Indice

CAPITOLO 1: GENERALITA’

1. DEFINIZIONI BASE

Prima di addentrarsi nella parte più specifica è necessario fornire una definizione base su che cosa è una logica programmabile. Una logica programmabile è un componente hardware in grado di esplicare una funzionalità programmabile da utente mediante un insieme di tool software in grado di supportare tutti i passi necessari del flusso di progettazione. Il risultato di questo flusso, definibile come programma di utente o applicativo o file di configurazione, è un file binario che contiene l’immagine virtuale della logica di utente che una volta caricata nel componente scelto fornirà un’ immagine reale del progetto.

Il caricamento del file nella logica programmabile è quell’operazione definita come programmazione del componente. Esistono varie metodologie di programmazione dei componenti e molte di queste si differenziano in base alle tecnologie costruttive. Essenzialmente possiamo ricondurci a due grosse categorie tecnologiche per analizzare i vari tipi di programmazione. La prima del tipo non volatile, consente una volta compiuta la programmazione di mantenere i dati di utente nel dispositivo per un numero indefinito di inserzioni o disinserzioni della tensione di alimentazione del circuito stampato che ospita la logica. La seconda del tipo volatile, deve essere realizzata ad ogni inserzione della tensione di alimentazione e necessita di un ulteriore componente, questa volta non volatile, che funga da supporto dati.

Le logiche della categoria non volatili possono essere programmate su un programmatore di memorie e poi saldate sul circuito stampato, dalla macchina per il test delle piastre (ATE - Automatic Test Equipment), da un microprocessore se presente a bordo del circuito stampato mediante una delle sue porte di I/O, da Personal Computer o Workstation mediante un cavo di download se queste supportano la programmazione in system. Di queste elencate la preferibile per i progettisti, il collaudo e la produzione è l’ultima se la logica lo consente. Questo perchè senza alterare le connessioni elettriche sul circuito stampato che ospita la logica, per i progettisti è possibile reagire rapidamente alla scoperta di bachi o a variazioni di specifica, per il collaudo perchè può sfruttare le logiche per specializzarle per un eventuale test funzionale della piastra prima di configurarle con la definitiva applicazione, per la produzione perchè può gestire un unico codice di fabbrica e modificare schede in campo senza sostituirle.

Le logiche della categoria volatile, che da qui in poi definiremo del tipo static RAM, possono essere programmate mediante microprocessore o mediante memoria esterna (EPROM seriale, EEPROM seriale, Flash, EEPROM, EPROM). Per prototipazione o test è possibile ricorrere al download o alla macchina di test ma questo come ricordato in precedenza resta valido fino alla presenza della tensione di alimentazione.

Le logiche del tipo static RAM per definizione sono riprogrammabili oltre che programmabili, proprio per il loro tipo di funzionamento; non tutte le logiche non volatili lo sono. Ci sono logiche che si programmano una sola volta (OTP - One Time Programmable), logiche che si riprogrammano previa estrazione dal circuito stampato (questo implica l’adozione di uno zoccolo) e logiche che si riprogrammano in system (ISP - In System Programmable) senza quindi la necessità di estrarle dal circuito stampato.

2. VANTAGGI

E' doveroso celebrare i vantaggi che si ottengono dall'utilizzo delle logiche programmabili anche se la programmabilità o riprogrammabilità dell'hardware li rende abbastanza evidenti.

In progetti in cui è presente un certo numero di logica digitale sparsa è possibile ridurre il numero dei componenti utilizzati e quindi dei codici che deve gestire la fabbrica.

Il tempo di prototipazione è una variabile dell'iter di progetto che si riduce. Infatti non ci sono attese legate ad esempio all'arrivo di componenti esterni o a realizzazioni di gate array. Inoltre è possibile progettare con una nuova filosofia. Infatti, ad esempio, nell'implementazione di un pll completamente digitale la realizzazione di un prototipo in tempi brevi e riconfigurabile altrettanto rapidamente consente di analizzare il comportamento di un simile circuito in laboratorio e non in una simulazione che pur se efficace non coprirà i secondi o i minuti di funzionamento che una accurata analisi richiederebbe.

Accanto al tempo di prototipazione si riduce anche quello di reazione a variazioni di specifica del progetto, che risulta essere meno pesante in termini di costo e tempo, di una rimascheratura del silicio come avviene nei gate array.

E' possibile compiere il remote upgrade cioè la riconfigurazione a distanza della logica. Se questa è ospite di un satellite, una sonda spaziale, un ponte radio posto in una località difficilmente raggiungibile è possibile, prevedendo a bordo un modem o un'intefaccia internet e un'architettura a microprocessore, mutare il funzionamento dell'hardware dalla centrale operativa con gli indubbi vantaggi che ne derivano.

Si è accennato al mondo dei gate array e proprio quel mondo viene utilizzato come termine di confronto da parte dei vari vendors di logiche programmabili sia in termini di densità, misurata in gate, che in termini di tecnologia. I gate di un gate array non sono i gate di una logica programmabile e i confronti che si fanno sono inopportuni anche se possono essere comodi da fare. Questa comodità comunque non deve divenire una fonte di rumore intenzionalmente aggiunto per catturare mercato e creare aspettative che poi si rivelano false. Sulla metrica comunque torneremo in un successivo capitolo.

Un primo confronto tra gate array e logiche programmabili che si può fare come pianificazione progettuale è basato sulla convenienza, previsto un certo numero di vendite del prodotto, dell'adozione dell'uno o dell'altro.

Molte ditte, indipendentemente della variabile vendite, partono a testa bassa e realizzano direttamente per un loro progetto un gate array imbarcandosi in lunghi tempi di sviluppo e costi. Le cose spesso in fase progettuale non vanno per il verso giusto e i tempi si dilatano e qualche volta i gate array una volta testati, per bachi sfuggiti alla simulazione o variazioni di specifica, sono da rifare.

In realtà occorrerebbe utilizzare un'analisi congiunta di molti fattori tra cui il tempo in cui si vuole aggredire il mercato con il proprio prodotto (time-to-market), le risorse disponibili in termini di progettisti e computer, il numero di pezzi da realizzare. Tipicamente a parità di funzionalità un gate array costa meno per singolo pezzo ma ha un costo iniziale legato alla mascheratura. Analizzando le due curve di costo legate al numero di campioni da produrre si trova il punto di intersezione (definito breakevent) al di sotto del quale conviene utilizzare una o più logiche programmabili, al di sopra del quale conviene utilizzare un gate array.

Come avviene in molti settori del mercato elettronico nessuno, per paura di esporsi troppo, fa una cosa così evidente ma nuova, finchè qualche ditta intraprendente la

utilizza come propria strategia. Ebbene,visto il successo che ne deriva (e sottinteso che bisogna anche saper fare oltre che saper organizzare) tutti gli altri dietro a rincorrere. Alla fine, mentre ci si chiede perchè non si è competitivi sul mercato, gli altri impongono le proprie idee nelle sedi di standardizzazione mondiale.

3. SVANTAGGI

Utilizzare una logica programmabile ha dei vantaggi ma è un dispositivo limitato e questo va tenuto in considerazione. Anche se la tecnologia progredisce in maniera rapida la capacità di un gate array risulta sempre superiore, perchè anche la tecnologia di questi ultimi evolve. I consumi in termini di potenza assorbita non sono paragonabili a parità di frequenza di funzionamento per tanti motivi tra cui l'efficacia di silicio utilizzato rispetto a quello disponibile e il tipo di tecnologia. Le frequenze di funzionamento raggiungibili nei gate array sono più alte di un ordine di grandezza. Inoltre, ma questo può non essere uno svantaggio, non devono essere previste circuiterie di programmazione.

4. TIPI DI LOGICHE

Possiamo dividere, da un punto di vista didattico, le logiche programmabili in due categorie: PLD (Programmable Logic Device) e FPGA (Field Programmable Gate Array). Qualche compagnia introduce le CPLD identificando in esse componenti di un tipo e dell'altro, ma per chiarezza è bene capire di cosa si parla prima di cambiar loro nome.

Le PLD sono costituite da blocchi logici molto potenti inseriti in una struttura predefinita di canali di interconnessione a ritardo controllato.

Gli FPGA sono formati da blocchi logici di ridotte potenzialità circondati da una complessa struttura di canali di interconnessione (o routing) segmentati.

Il blocco logico di una PLD si compone di una somma di prodotti e di un registro che può essere scavalcato se non c'è bisogno dell'elemento sequenziale. La somma di prodotti è realizzata mediante un and-array che realizza un numero anche elevato di prodotti logici o product term (ad esempio 20 o 24) e una or che ne può raccogliere fino a 20 senza ricorrere ad espansioni: questo giustifica la potenzialità della parte combinatoria.

Il blocco logico di un FPGA è basato sulla look-up table (LUT) che rappresenta una funzione booleana di quattro ingressi e una uscita e il flip flop che rappresenta l'elemento sequenziale anche in questo caso escludibile se necessario. La prima novità che si incontra nel mondo del programmabile rispetto agli altri campi della componentistica elettronica è che la LUT quando non è usata per implementare la logica di utente può essere vista come una RAM 16x1.

Analizzando l'architettura si evince che le PLD garantiscono la predicibilità delle prestazioni ancor prima di essere implementate mentre in un FPGA le prestazioni dipendono da come il software (sistema di sviluppo) partiziona la logica, la piazza all'interno dei siti a sua disposizione e ne esegue l'interconnessione.

Sia una PLD che un FPGA mettono a disposizione le loro risorse ma non tutte verranno utilizzate dalla logica di utente. In una situazione di funzionamento tipica esisteranno un certo numero di blocchi logici o siti o celle, attivati e un certo numero disattivati.

Il processo di discretizzazione della logica nei possibili siti che la tecnologia mette a disposizione nel silicio si chiama Partitioning. La partitioning appartiene a un mondo virtuale o logico in quanto la logica partizionata non è stata ancora legata ad una coordinata di un sito fisico. Quest'operazione la realizza il piazzamento o Placement sulla base di sofisticati algoritmi che stabiliscono la miglior localizzazione delle celle tra tutte quelle disponibili. Le celle non occupate dalla logica di utente sono da considerarsi superflue. Queste resteranno comunque presenti all'interno del silicio passivamente o attivamente contribuendo, in questo secondo caso e se necessario, alla successiva operazione di interconnesione o Routing tra le celle attive.

5. TIPOLOGIE DI PLD

Le PLD sono l'evoluzione tecnologica e architetturale delle logiche del tipo PAL (Programmable Array Logic) che erano del tipo OTP cioè one-time-programmable, e delle GAL (Generic Array Logic) che si presentarono sul mercato come i primi circuiti riprogrammabili.

Essenzialmente si possono individuare tre categorie. Le One Time Programmable, programmabili da programmatore di eprom; le riprogrammabili, programmabili anch'esse da programmatore di eprom ma cancellabili mediante raggi ultravioletti (tecnologia UV-eprom) o elettricamente (tecnologia EEprom); le riprogrammabili in system, programmabili attraverso microprocessore presente sulla board o un cavo di download e un opportuno software di programmazione residente su computer, oppure ancora mediante programmatore di eprom nei casi in cui possono essere asportate dal circuito stampato.

6. TIPOLOGIE DI FPGA

Gli FPGA non hanno precursori storici, sono stati introdotti nel 1984 come elementi innovativi.

Anche in questo caso è possibile individuare tre categorie. Le One Time Programmable basate sulla tecnologia Antifuse, le riprogrammabili e configurabili obligatoriamente al power on basate sulla tecnologia Static Ram, le riprogrammabili in system non volatili basate sulla tecnologia Flash.

7. ELEMENTI IBRIDI

Per catturare l'attenzione del mercato e accrescere nello stesso tempo le potenzialità delle logiche sono nate soluzioni che da un punto di vista didattico possono essere definite ibride, da un punto di vista commerciale CPLD o PLD con elementi di memoria embedded.

Alla prima categoria appartengono quelle logiche che, abbandonata la struttura delle interconnessioni globali, e pur mantenendo la struttura combinatoria basata sulla somma di prodotti, hanno realizzato strutture di routing raccolte in fasci. Sicuramente ne derivano vantaggi dal punto di vista della potenza dissipata, ma la completa predicibilità delle prestazioni viene sostituita da una predicibilità locale perchè a priori non si sa quale saràil numero di risorse di routing utilizzate e quindi a quanti gruppi di fasci mediamente le interconnessioni apparterranno.

Alla seconda categoria appartengono le logiche programmabili che, mantenendo inalterata la struttura basata sui product term e interconnessioni globali, includono elementi di memoria quali FIFO, Ram Dual Port o Single Port.

8. CRITERI DI SCELTA

I fattori che conducono alla scelta del dispositivo più opportuno per l'applicazione d'utente dipendono dalle specifiche del progetto e dal tipo di mercato in cui confluisce il prodotto. Questi fattori possono essere generali o specifici. Tra i primi si può annoverare la velocità di funzionamento, la capacità e il prezzo per componente. Tra i secondi, sulla base della specifica di funzionamento circuitale e sfruttando le prestazioni che derivano dal tipo di architettura, si può orientare la scelta verso le PLD quando si tratta di realizzare interfacce micro, logica di decodifica, macchine a stati complesse, contatori ad alte prestazioni con un numero elevato di bit di stato. La scelta può ricadere verso gli FPGA quando occorre utilizzare un elevato numero di registri, RAM embedded di dimensioni variabili, e quando si riesce a ben equilibrare la parte sequenziale con quella combinatoria.

Nelle tipiche applicazioni si riscontra mediamente, anche se non è una regola, che per ogni elemento sequenziale occorrono tre elementi combinatori quando si pensa alla struttura a LUT. Quindi, utilizzando questa regola empirica, è possibile dimensionare il componente da utilizzare come numero di celle tre volte superiore al numero dei flip flop del progetto in esame. Nei casi in cui il numero dei livelli di logica necessari ad implementare la funzionalità d'utente come parte combinatoria degrada le prestazioni in termini di velocità, è opportuno ricorrere alle strutture logiche tipiche delle PLD. Queste sono in grado di realizzare in un unico livello di logica quello che in FPGA appesantisce l'elaborazione combinatoria. L'unica via di uscita nel caso in cui si voglia a tutti i costi utilizzare una filosofia LUT based è quella di creare progetti del tipo pipelined cioè compiere il maggior numero possibile di riletture della parte combinatoria, riequilibrando così la differenza tra parte combinatoria e parte sequenziale e puntando alla riduzione massima dei livelli di logica. Per quanto abili si possa essere e tenendo presente che il processo di progettazione non ha una durata illimitata, non sempre si riesce a raggiungere questo obiettivo.

Le azioni da compiere nella scelta del componente target per la applicazione di utente, partono dalla scelta del tipo di logica (FPGA o PLD), della famiglia da utilizzare all'interno del tipo di logica (che deriva, a sua volta, dalla scelta dell'architettura e del fornitore), dello speed grade che caratterizza la velocità intrinseca della tecnologia che si sceglie, del package che riflette il numero di I/O che si desiderano e dello spazio che si vuole occupare sul circuito stampato.

Questa scelta non è sempre una cosa agevole perchè ci sono fornitori che tra famiglie, package e speed grade offrono fino a mille combinazioni. Per ovvie esigenze logistiche non si possono normalizzare in una ditta mille componenti e renderli disponibili tutti a magazzino.

9. FLUSSO DI PROGETTO

La creazione di una logica programmabile passa attraverso un numero di passi che a grandi linee la rendono simile a quella di un gate array.

Si inizia con la fase di Design Concept in cui si descrive cosa si vuole ottenere da un punto di vista funzionale e quali sono le interfacce con il mondo esterno.

La prima fase operativa, in cui vengono utilizzati diversi strumenti software di progettazione CAD, è quella definita del Design Entry, cioè la creazione del progetto vero e proprio. Il Design Entry può avvenire essenzialmente in tre modi diversi interoperabili o meno tra loro. E' possibile realizzare una descrizione della logica d'utente con linguaggi di descrizione di alto livello come il VHDL o il Verilog (HDL - High Description Language) , che implicano il successivo utilizzo di un sintetizzatore di logica. E' inoltre possibile ricorrere ad una descrizione mediante schematico (tipico esempio ViewLogic) oppure mediante equazioni. In questi ultimi due casi non è necessario ricorrere ad un sintetizzatore di logica e quindi ad investimenti aggiuntivi anche se si perdono le enormi potenzialità dei linguaggi di descrizione di alto livello che si hanno in termini di leggibilità, rapidità di progettazione, migrazione di tecnologia, riutilizzo.

La fase di progettazione è assistita da una fase di simulazione funzionale, cioè una simulazione senza tenere conto dei ritardi che introdurrà nel funzionamento reale la tecnologia. E' un passo molto importante perchè consente di valutare la rispondenza della logica progettata alle specifiche di progetto.

La successiva fase definita come Design Processing si può dividere in due processi di cui il primo può essere assente: la sintesi e il fitting.

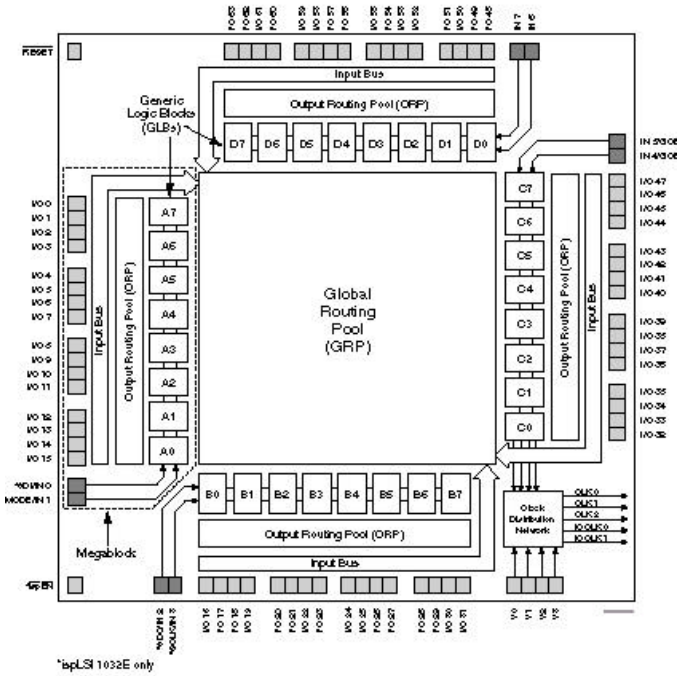
Nella sintesi vengono tradotte le funzioni dell'utente espresse nei codici prodotti dai linguaggi di alto livello in elementi base appartenenti alla tecnologia scelta. Queste informazioni vengono passate mediante un file denominato netlist al fitter, meglio definito come sistema di sviluppo, che effettua il partitioning, place and route e crea il file necessario per la programmazione della logica. Prima di testare il prototipo, viene compiuta, mediante dei programmi messi a disposizione dai sistemi di sviluppo, una analisi statica dei ritardi per valutare le performance del progetto una volta inserito nel componente target. Eventualmente è possibile effettuare una simulazione timing (cioè con i ritardi) per validare ciò che si era visto con ritardi nulli. C'è da tener presente che per progetti sincroni, dopo una attenta simulazione funzionale, una analisi statica dei ritardi condotta con giudizio, è sufficiente a garantire i controlli necessari.

Il ciclo si chiude con la programmazione del componente e il suo test.

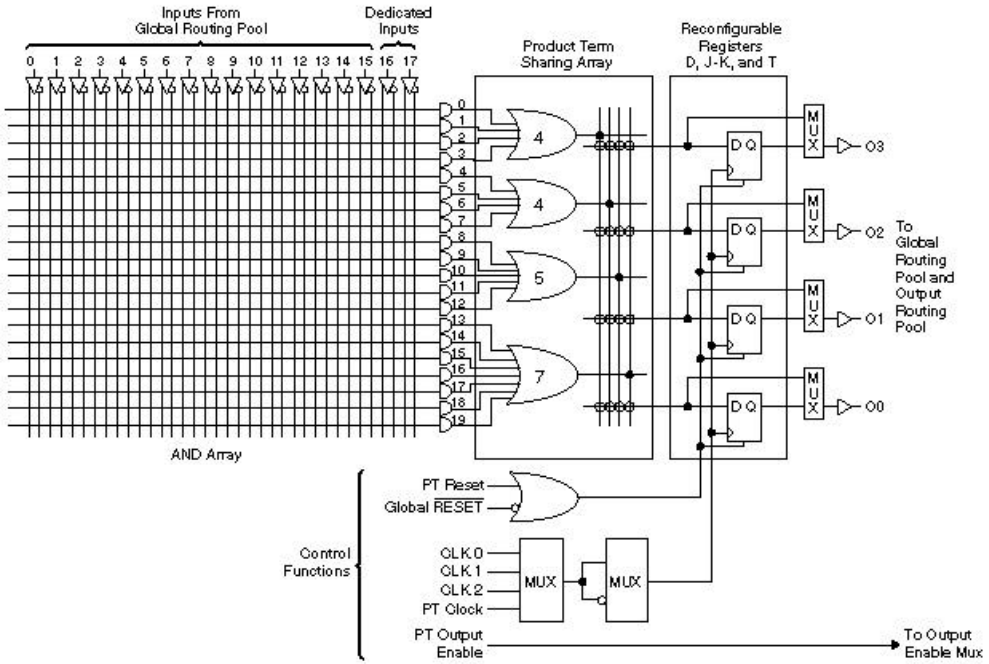
10. LPGA

Si sta evolvendo in questo periodo una tecnologia laser che consente di parlare di LPGA (Laser Prototypes Gate Array), come primo passo verso un gate array programmabile. Lo scopo è quello di fornire in tempi brevi, tipicamente una settimana, un componente con prestazioni, in termini di velocità di funzionamento e densità, simili a quelle di un gate array. Utilizzando la tecnologia QuickSystem (One Time Programmable) si è in grado di fornire in questi tempi un piccolo lotto di produzione per poi, una volta compiute le prove necessarie alla validazione della logica di utente, migrare su grossi volumi utilizzando la tecnologia HardArray.

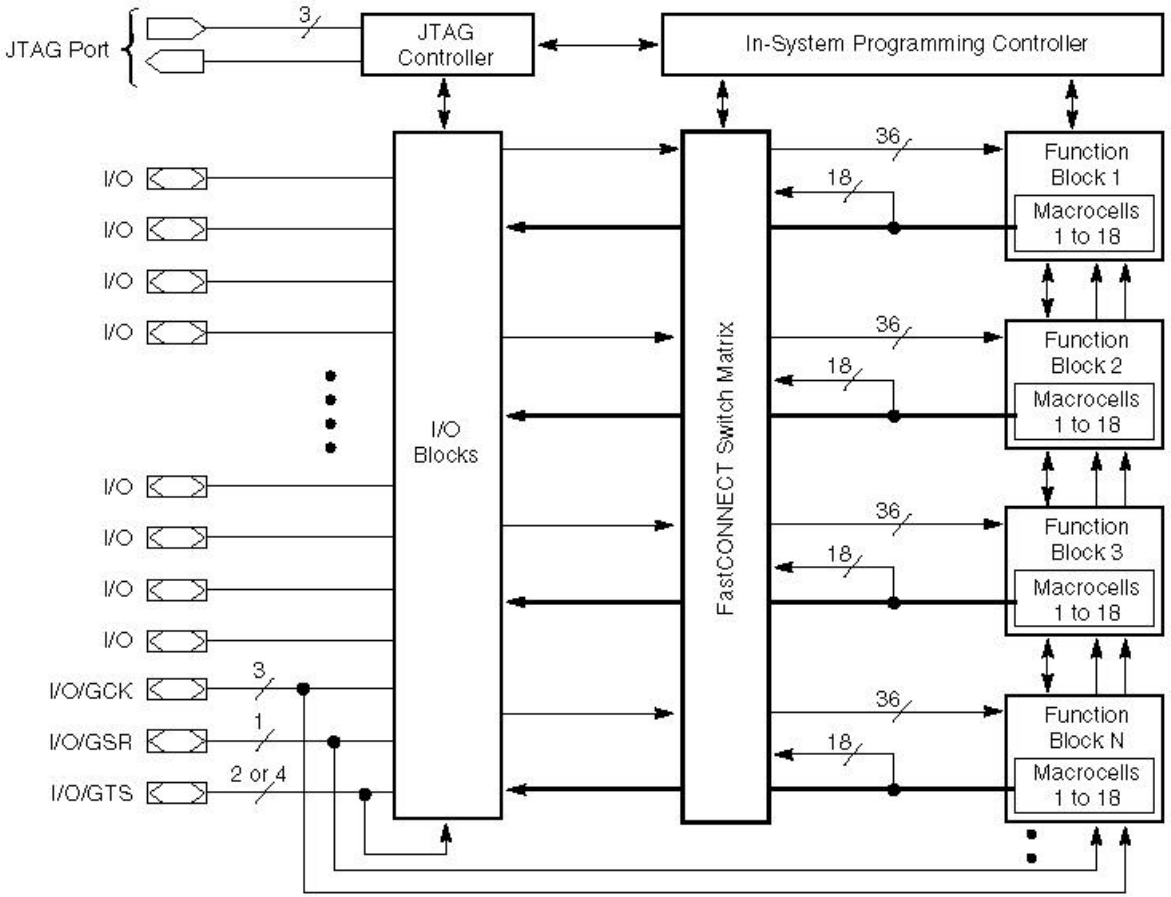
Architettura di PLD con struttura ad interconnessione globale (Lattice Databook)



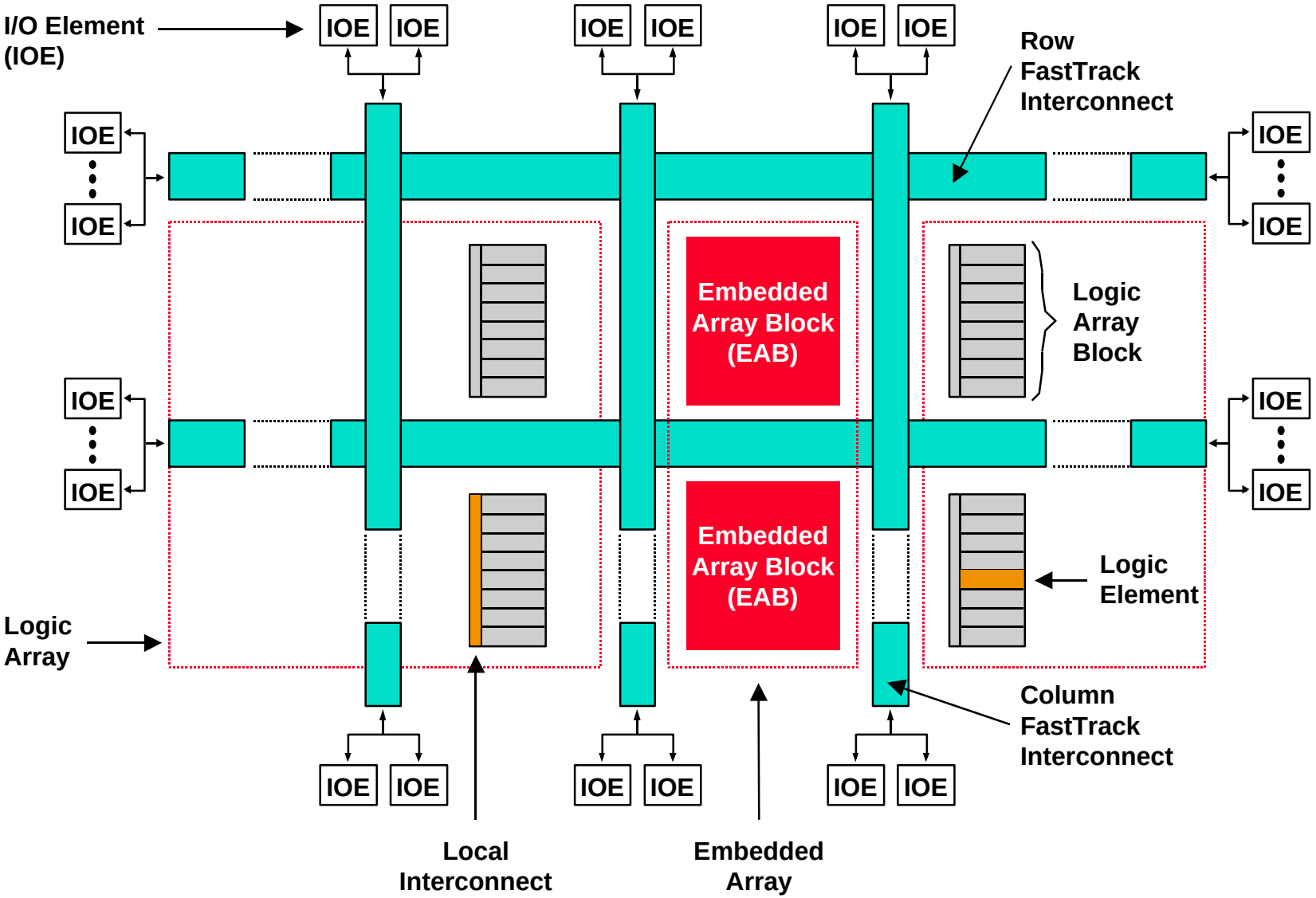
Elemento logico della famiglia di PLD ad interconnessione globale (Lattice Databook)



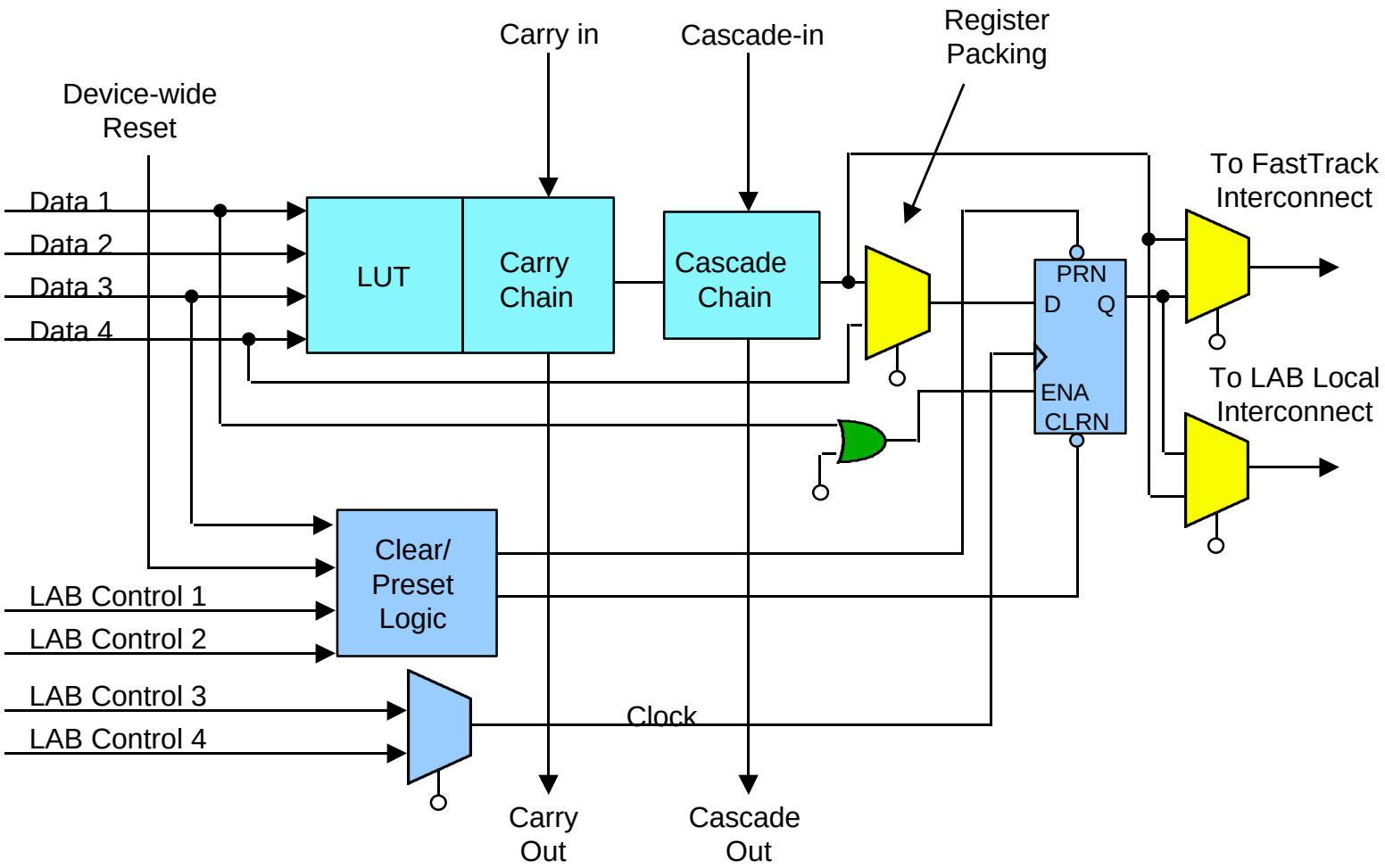
Architettura di PLD della famiglia XC9500 (XILINX Databook)



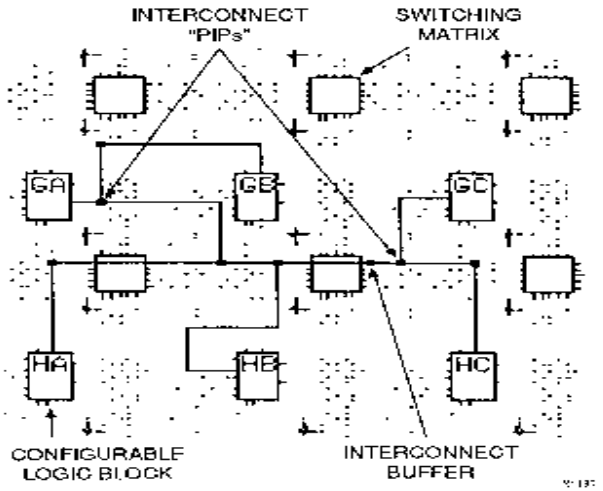
Architettura di FPGA (Altera)



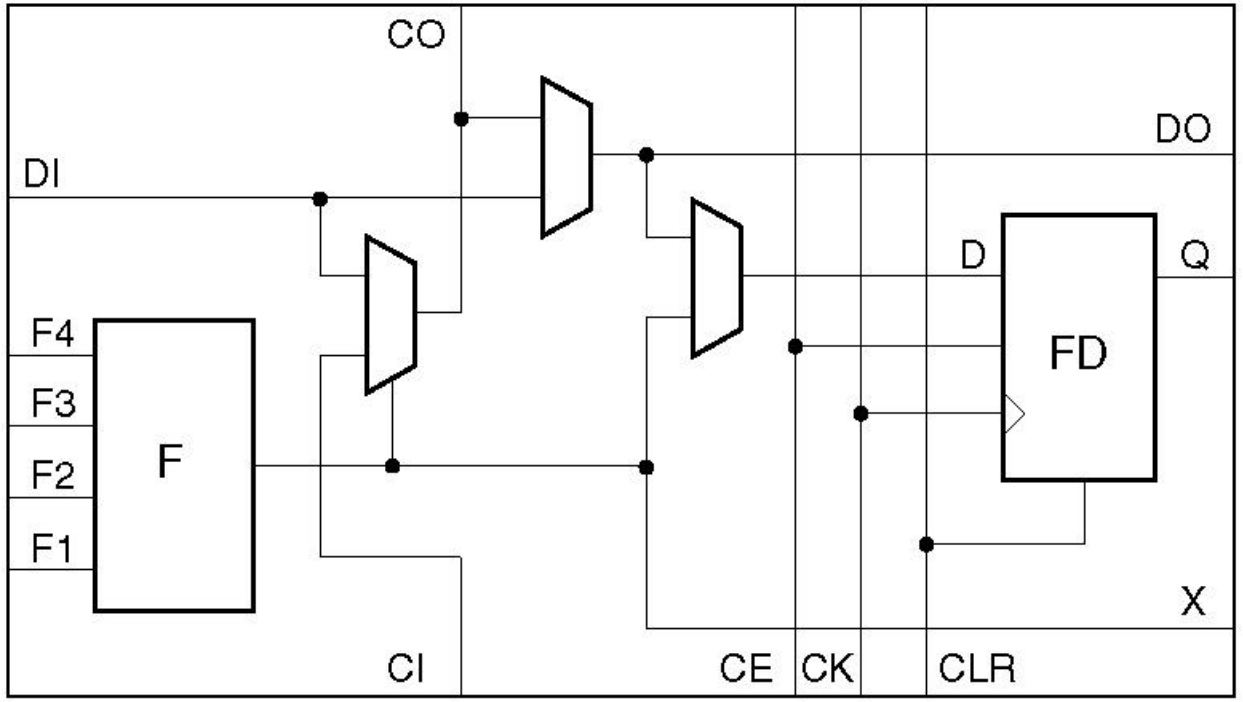
Cella logica della famiglia FLEX 10K (Altera)



Architettura di FPGA (Xilinx Databook)



Cella logica della famiglia XC5200 (Xilinx Databook)



FPGA product selection guide

Company	Family	Technology
Actel	A1200XL, A1400, A3200DX, A42MX, A54SX,	Antifuse
	proASIC	Flash
Altera	Flex 6000, 8000, 10K /A /B /E, Apex 20K	SRAM
Atmel	AT40K	SRAM
Dynachip	DL5000, 6000	SRAM
Lucent	ATT3000, ORCA 2A, 2T, 3C, 3T	SRAM
Quicklogic	pASIC 1, 2, 3	Antifuse
Vantis	VF1	SRAM
Xilinx	XC4000E /EX /L /XL /XLA /XV XC5200, XC3000, XC3100, Spartan /XL, Virtex	SRAM