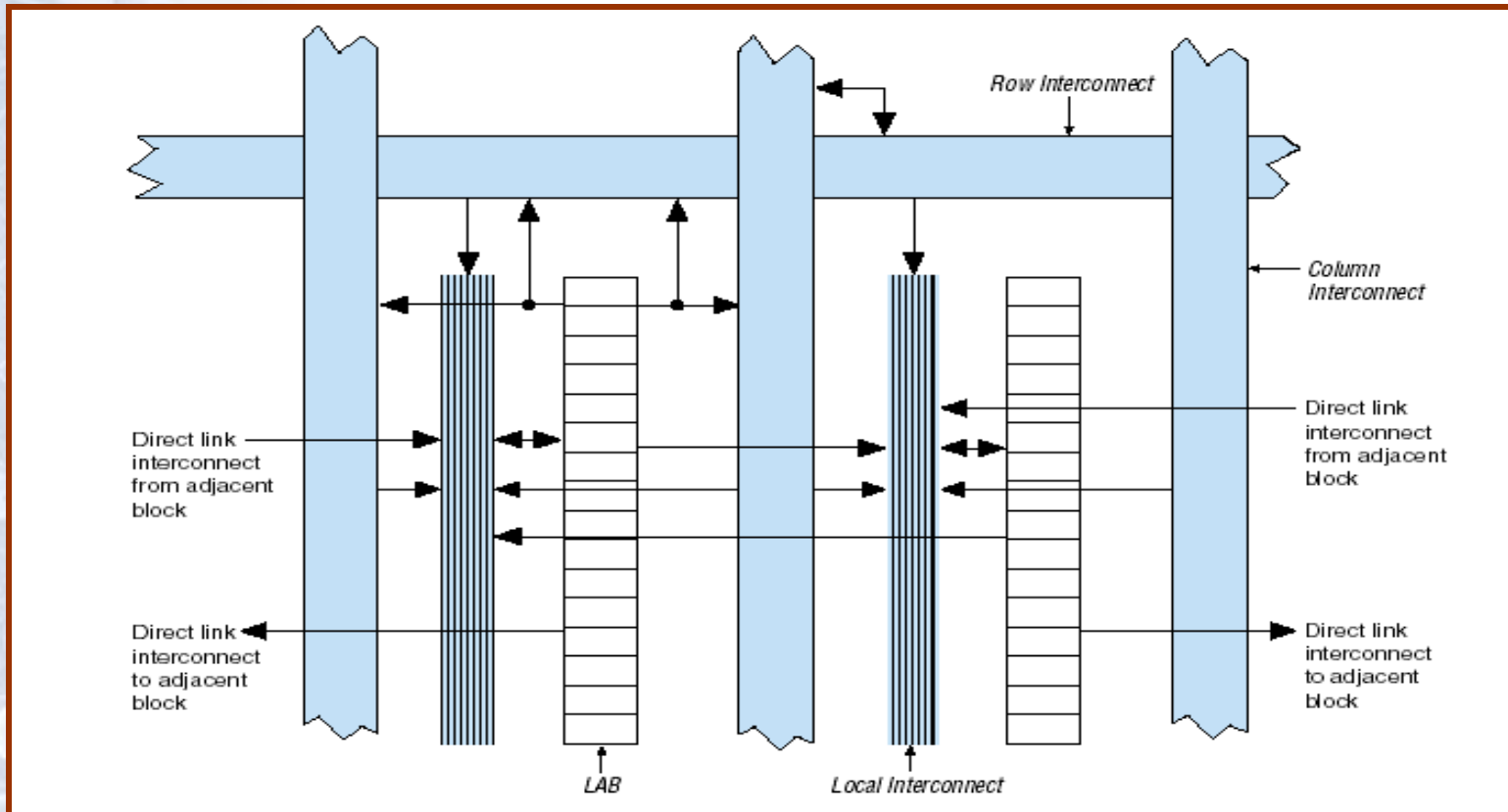


**Table 2-1. Cyclone II Device Resources**

Device	LAB Columns	LAB Rows	LEs	PLLs	M4K Memory Blocks	Embedded Multiplier Blocks
EP2C5	24	13	4,608	2	26	13
EP2C8	30	18	8,256	2	36	18
EP2C20	46	26	18,752	4	52	26
EP2C35	60	35	33,216	4	105	35
EP2C50	74	43	50,528	4	129	86
EP2C70	86	50	68,416	4	250	150

# Cyclone II Logic Array

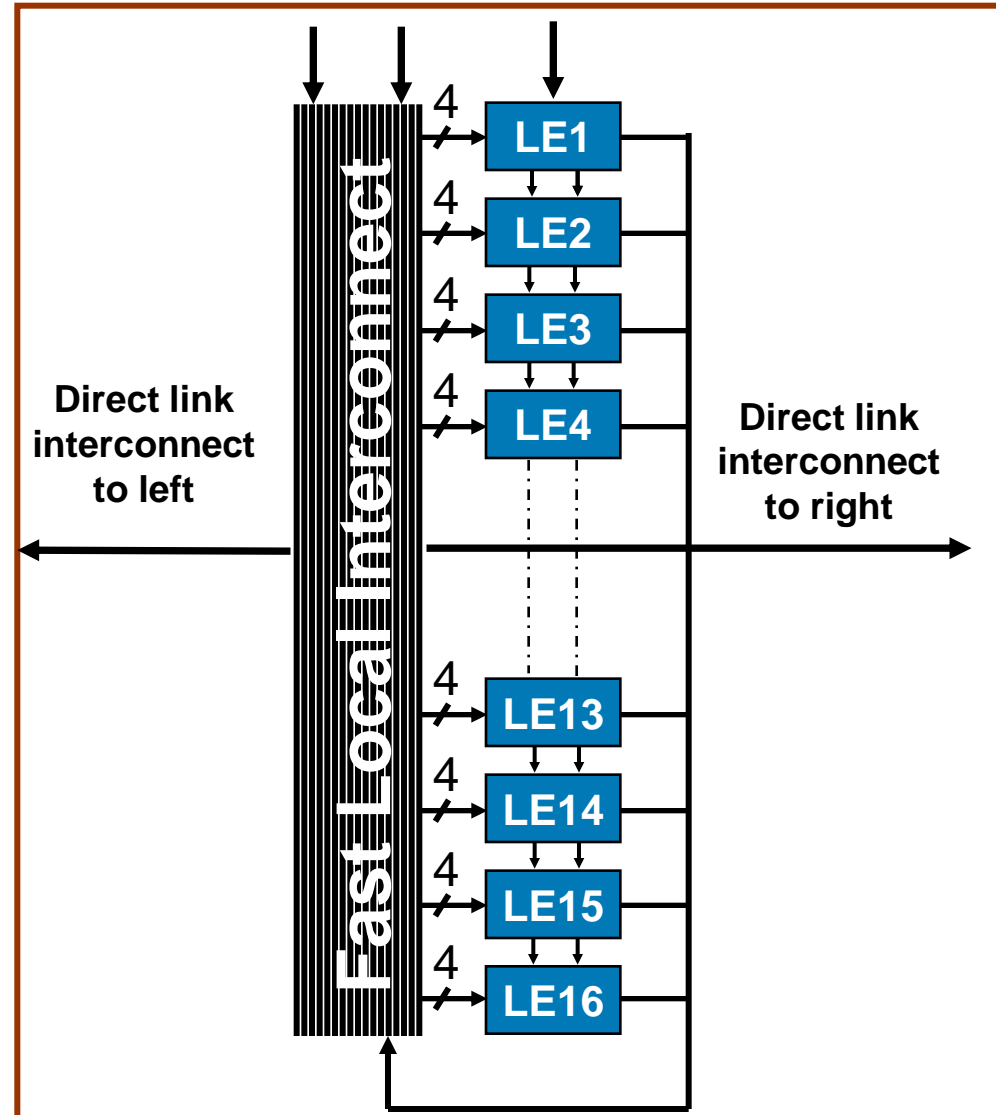
- LAB (logic array blocks)
- Interconnections





# Cyclone II Logic Array Block (LAB)

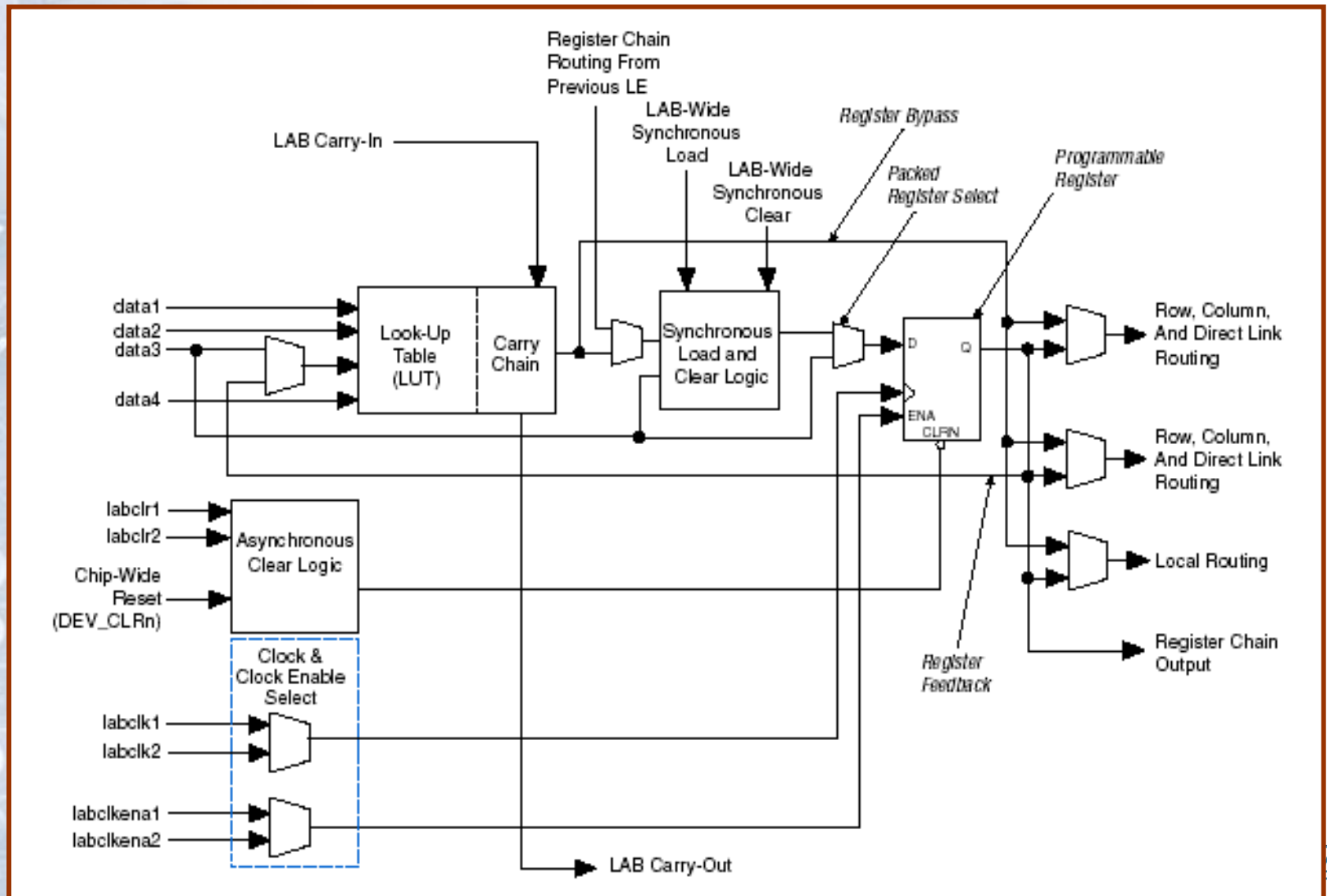
- 16 LEs
- Local Interconnect
- LE carry chains
- Register chains
- LAB Control Signals
  - 2 CLK
  - 2 CLK ENA
  - 2 ACLR
  - 1 SCLR
  - 1 SLOAD
- Device Global Reset
  - DEV\_CLRn



# Cyclone II - LE

- A four-input look-up table (LUT), which is a function generator that can implement any function of four variables
- A programmable register
- A carry chain connection
- A register chain connection
- The ability to drive all types of interconnects: local, row, column, register chain, and direct link interconnects
- Support for register packing
- Support for register feedback

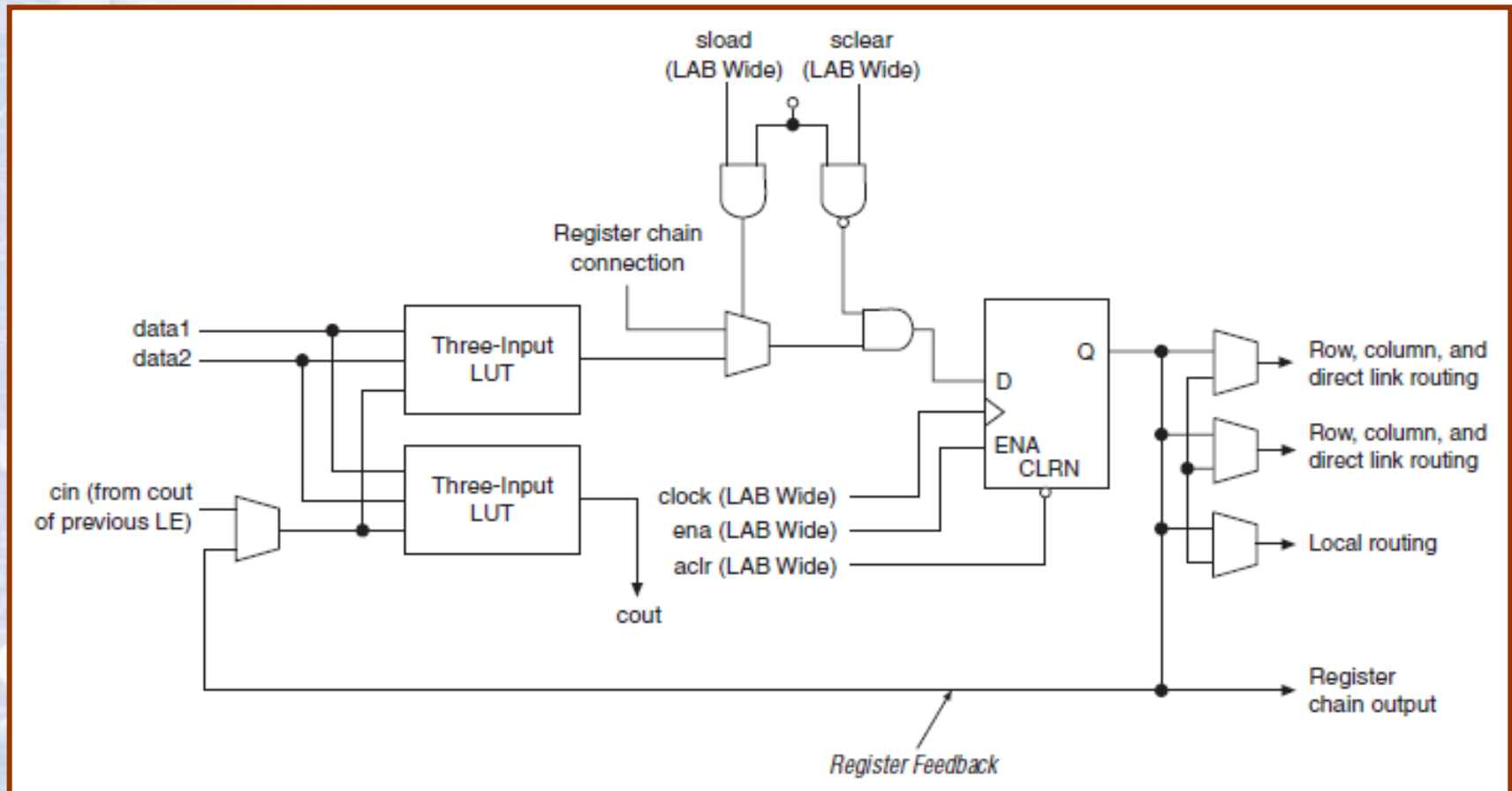
# Cyclone II Logic Element (LE)





# LE in Arithmetic Mode

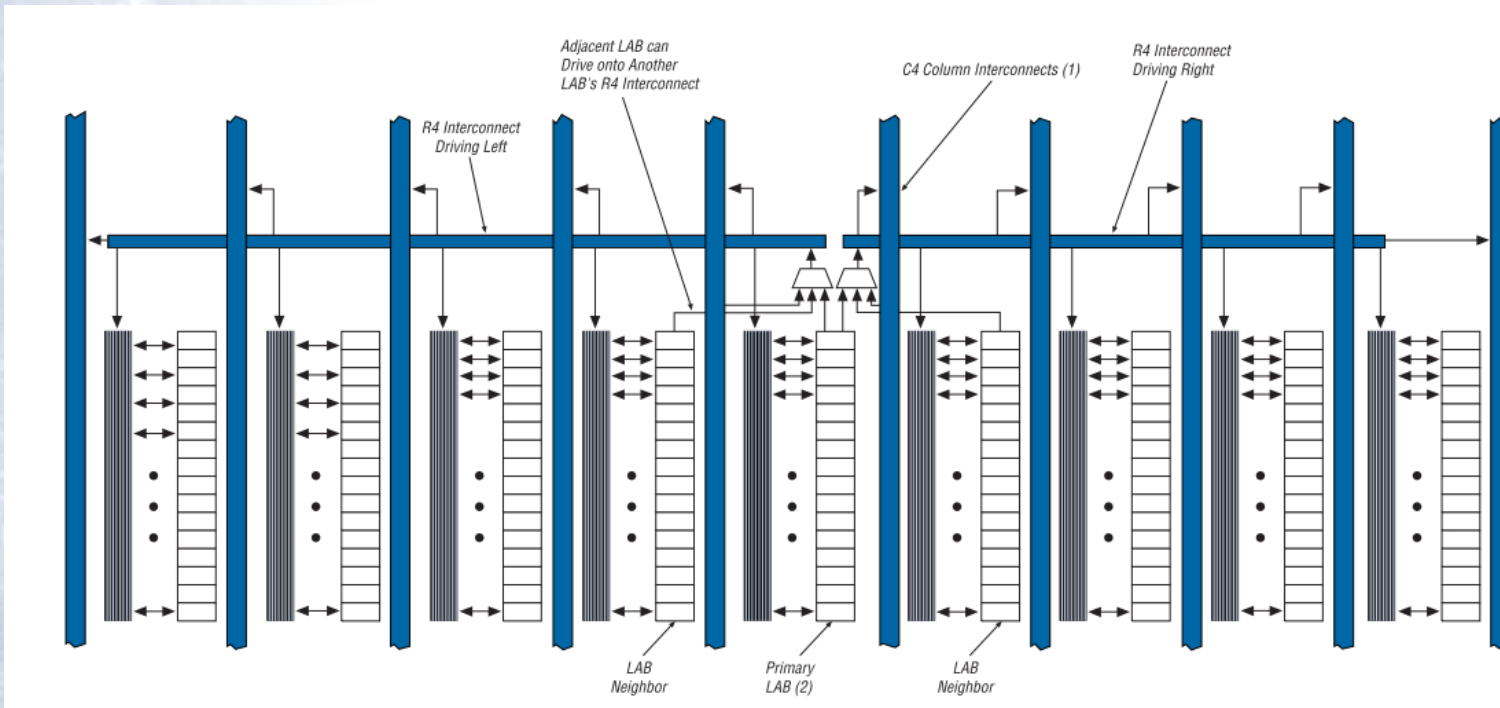
- Ideal for implementing adders, counters, accumulators, and comparators.



# Multitrack Interconnections

Connections between LEs, M4K, multipliers, device I/O pins

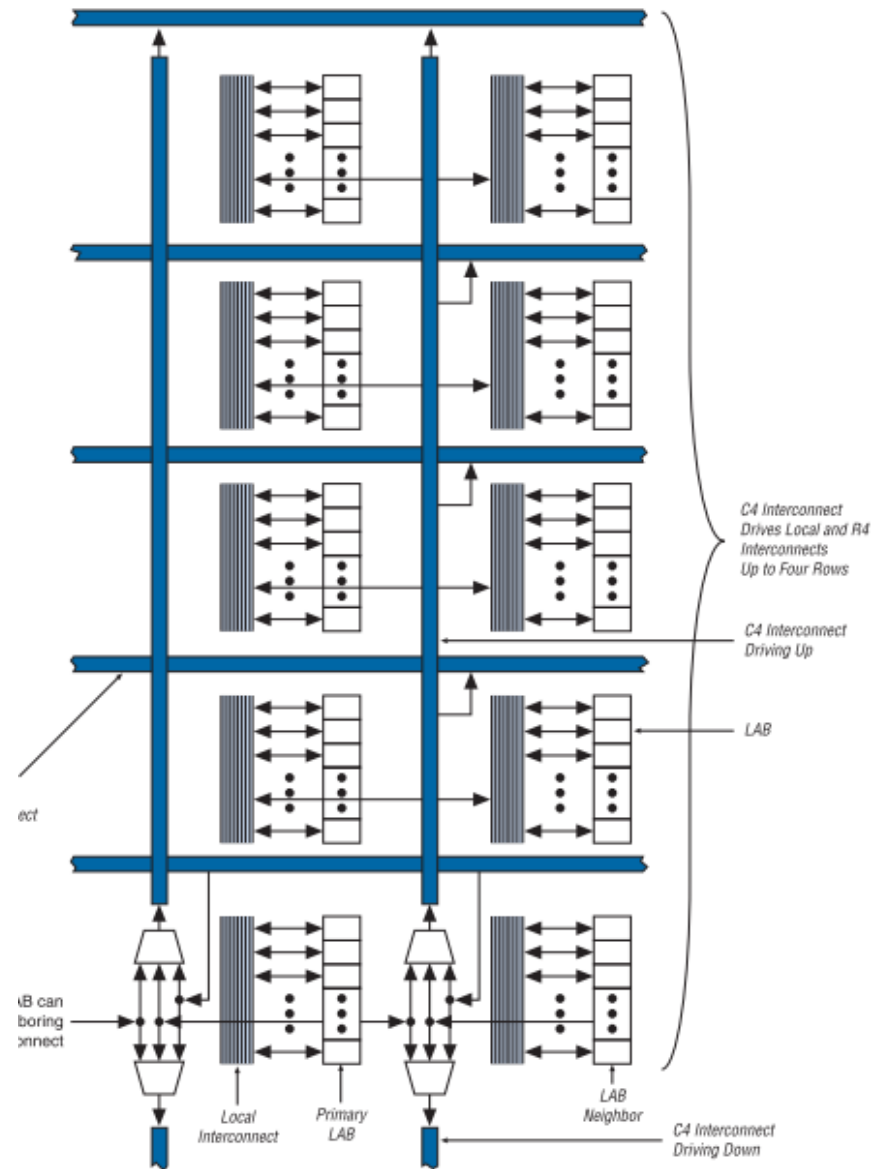
- Fast Local Interconnect
- Row interconnections
  - Direct link interconnects between LABs and adjacent blocks
  - R4 Interconnects traversing four blocks to the right or left
  - R24 interconnects for high-speed access across the length of the device





# Multitrack Interconnections

- Columns interconnections
  - Register chain interconnects within an LAB
  - C4 interconnects traversing a distance of four blocks in an up and down direction
  - C16 interconnects for high-speed vertical routing through the device



# Connections

**Table 2-2. Cyclone II Device Routing Scheme (Part 1 of 2)**

Source	Destination												
	Register Chain	Local Interconnect	Direct Link Interconnect	R4 Interconnect	R24 Interconnect	C4 Interconnect	C16 Interconnect	LE	M4K RAM Block	Embedded Multiplier	PLL	Column IOE	Row IOE
Register Chain								✓					
Local Interconnect								✓	✓	✓	✓	✓	✓
Direct Link Interconnect		✓											
R4 Interconnect		✓		✓	✓	✓	✓						
R24 Interconnect				✓	✓	✓	✓						
C4 Interconnect		✓		✓	✓	✓	✓						
C16 Interconnect				✓	✓	✓	✓						
LE	✓	✓	✓	✓		✓							
M4K memory Block		✓	✓	✓		✓							
Embedded Multipliers		✓	✓	✓		✓							
PLL			✓	✓		✓							
Column IOE						✓	✓						
Row IOE			✓	✓	✓	✓							

# Clock Network

- Up to 16 global clock networks
- Up to four PLLs
- Global clock network dynamic clock source selection
- Global clock network dynamic enable and disable

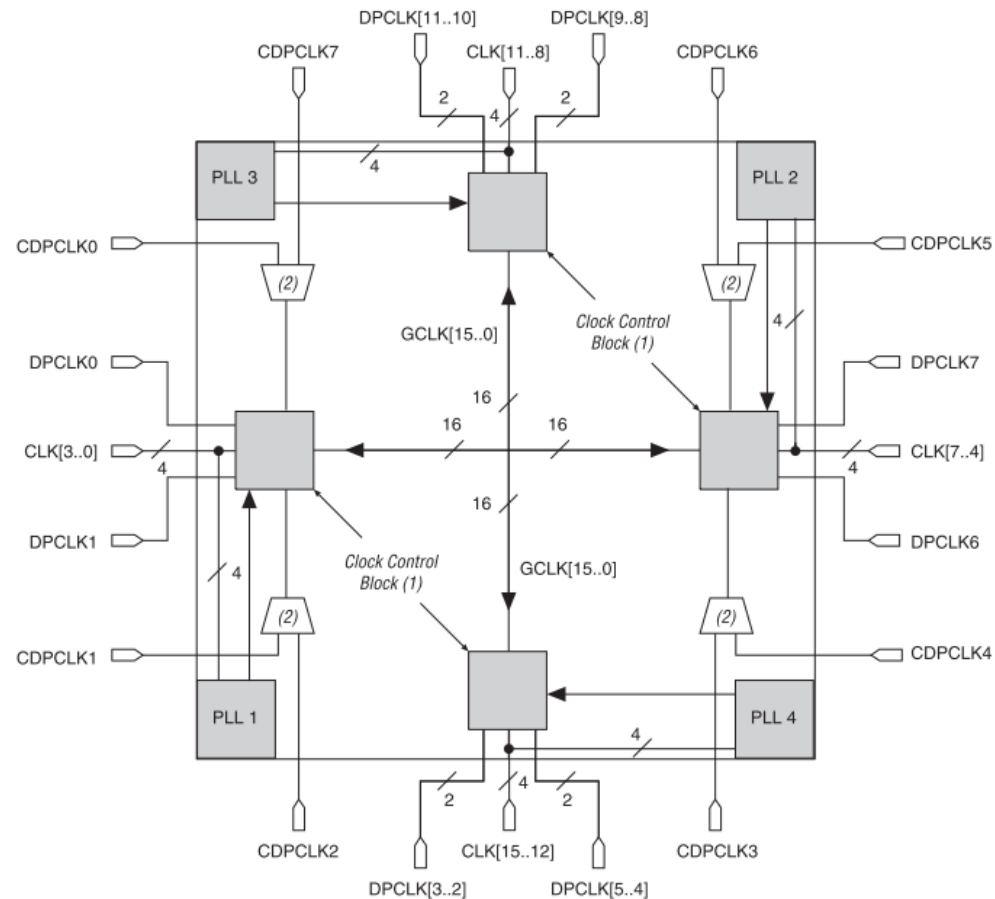
Each global clock network has a **clock control block** to select from a number of input clock sources (PLL clock outputs, CLK pins, DPCLK pins, and internal logic) to drive onto the global clock network

*Table 2–3. Cyclone II Device Clock Resources*

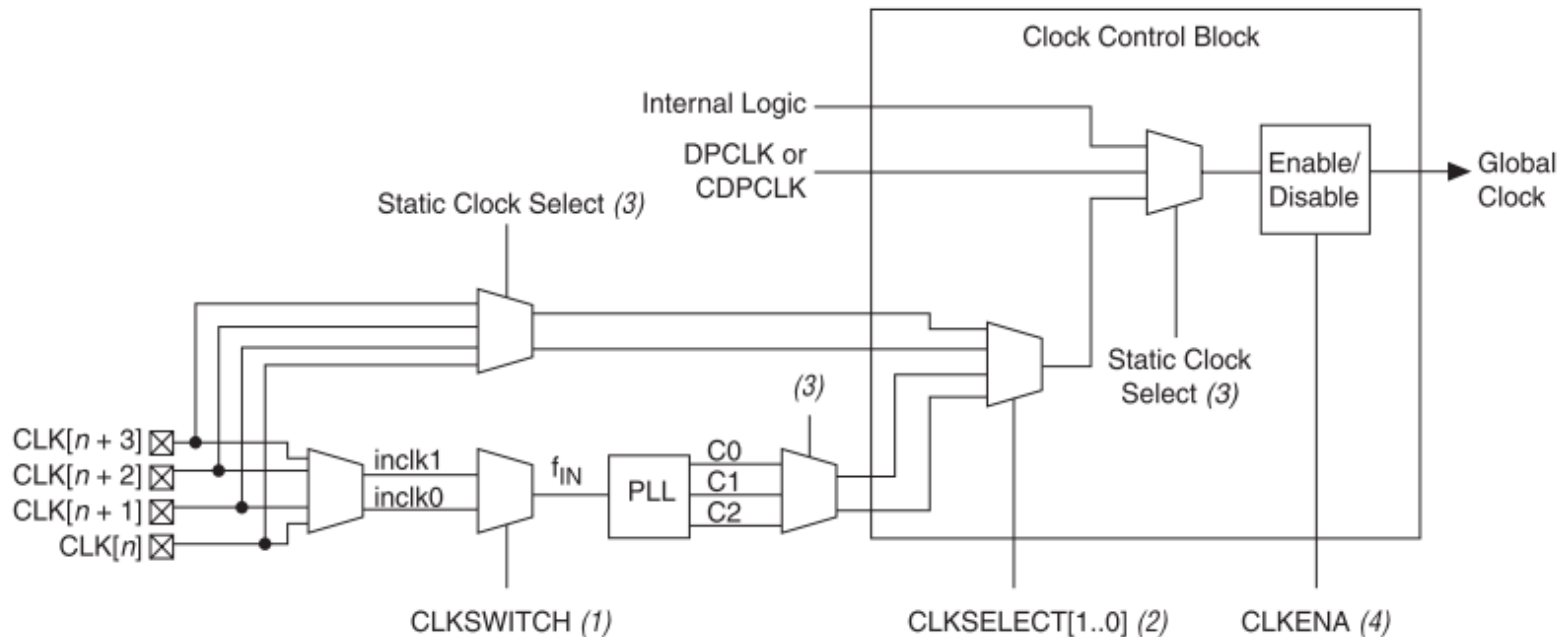
Device	Number of PLLs	Number of CLK Pins	Number of DPCLK Pins	Number of Global Clock Networks
EP2C5	2	8	8	8
EP2C8	2	8	8	8
EP2C20	4	16	20	16
EP2C35	4	16	20	16
EP2C50	4	16	20	16
EP2C70	4	16	20	16

# Clock Network Components

- 16 CLK: dedicated pin
- 12 DPCLK: dual purpose clock
- 8 CDPCLK: Corner DPCLK (note: mux delay)
- 4 Clock Control Blocks
- 16 Clock Networks



# Clock Control Block



## Notes to Figure 2-13:

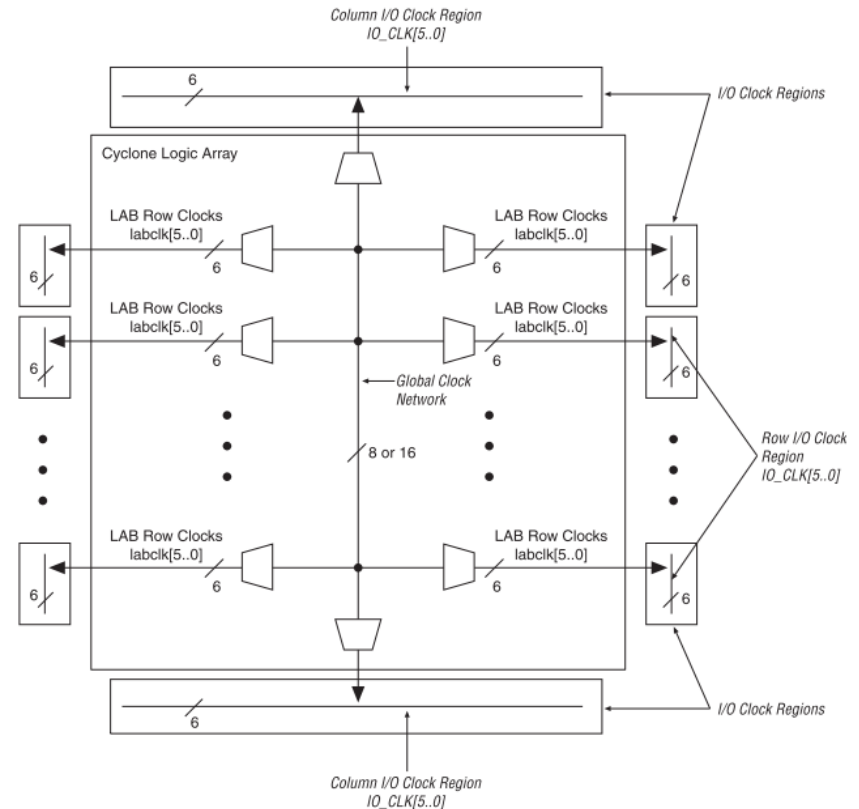
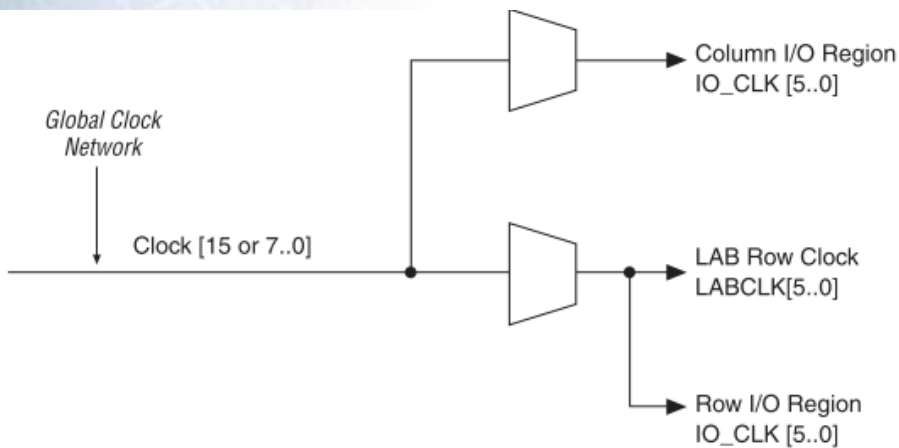
- (1) The CLKSWITCH signal can either be set through the configuration file or it can be dynamically set when using the manual PLL switchover feature. The output of the multiplexer is the input reference clock ( $f_{IN}$ ) for the PLL.
- (2) The CLKSELECT [1 . . 0] signals are fed by internal logic and can be used to dynamically select the clock source for the global clock network when the device is in user mode.
- (3) The static clock select signals are set in the configuration file and cannot be dynamically controlled when the device is in user mode.
- (4) Internal logic can be used to enabled or disabled the global clock network in user mode.

# Clock Distribution

Cyclone II devices contains 16 global clock networks. The device uses multiplexers with these clocks to form **six-bit buses** to drive

- column IOE clocks,
- LAB row clocks,
- row IOE clocks

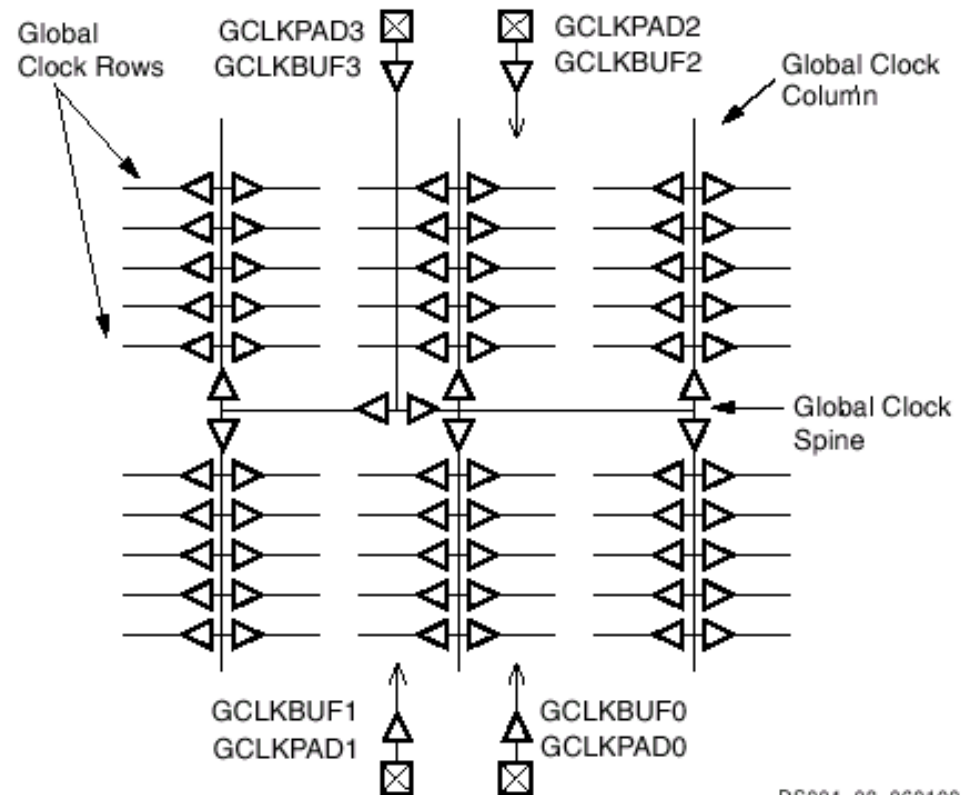
Another multiplexer at the LAB level selects **two** of the six LAB row clocks to feed the LE registers within the LAB





# Clock distribution Nets on Xilinx

- High speed
- Low skew
- 4 distribution nets
- 4 dedicated input PADS
- 4 dedicated Global buffers with inputs or
  - from clock pad
  - from internal signal



DS001\_08\_060100

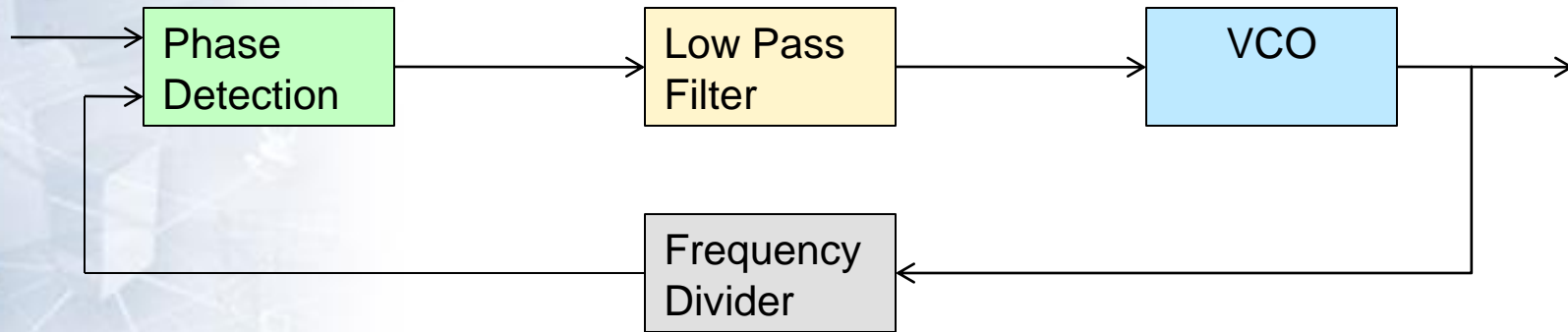
Figure 9: Global Clock Distribution Network

# PLL

Cyclone II PLLs provide general-purpose clocking as well as support for the following features:

- Clock multiplication and division
- Phase shifting
- Programmable duty cycle
- Up to three internal clock outputs
- One dedicated external clock output
- Clock outputs for differential I/O support
- Manual clock switchover
- Gated lock signal
- Three different clock feedback modes
- Control signals

# PLL (teoria)



- *Phase Detection*: fornisce un segnale proporzionale allo sfasamento tra i segnali in ingresso
- *Low Pass Filter*: integra le variazioni del segnale precedente
- *VCO*: oscillatore locale controllato in tensione
- *Freq. Divider* (eventuale): utile per modificare la freq. del segnale in uscita

# PLL (applicazioni)

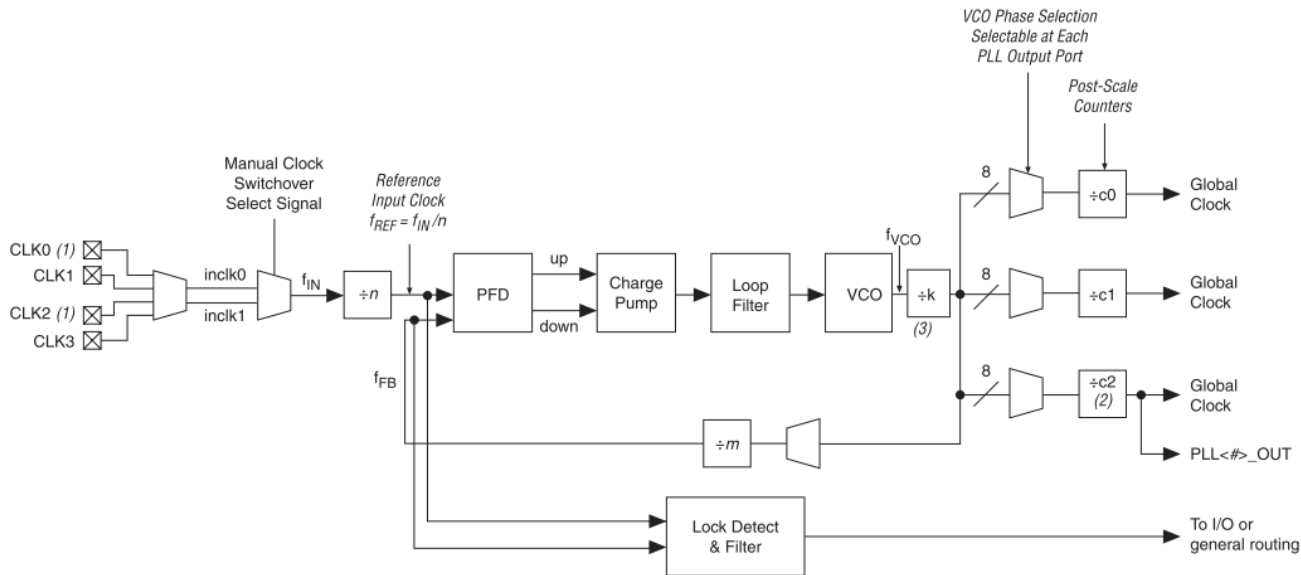
## ■ Analogiche:

- Ricostruzione della “portante” da un segnale modulato
- Demodulatore FM (il segnale in ingresso al VCO è il segnale demodulato)
- Sintetizzatore di frequenza (in ingresso si mette un segnale stabile e usando il divisore si possono sintetizzare segnali a frequenza più elevata)

## ■ Digitali

- Recupero del clock (da un segnale digitale sincrono)
- Deskewing o Riallineamento del clock
- Generazione di clock a diversa frequenza e distribuzione

# Cyclone II - PLL



- Clock Multiplication and Division: Using freq. dividers ( $m/n$ )
- Phase Shift (delay compensation)
- Programmable Duty Cycle: Using C0,C1,C2
- Clock Outputs: 3 for global clock nets and 1 to a dedicated Out Pin
- Manual clock switchover: to switch between different sources
- Lock signal available
- Clock feedback modes: zero delay, normal mode, no compensation mode
- Control signals: `pllenable`, `areset` `pfdena`

# Embedded Memory M4K

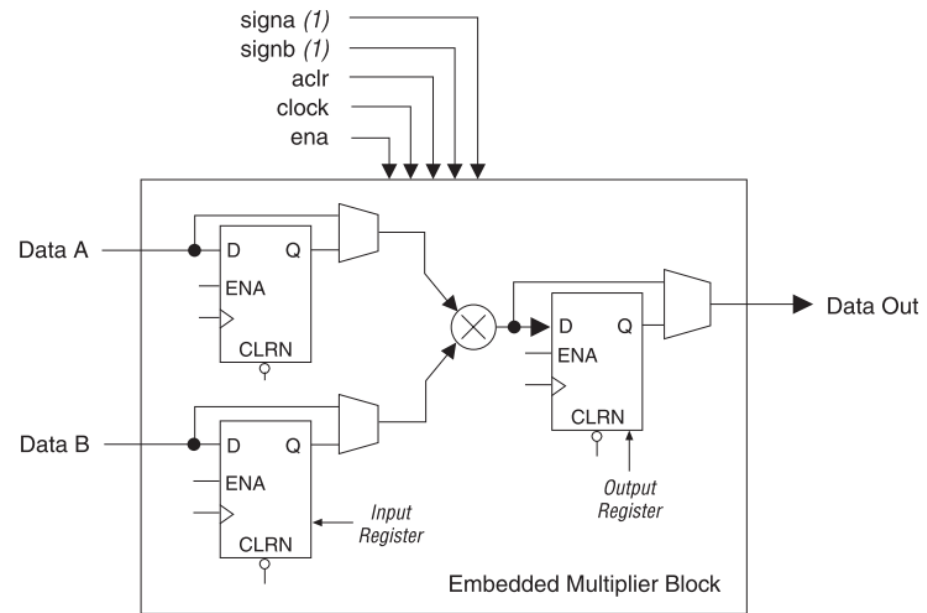
- Input and output register integrated
- Each M4K block can implement various types of memory
  - single-port RAM (simultaneous reads and writes are not required)
  - simple dual port RAM (supports a simultaneous read and write)
  - true dual-port memory (any combination of two-port operations)
  - ROM
  - Single clock and dual clock FIFO buffers.

*Table 2–6. M4K Memory Capacity & Distribution in Cyclone II Devices*

Device	M4K Columns	M4K Blocks	Total RAM Bits
EP2C5	2	26	119,808
EP2C8	2	36	165,888
EP2C20	2	52	239,616
EP2C35	3	105	483,840
EP2C50	3	129	594,432
EP2C70	5	250	1,152,000

# Embedded Multipliers

- Configurable as:
  - 1 Multiplier 18x18
  - 2 Multipliers 9x9
- Up to 250MHz
- Input/output registers available
- Signed/Unsigned operation



**Table 2-11. Number of Embedded Multipliers in Cyclone II Devices** *Note (1)*

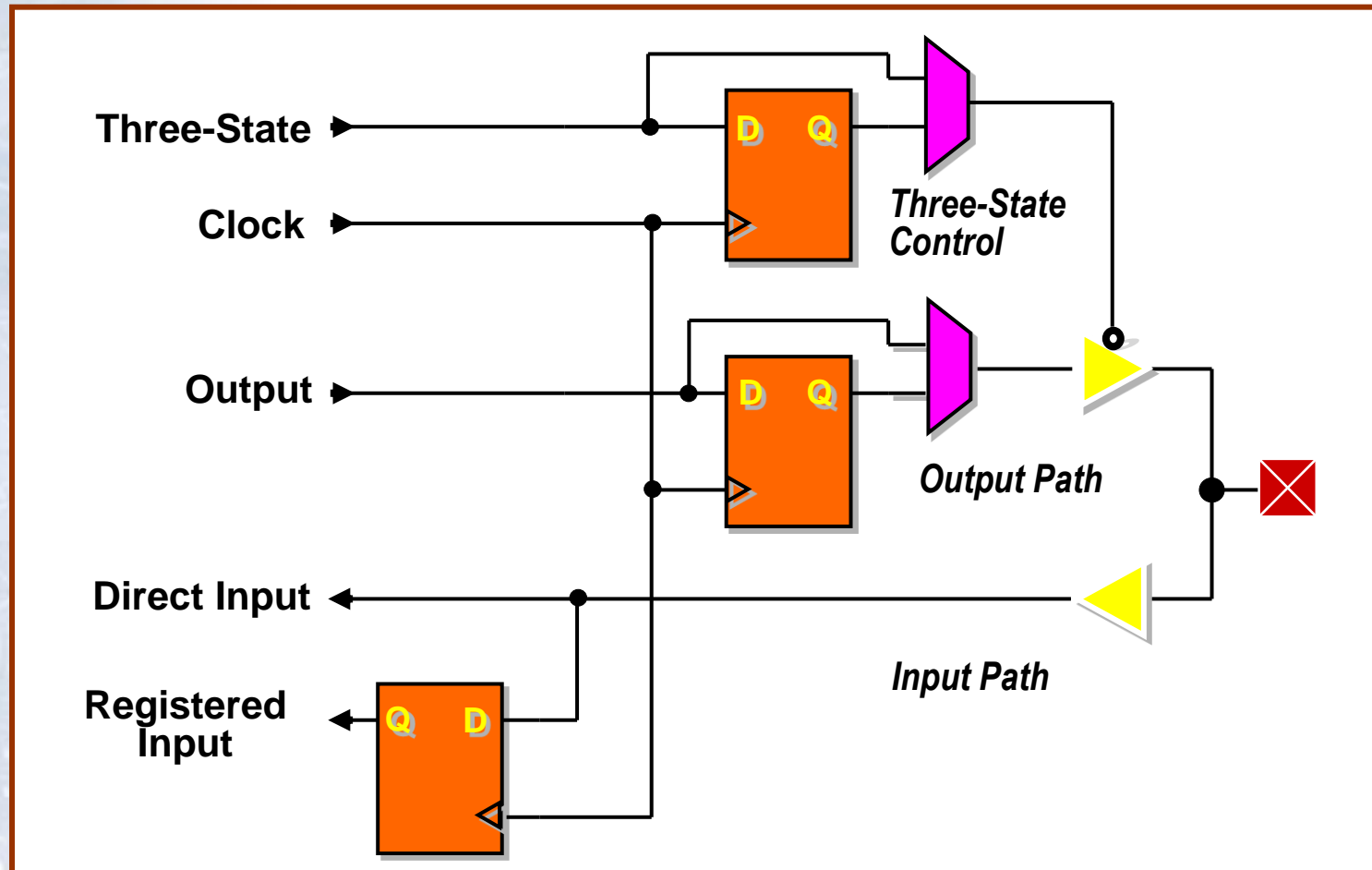
Device	Embedded Multiplier Columns	Embedded Multipliers	9 × 9 Multipliers	18 × 18 Multipliers
EP2C5	1	13	26	13
EP2C8	1	18	36	18
EP2C20	1	26	52	26
EP2C35	1	35	70	35
EP2C50	2	86	172	86
EP2C70	3	150	300	150



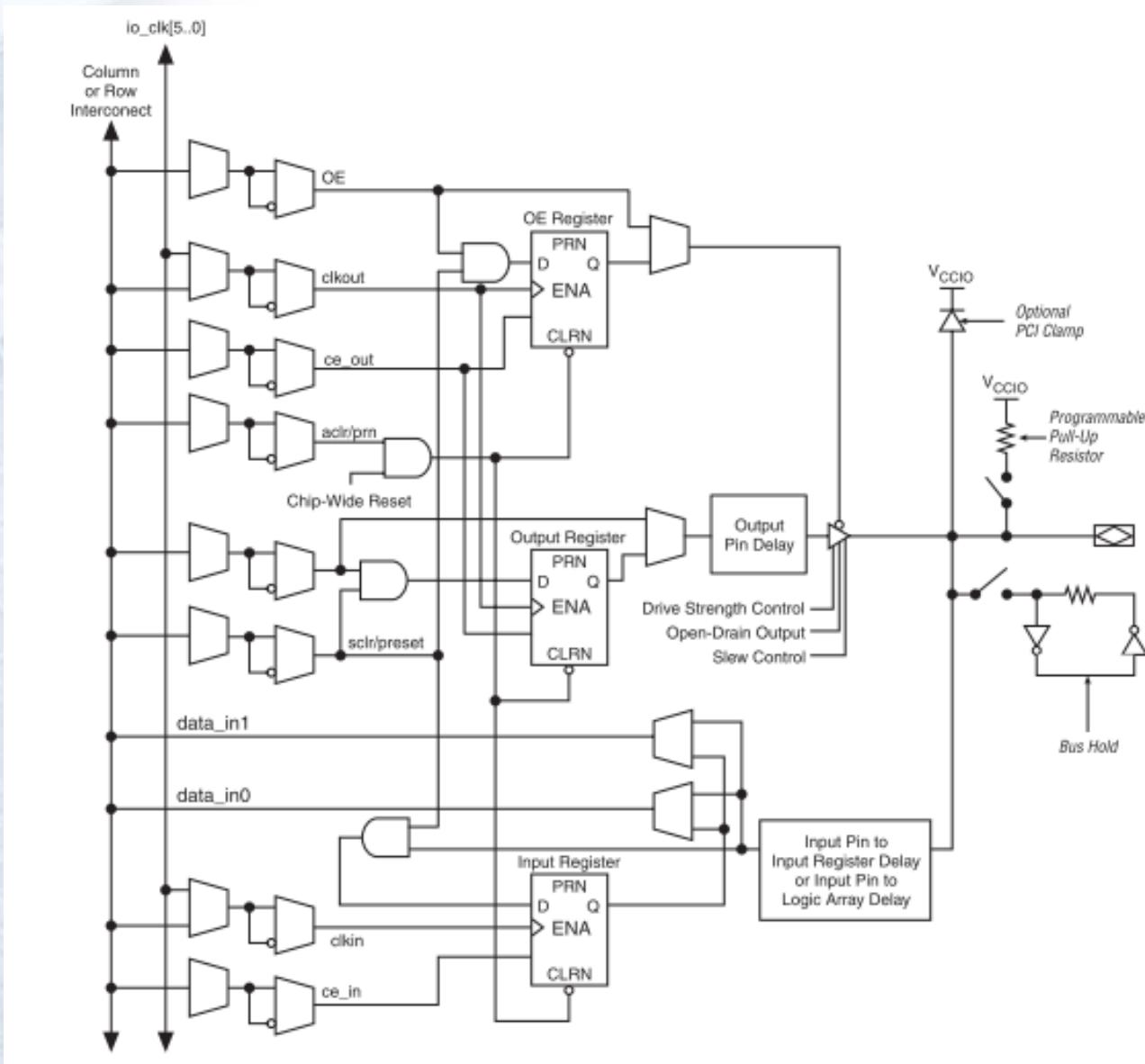
# Cyclone II I/O Features

- In/Out/Tri-state
- Flip-flop option
- Series resistors
- Different Voltages and I/O Standards (3.3-V, 64- and 32-bit, 66- and 33-MHz PCI compliance )  $V_{REF}$  pins
- Output Drive strength control (standards have several levels of drive strength that you can control)
- Slew rate control
- Differential and single-ended I/O standards
- Joint Test Action Group (JTAG) boundary-scan test (BST) support
- Weak pull-up resistors during configuration
- Bus-hold circuitry
- Programmable pull-up resistors in user mode
- Programmable input and output delays
- DQ and DQS I/O pins (for memory interface control)

# Cyclone II I/O Buffer



# Cyclone II I/O Buffer



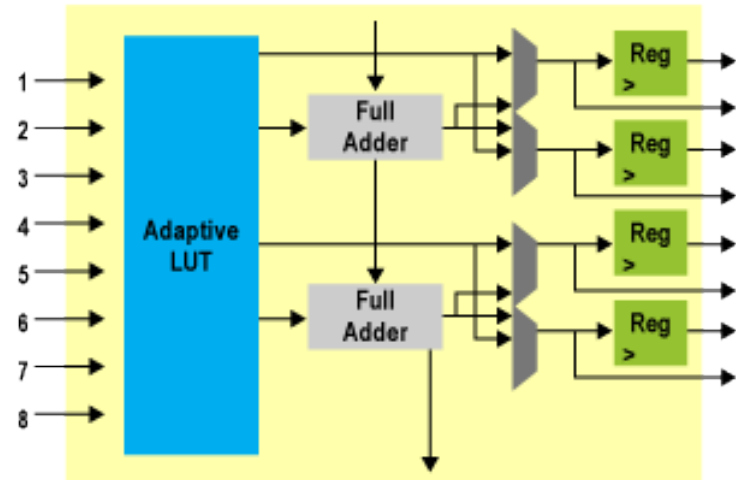
# STRATIX - V

# ALM - MLAB

## The Adaptive Logic Module:

- 8 bit LUT
- 2 Adders
- 4 Register
- 4 Outputs

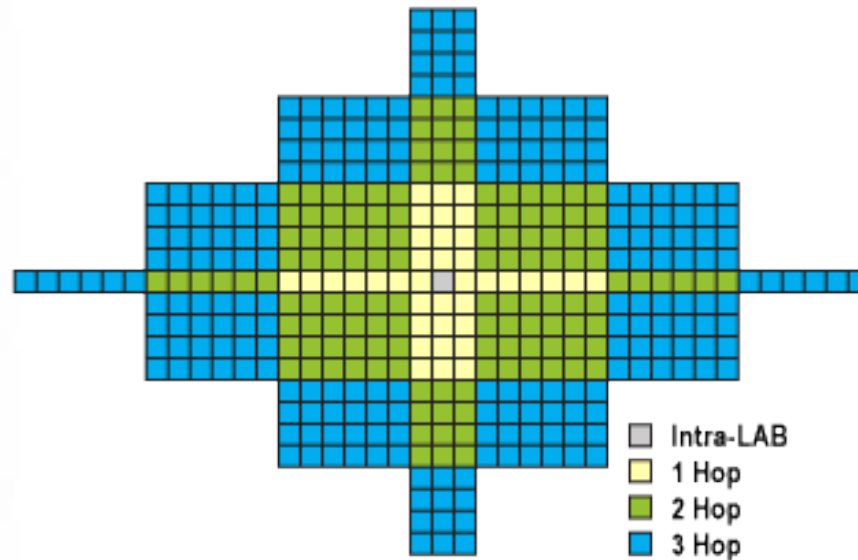
Stratix V ALM Advantages



The core of a Stratix series FPGA includes a logic array block (LAB), comprised of regular ALMs or configured as a simple, 640-bit dual-port SRAM block (known as a MLAB)

# Multi Track Routing Architecture

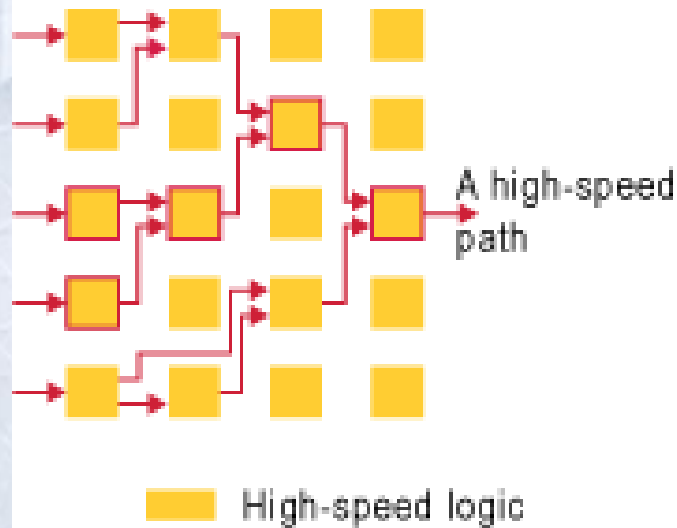
Stratix V - Multi-Track Routing Architecture



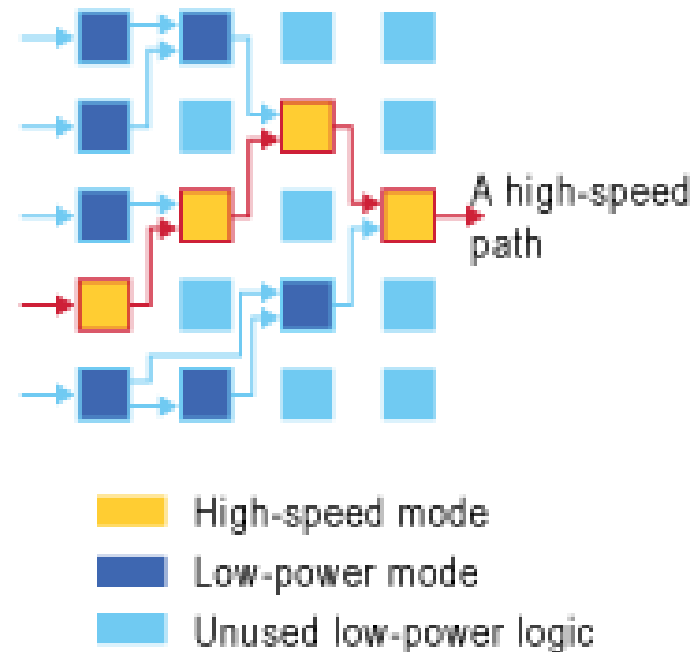
Hops	Reachable Logic Elements (LEs)
1	1,007
2	3,498
3	6,042
<b>Total</b>	<b>10,547</b>

# Low Power Consumption

Standard FPGA

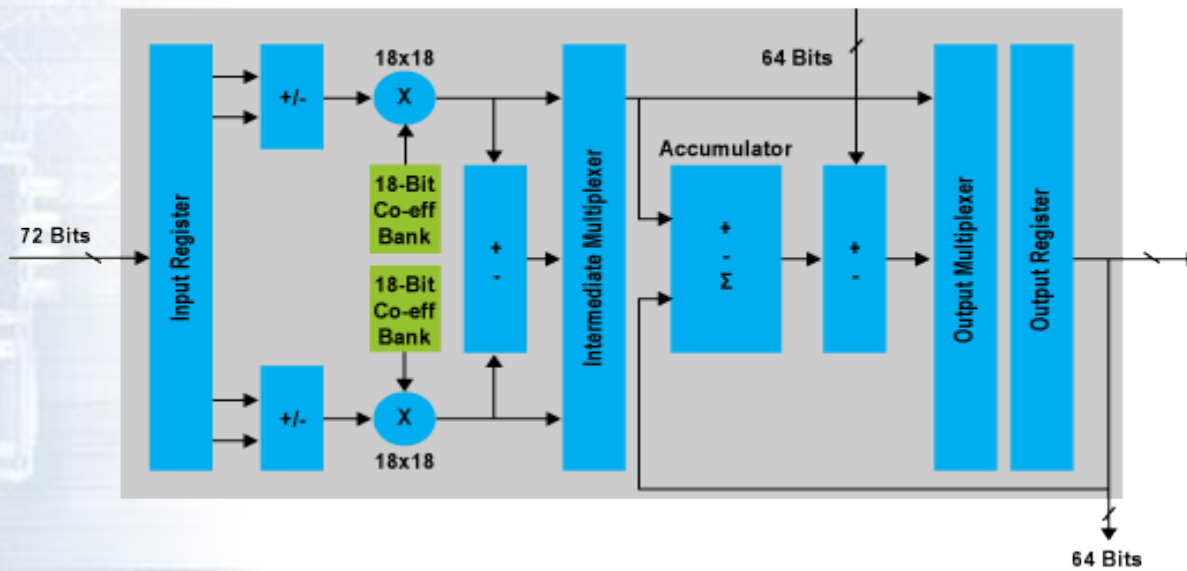
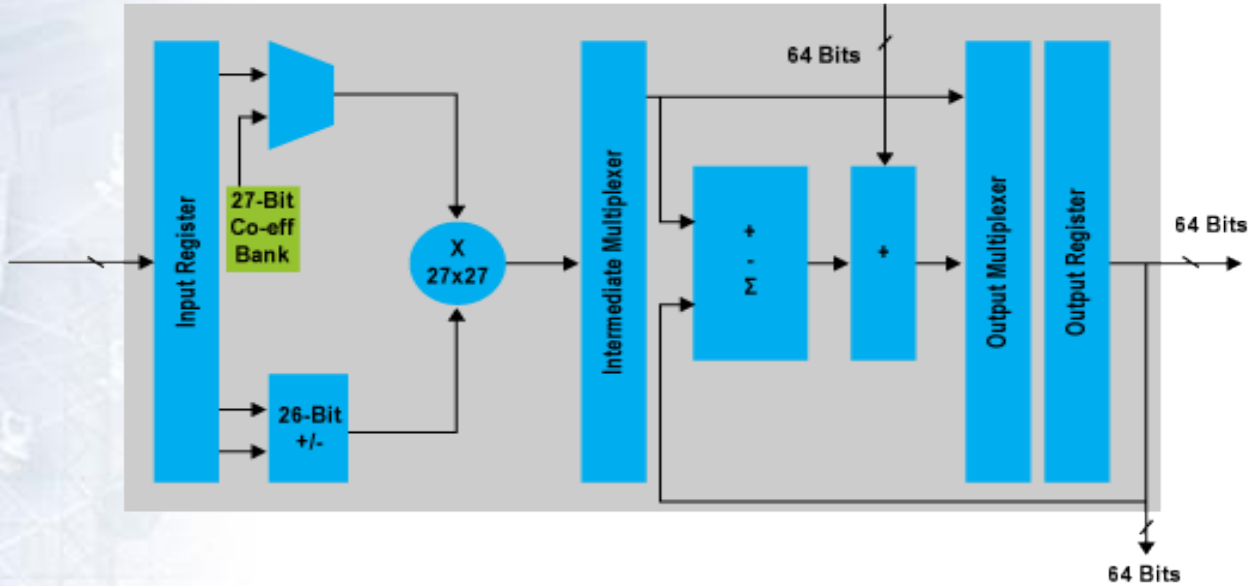


Stratix IV FPGA with Programmable Power Technology





# DSP Configurable Block



# Features:

- Up to 4096 DSP Blocks
- Up to 2688 M20K Blocks (52 Mbit)
- Up to 1.052.000 LE
- Up to 397.000 ALM Block
- Up to 48 14.1 Gbit/s Transceivers

And ....

- External Memory Interfaces
- Design Security
- Partial Reconfiguration
- Single Event Upset
- Remote System Upgrades

# FPGA Configuration

# Generalità

- La memorizzazione dei dati è di tipo SRAM (Volatile)
- Una FPGA **DEVE** essere configurata ad ogni accensione
  - Vi sono 4 metodi di programmazione
    - Active Serial (AS) a 20 MHz necessario un chip di memoria seriale
    - Passive Serial (PS) necessario un dispositivo “master”
    - Fast Active Serial (AS) a 40 MHz
    - JTAG (protocollo dedicato)
  - Il metodo di programmazione dipende dallo stato di 2 appositi piedini : MSEL [ 1 . . 0 ]

<b>Configuration Scheme</b>	<b>MSEL1</b>	<b>MSEL0</b>
AS (20 MHz)	0	0
PS	0	1
Fast AS (40 MHz) (1)	1	0
JTAG-based Configuration (2)	(3)	(3)

# Dispositivi di Memoria

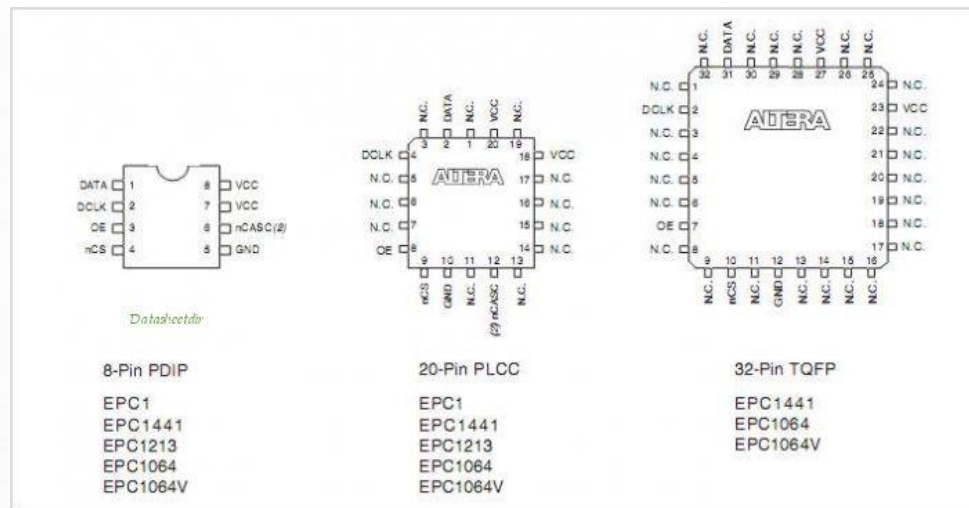
- Esistono in commercio opportuni dispositivi di memoria per interfacciarsi facilmente all'FPGA

- EPCSx (x=1,4,16, 64, 128)

Memorie seriali interfacciabili con 4 piedini

- EPCx (x=4, 8, 16)

Sono dispositivi di programmazione più complessi dei precedenti che gestiscono attivamente la fase di programmazione



# Nella scheda DE1

- Programmazione On Power ON attraverso
  - EPCS16 e protocollo AS
- Programmazione successiva
  - via JTAG (gestito da USB Blaster)

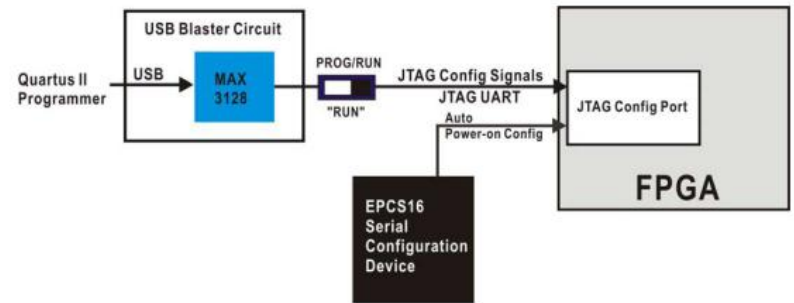


Figure 4.1. The JTAG configuration scheme.

- Programmazione della memoria EPCS16
  - Tramite USB Blaster

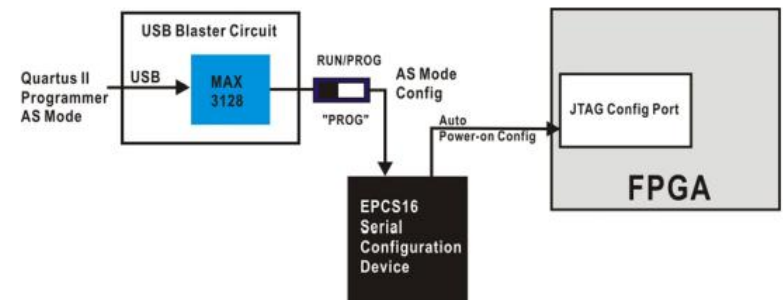


Figure 4.2. The AS configuration scheme.



# File di configurazione

- Non dipende dallo stato di occupazione finale della FPGA ma solo dal tipo di FPGA da programmare
- Si possono però impiegare algoritmi (proprietary) di compressione dei dati) – La FPGA esegue la decompressione (solo per AS e PS; No per JTAG)

*Table 13–3. Cyclone II Raw Binary File (.rbf) Sizes* *Note (1)*

Device	Data Size (Bits)	Data Size (Bytes)
EP2C5	1,265,792	152,998
EP2C8	1,983,536	247,974
EP2C15	3,892,496	486,562
<b>EP2C20</b>	<b>3,892,496</b>	<b>486,562</b>
EP2C35	6,858,656	857,332
EP2C50	9,963,392	1,245,424
EP2C70	14,319,216	1,789,902

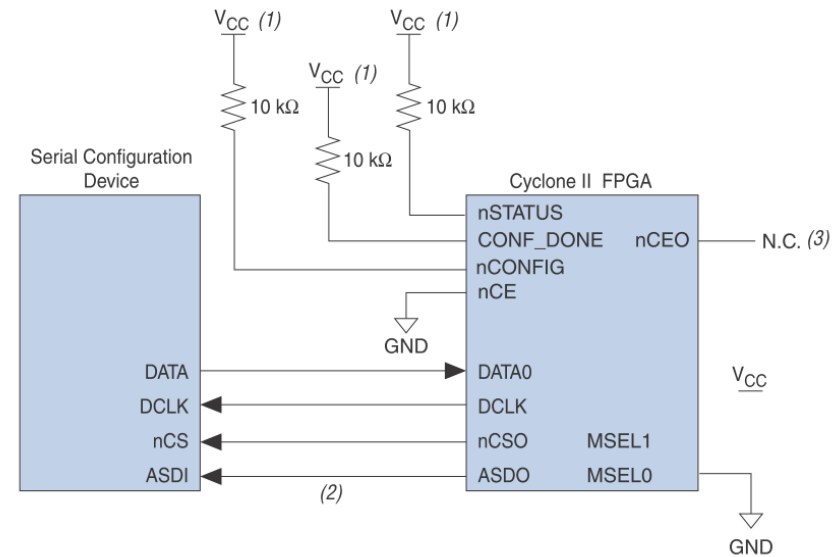


# Active Serial (AS)

- L'FPGA svolge un ruolo attivo nella programmazione
- Si interfaccia tramite 4 Pin alla memoria seriale
  - DCLK (genera il clock)
  - nCS (pilota il chip select)
  - ASDI (Dati di controllo)
  - DATA (riceve i dati di configurazione)
- Si interfaccia eventualmente con l'ambiente esterno attraverso
  - nSTATUS
  - CONF\_DONE
  - nCONFIG
  - nCE (chip enable)

Che hanno funzione sia di ingresso (per ricevere comandi) che di uscita (per segnalare lo stato o eventuali errori)

  - nCE (chip enable)



# Fasi di Programmazione

- Power UP
  - Attende che i Valori di tensione superino una certa soglia
- Reset Stage
  - La FPGA resetta il suo stato
  - La FPGA abbassa nSTATUS e CONF\_DONE
  - Mette in Tri-state tutti i PIN di I/O con deboli resistenze di pull-up
  - Dopo circa 100 ms rilascia nSTATUS se questo assume il valore alto (con resistenza di pull-up) va alla fase successiva.
- Configuration Stage
  - Tramite oscillatore interno genera DCLK (a 20 o 40 MHz)
  - Abbassa nCS0 (per abilitare la memoria)
  - Genera i segnali di controllo tramite ASDO
  - Dopo aver ricevuto tutti i dati rilascia CONF\_DONE. Se questo assume il valore alto (con resistenza di pull-up) va alla fase successiva.

# Fasi di Programmazione

## ■ Initialization Stage

- Il clock di inizializzazione può essere generato tramite generatore interno a 10 MHz o prelevato dall'esterno tramite il piedino CLKUSR. Questo piedino può essere usato per controllare l'inizializzazione (l'inizializzazione è ultimata dopo 299 cicli)
- Si può sfruttare (opzionale – via software) il piedino INIT-DONE per ottenere la segnalazione che la fase di inizializzazione è ultimata

## ■ User Mode

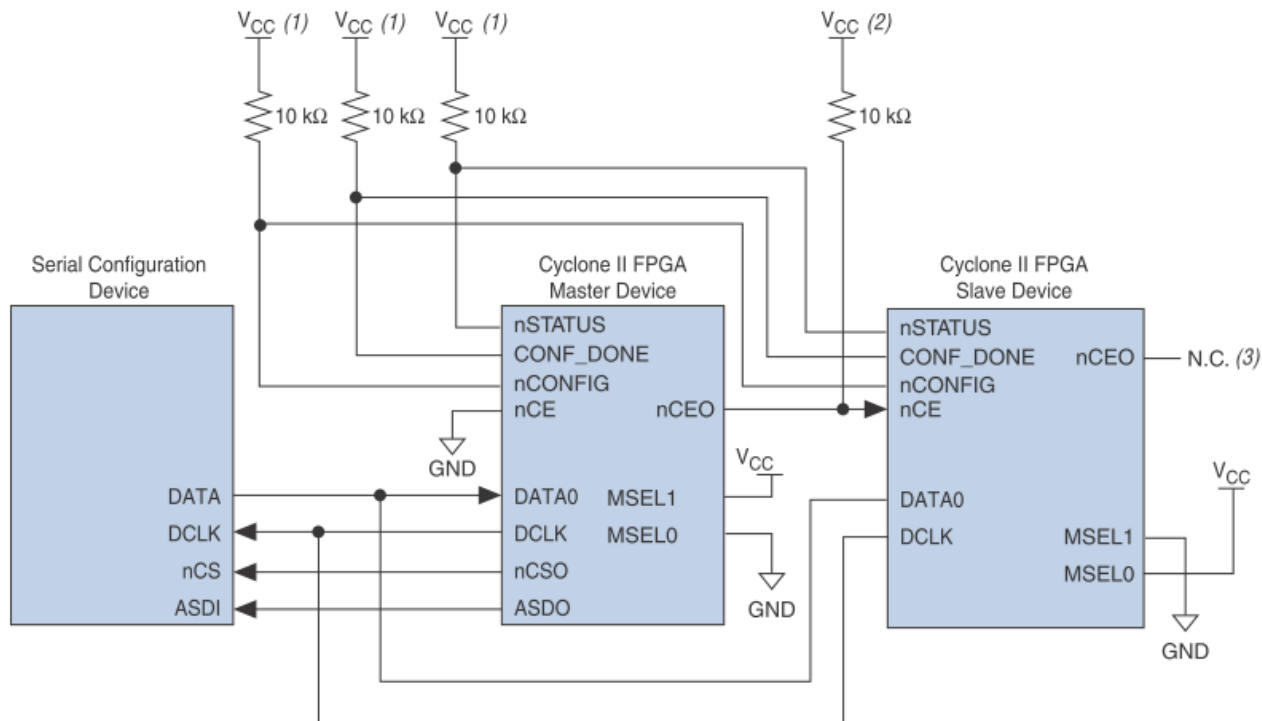
- Tutti i piedini di I/O prendono la loro funzione definitiva
- Si può nuovamente ri-programmare l'FPGA abbassando per almeno 2 $\mu$ s nCONFIG.

## ■ Errori

- Se durante la configurazione viene rilevato un errore si abbassa nSTATUS e CONF\_DONE rimane basso.
- Se via Software è stata attivata l'opzione "Auto-restart configuration after error" l'FPGA resetta la memoria con un impulso su nCS0 e ricomincia

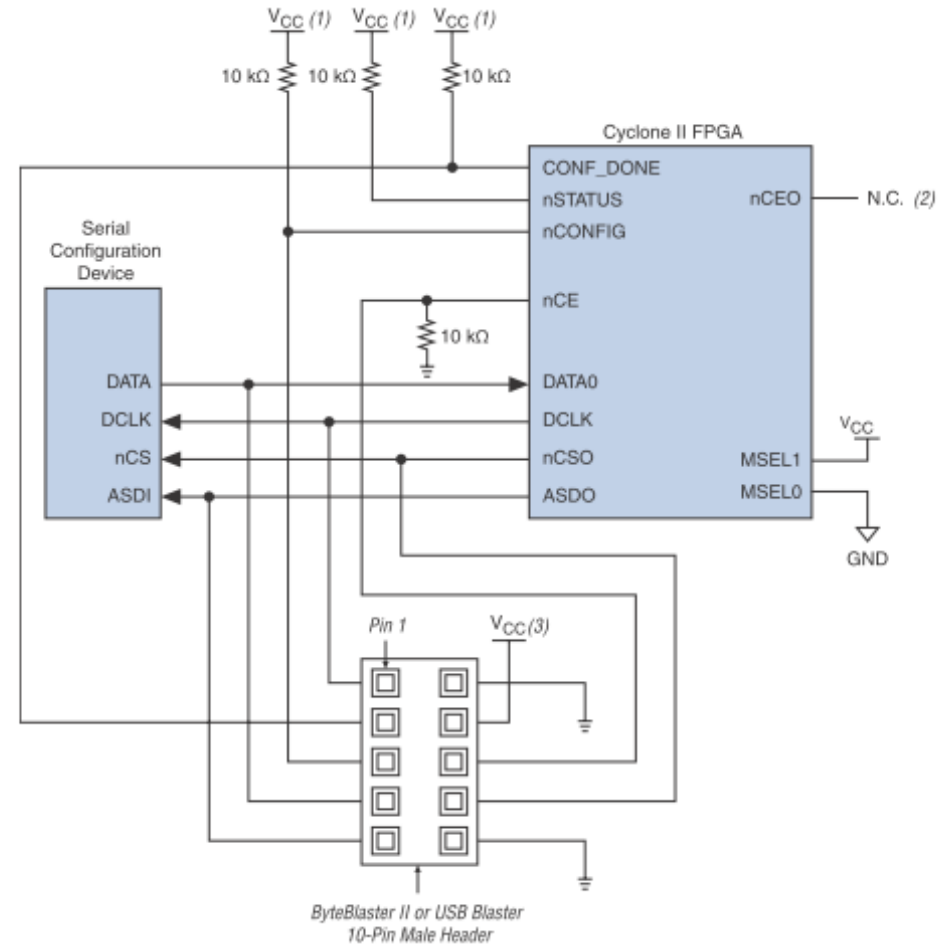
# Multiple Device Configuration

- Una sola memoria
- Il primo Dispositivo funziona da master, controlla il flusso di dati e l'attivazione del secondo dispositivo e funziona in AS
- Tutti gli altri dispositivi funzionano in PS
- Grazie a CONF\_DONE condiviso tutti i dispositivi si inizializzano contemporaneamente



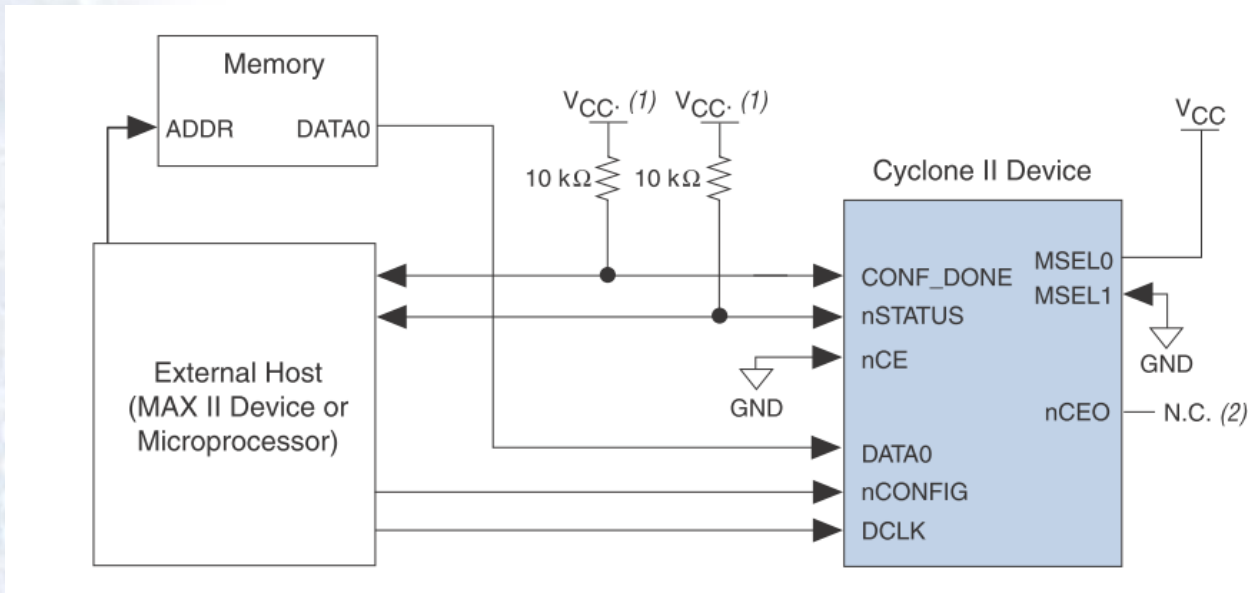
# Programmazione della Memoria

- Utilizzo di USB-Blaster
- Programmazione on-board
- Tramite nCE si disabilita l'FPGA mentre si configura la memoria
- Ultimata la programmazione della memoria l'FPGA viene attivata e questa accede ai dati della memoria.



# Passive Serial (PS)

- L'FPGA ha un ruolo passivo
- Il flusso di dati viene controllato da un dispositivo esterno (es. una CPLD o un microprocessore) ... si può impiegare anche un "Enhanced Configuration Device" (EPCx)
- L'FPGA interagisce con esso tramite nSTATUS e CONF\_DONE



# JTAG

- Il protocollo JTAG, nato per la testabilità dei circuiti ha accesso a tutti i piedini, può pertanto essere impiegato per programmare e controllare il dispositivo
  - Es. può controllare i piedini CONF\_DONE o nSTATUS durante la configurazione
- Impiega 4/5 piedini dedicati

