CAPITOLO 3 : FPGA

1. SCENARIO

Uno studio compiuto dalla società di ricerche di mercato dall'autorevole firma McKinsey & Co. afferma che sei mesi di ritardo nell'ingresso di un prodotto in un mercato altamente competitivo producono un danno pari ad un terzo del potenziale profitto del prodotto nel suo tempo di vita.

Chi produce in un mercato consolidato, in cui il prodotto non evolve rapidamente, non subisce la pressione del time-to-market. Chi si inserirse in un regime ad elevata concorrenza, deve escogitare tutte le possibili strategie al fine di essere competitivo.

La disponibilità di componenti flessibili a diverse applicazioni, un ristretto tempo di sviluppo, piccoli costi iniziali, rischi ridotti in termini di NRE, minori risorse da dedicare alla simulazione, riprogrammabilità sia nell'evoluzione del progetto sia in campo, fornisce numerosi input a chi esamina le strategie da utilizzare per aggredire il mercato con prodotti sempre più complessi e realizzati in un tempo sempre minore.

Gli FPGA (Field Programmable Gate Array), tradizionalmente usati per integrare logica sparsa (glue logic), grazie alle loro caratteristiche che ben si adattano a quelle che richiede il mercato in regime competitivo, sono stati considerati non più come elementi marginali di un progetto, bensì come elementi portanti. E' in tal modo iniziata l'erosione di quelle zone di mercato tradizionalmente di dominio dei gate array.

Questa trasformazione ha condotto, non senza rischi, ad un cambio di mentalità nel modo di realizzare i progetti e proporli al mercato. Infatti l'approccio al mondo dei riprogrammabili con filosofia basata sull'esperienza maturata nel mondo dei gate array è probabilmente la strada più impervia che si possa percorrere. Ma è anche il modo più veloce per rendersi conto che occorre un cambio di impostazione.

2. METRICA

Molti vendor descrivono la capacità di un FPGA in termini di numero di gate, cioè con il numero di nand a due ingressi che sarebbero richiesti per implementare la funzionalità di utente. E' una metrica familiare ai progettisti di gate array e in teoria permetterebbe il confronto, in termini di capacità, tra una logica programmabile e un gate array. Tuttavia un FPGA non è basato sulle nand a due ingressi e questo tipo di confronto non solo è inopportuno ma è anche una fuorviante strategia di marketing. 10000 gate di un FPGA non saranno mai 10000 gate di un gate array a meno di una coincidenza.

La maggior parte degli FPGA è composta da un elemento combinatorio denominato look-up table (LUT) e da un registro ad esso dedicato. La LUT implementa quella che nell'algebra booleana viene indicata come funzione booleana generalizzata di N ingressi. LUT e registro formano una Logic Cell. Ad esempio l'FPGA XC3000 ha 2 logic cell per array element, l'XC4000 2,375, l'XC5200 4, la FLEX10K 8, la FLEX 6000 10, l'ORCA 2C 4.

Confrontando l'XC4028 della Xilinx (2432 logic cell e 28000 gate), la FLEX 10K40 dell'Altera (2304 logic cell e 29000 gate) e la ORCA 2C26 della Lucent (2304 logic cell e 26000 gate) si evince che è più importante conoscere il numero delle logic cell che dei gate.

L'area occupata da un progetto in un FPGA può basarsi, quindi, su una determinazione del numero di flip flop componenti il progetto e di un prudente triplice numero di LUT associate a questi flip flop. Il numero di LUT, essendo nella maggior parte dei progetti superiore al numero dei flip flop, fornirà la base per calcolare il numero degli array element necessari. L'ottimo in termini di sfruttamento delle risorse disponibili è la realizzazione di un FPGA con un numero di LUT pari al numero di flip flop, ma questo è difficilmente realizzabile.

3. FPGA NON STATIC RAM

La tecnologia costruttiva degli FPGA conduce ad una prima suddivisione in termini di singola programmabilitào riprogrammabilità Le tecnologie one time programmable a loro volta possono essere del tipo laser trimmed, Antifuse, EPROM, le altre sono del tipo EEPROM, Flash, static RAM.

L'utilizzo delle tecnologie non volatili può risultare penalizzante nei confronti di quelle che offrono garanzie di riconfigurazione, ma esistono delle applicazioni in cui, grazie alle caratteristiche tecnologiche, le si preferisce alle seconde.

Ad esempio la tecnologia Antifuse non consente clonazioni del silicio programmato e questo nel caso di decoder televisivi può essere un vantaggio. Inoltre sopporta elevate radiazioni elettromagnetiche senza subire alterazioni funzionali, e questo può tornare utile in campo militare e spaziale.

Ma non tutto ciò che vola nello spazio ha bisogno di caratteristiche spinte ed infatti la NASA, a bordo dello Space Shuttle, usa due FPGA static RAM con range di funzionamento commerciale.

Gli FPGA non static RAM si dividono in due categorie: row-based FPGA e fine-grain FPGA. I row-based FPGA sono composti da blocchi di logica organizzata in righe intervallate da canali di interconessione orizzontali programmabili. I fine grain-FPGA basano la loro architettura su cellule tipo sea-of-gates, tipiche dei gate array. L'interconnettività locale è garantita da un primo livello gerarchico. Routing locale e celle formano delle isole a loro volta connesse da una secondo livello gerarchico.

4. FPGA STATIC RAM

La tecnologia Antifuse attraente per i suoi elementi piccoli e veloci da programmare, richiede un grosso numero di transistor intorno ai bordi del chip per fondere il polisilicio amorfo usato negli stessi antifuse. Questi transistor usati durante la programmazione del dispositivo e quelli usati per la testabilità in ogni antifuse minimizzano i vantaggi ottenuti in termini di area con i piccoli elementi da programmare. A questa considerazione si somma quella che per realizzare una struttura tipica di logic cell occorrono un grande numero di antifuse implicando un enorme tempo di programmazione paragonato a quello che si ha nel caso static RAM e un incremento dei costi. Le tecnologie EPROM, EEPROM e Flash soffrono di problemi simili anche se garantiscono una funzionalitànon volatile.

La tecnologia static RAM presenta maggiori semplificazioni da un punto di vista costruttivo, consente un migliore sfruttamento delle risorse del silicio, ha il vantaggio di essere riprogrammabile per un numero infinito di volte.

Un FPGA basato sulla tecnologia static RAM è composto, nella sua configurazione base, da un insieme di elementi logici che ospiteranno la funzionalità programmata dall'utente, da un insieme di interconnessioni che forniscono l'interconnettività necessaria, da porte di I/O che consentono l'accesso al mondo esterno, da linee

predefinite di clock e reset, dalla circuiteria necessaria alla programmazione, dalla memoria di configurazione. L'utilizzo di logica, interconnessioni e porte di I/O è regolato dal contenuto della memoria di configurazione del dispositivo che viene riempita all'atto della programmazione e specializzata in base al tipo di applicazione.

L'elemento base della memoria di configurazione è la Static Configuration Memory Cell. Ciascuna di queste celle controllerà un particolare tipo di risorsa disponibile nell'FPGA.

La cella è composta da due porte CMOS invertenti e un pass transistor che viene attivato per le operazioni di scrittura (programmazione della cella) o lettura (verifica della cella). Durante il normale funzionamento la cella fornisce un continuo controllo all'elemento che indirizza e il pass transistor è aperto evitando eventuali perturbazioni al contenuto immagazzinato dalle porte CMOS (0 o 1 logico). Durante la programmazione il pass transistor offre un percorso ai dati che devono entrare nel circuito di controllo di configurazione composto dalle porte CMOS. Durante la verifica il dato immagazzinato fluisce in direzione contraria.

5. ARCHITETTURA DI UN FPGA

L'array element è la parte preposta alla realizzazione della logica di utente. A seconda delle famiglie e del tipo di costruttore prende il nome di CLB (Configurable Logic Block), LAB (Logic Array Block), PFU (Programmable Function Unit). Ciascun array element può essere visto come insieme di un certo numero di elementi base che prendono il nome di logic cell (LC) o logic element (LE). Questi elementi, a loro volta, sono composti da una LUT che può in certi casi essere usata come ram 16x1 e da un flip flop che in certi casi può essere sostituito da un latch. Tipicamente le LUT hanno 4 ingressi e realizzano quindi funzioni di quattro variabili. In certi casi è possibile estenderne il numero a nove utilizzando due LUT e uno o due multiplexer previsti nell'architettura per tale scopo. L'uscita del segnale da una logic cell può avvenire escludendo o meno il flip flop, oppure pilotando un buffer tri-state interno (in quelle famiglie di componenti che ne prevedono l'esistenza).

Al fine di aumentare le prestazioni in termini di velocità, le LUT, possono essere collegate in modalità Carry Chain. In questa modalità una porzione di LUT genera la somma di due variabili, l'altra un segnale di riporto e produce un segnale di uscita e uno di riporto che alimenterà la successiva LUT. Per aumentare invece le prestazioni in termini di ampio fan-in, le LUT possono essere collegate in Cascade Chain. In questo caso LUT adiacenti sono usate per calcolare porzioni di funzioni in parallelo e la catena viene usata per connettere i valori intermedi. Queste due modalità operative sono gestite dai programmi di sintesi e piazzamento in maniera automatica.

Il clock e il segnale di reset hanno linee a basso skew ed alto fan-out ad essi dedicati. In base al tipo di famiglia scelto è possibile avere due, quattro, otto linee di clock e fino a ventiquattro linee a basso skew da utilizzare sia come ulteriori linee di clock sia come reset o segnali di enable. Tipicamente i pin di clock sono I/O riservati e sono connessi direttamente a queste linee predefinite.

Accanto ai numerosi array element che forniscono dei siti, occupabili o meno dalla logica di utente, gli FPGA possono avere degli elementi embedded di supporto quali blocchi di memoria (single port o dual port), oscillatori interni, ampi decoder di segnali disposti alla periferia del chip, buffer tri-state interni e linee ad essi dedicati, pll interni per minimizzare lo skew sulle linee di clock nel caso di dispositivi di grosso

taglio o aumentare la frequenza interna di 2 o 4 volte rispetto a quella fornita dall'esterno.

Si è parlato di elementi di memoria. Da quanto detto è possibile in sintesi affermare che alcuni FPGA non hanno memoria disponibile, altri consentono di riciclare le LUT come elementi di memoria, e in questo caso si parla di memoria distribuita, altri hanno blocchi di memoria embedded ma di dimensioni rigide (tipicamente 2048x1, 1024x2, 512x4, 256x8, 128x16 bit), altri hanno sia blocchi di memoria embedded che memoria distribuita.

I blocchi di I/O, programmabili da utente, forniscono l'interfaccia tra i pin esterni e la logica interna. Ogni I/O controlla un pin e può essere configurato come input, output, porta bidirezionale. E' possibile programmare le uscite, individualmente e dove la famiglia lo consente, sia in termini di slew rate, sia in termini di corrente di uscita, sia in termini di livello logico (CMOS, TTL, LVCMOS, LVTTL), sia come open-drain, sia dotandole di una pull-up o pull-down. Il segnale entrante o uscente può essere riletto mediante flip flop o latch.

Le interconnessioni o canali di routing stabiliscono le modalità di collegamento tra gli array element o tra un array element e una porta di I/O. Esistono due tipi di interconnessione: segmentata (Segmented Routing Channels) o canalizzata (FastTrack Interconnect).

6. FASTTRACK INTERCONNECT

La struttura di interconnessione FastTrack è tipica dei componenti Altera. Questa modalità di interconnessione si realizza mediante dei fasci di fili raggruppati in righe o colonne che attraversano il componente in tutta la sua dimensione. Esistono interconnessioni a tre livelli tipiche delle famiglie FLEX6000, FLEX8000, FLEX10K, e interconnessioni a quattro livelli tipiche della famiglia APEX20K.

Nelle strutture ad interconnessione a tre livelli i Logic Element sono organizzati in gruppi di otto o dieci per formare un blocco denominato Logic Array Block (LAB). I LAB sono organizzati in forma di matrice di righe e colonne e tra questi sono inseriti dei canali di routing sia orizzontali che verticali che forniscono le necessarie connessioni. L'interconnettività all'interno di un LAB è fornita da canali di interconnessione locali denominati LAB Local Interconnect. Il passaggio da colonna a riga è compiuto mediante dei multiplexer programmati in fase di configurazione e selezionati, in base al tipo di routing compiuto, mediante le celle static RAM della memoria di configurazione.

Quando un LE di un LAB deve essere connesso a un LE di un qualsiasi altro LAB del dispositivo il segnale viene immesso in un canale della colonna che lambisce il LAB, poi in un canale della riga che serve il LAB destinatario e quindi raggiunge il LE mediante un canale del Local Interconnect. I canali usati per questo collegamento appartenenti sia ad una riga che ad una colonna, anche se in misura minore rispetto alla loro lunghezza totale, non possono più essere utilizzati per un altro collegamento.

Se il LAB destinatario appartiene alla stessa riga del sorgente è possibile evitare il passaggio mediante colonna entrando direttamente nella riga. Se il LE sorgente appartiene allo stesso LAB del LE destinatario allora è possibile evitare anche il passaggio della riga per compiere direttamente la connessione mediante l'interconnessione interna del LAB.

Nella famiglia FLEX6000 il passaggio da un LAB a quello adiacente e sulla medesima riga avviene sfruttando il LAB Local Interconnect che in questo caso può essere condiviso da due LAB adiacenti.

Nella famiglia APEX20K sono previsti quattro livelli di interconnessione. Dieci Logic Element sono raggruppati in un LAB e sedici LAB e un blocco embedded ESB (Embedded System Block) sono raggruppati insieme per formare un MegaLAB. L'ESB può essere una ram, una rom, una cam, un blocco di sedici macrocelle basate sulla struttura a product term, xor e flip flop. L'interconnessione tra i LAB adiacenti è fornita dal Local Interconnect in maniera simile a come avviene nella FLEX6000, mentre l'interconnessione all'interno del MegaLAB è garantita dal MegaLAB Interconnect. All'interno del componente i MegaLAB sono connessi mediante canali di routing costituiti da righe e colonne.

Sia per le interconnessioni a tre livelli che a quattro, alcuni blocchi di I/O sono direttamente connessi con i canali verticali, altri con quelli orizzontali.

7. ROUTING SEGMENTATO

La realizzazione di una simile rete di interconnessione si basa su elementi di interconnessione o segmenti, su una struttura gerarchica di questi e su matrici di switch che garantiscono il congiungimento di particolari segmenti tra quelli possibili.

La famiglia XC5200 della Xilinx si presenta come valido esempio per la descrizione di una struttura di routing gerarchico segmentato. Infatti in questo tipo di componente si possono individuare sei livelli di routing.

Questa famiglia è composta da un certo numero di celle di I/O programmabili, da blocchi di logica programmabile e da interconnessioni programmabili. Le interconnessioni sono di tipo distribuito e di tipo locale. L'insieme delle interconnessioni locali e della logica forma il Versablock. Quelle distribuite nel componente connettono i Versablock attraverso delle matrici di switch definite General Routing Matrix (GRM).

Un Versablock è composto da un Configurable Logic Block (CLB) e una struttura di routing che lo circonda. Ogni CLB ha quattro logic cell (LC) contenente ciascuna un function generator a quattro ingressi, un elemento di memoria, la logica di controllo ad essa associata. Per ogni LC ci sono 5 ingressi e tre uscite. Quando viene utilizzata una LUT come parte combinatoria è possibile utilizzare il flip flop per rileggere il dato presente sul quinto ingresso della LC o fornire una connessione feedthrough tra quest'ingresso e una uscita. Questa connessione che sfrutta la Logic Cell per effettuare il routing costituisce il primo livello della gerarchia di routing ed è anche quella a ritardo minore.

Costruiti intorno alle quattro Logic Cell esistono delle strutture di routing preposte a garantire la completa connettività degli ingressi e delle uscite di ogni LC in un dato CLB. Queste interconnessioni che appartengono al secondo livello della gerarchia si chiamano Local Interconnect Matrix (LIM).

La connettività tra CLB adiacenti, ricalcando uno schema a croce, e senza fare ricorso a risorse di routing di livello superiore, è assicurata dalle Direct Connect. Queste linee di routing dedicate tra CLB adiacenti unite a quelle dedicate all'interno dei singolo VersaBlock dette LIM consentono la minimizzazione dei problemi locali di congestione del routing e la possibilità di creare zone di logica ad alta velocità in cui il ritardo dovuto alle connessioni è minimo.

Quando il VersaBlock deve essere connesso ad altri non adiacenti secondo uno schema a croce, il LIM viene collegato al General Routing Matrix mediante un fascio

di connessioni. Oltre alle connessioni provenienti dal LIM affluiscono nel GRM vari segmenti di differente lunghezza provenienti sia orizzontalmente che verticalmente da altri GRM. In base alla destinazione da raggiungere i segmenti sono connessi elettricamente da un pass transistor denominato Programmable Interconnect Point (PIP) a sua volta controllato da un elemento della memoria di configurazione.

Possono entrare in un GRM segmenti del tipo Single-length, Double-length, Longline che in ordine risalgono la gerarchia del routing. Le prime connettono due GRM adiacenti, le seconde per diminuire i ritardi di attraversamento che inevitabilmente si incontrano nelle matrici di interconnessione, ne connettono due non adiacenti. Le longline infine, sono usate per segnali ad elevato fan-out, tri-state buffer, connessioni a basso skew, propagazione di segnali per destinazioni lontane nel componente. Queste attraversano interamente il die e sono raggruppate in fasci orizzontali e verticali.

La rete dedicata a segnali globali, chiamata Global Line, può essere vista come un ulteriore livello della gerarchia di routing. Questa rete è pilotata da buffer di tipo globale ed è pensata per il trasporto di clock o altri segnali di controllo al fine di massimizzare la velocità e minimizzare gli skew mentre si distribuisce il segnale a diversi CLB sparsi nel componente. Le Global Line raggiungono i GRM e forniscono i segnali di clock, reset, enable per i flip flop delle LC oppure possono alimentare i function generator.

Sia nel caso delle longline che in quello delle global line la connessione verso i GRM è di tipo monodirezionale con la direzione del segnale entrante nei GRM. L'unico modo possibile per raggiungere una longline è attraverso i buffer tri-state di cui è dotato il CLB. Quindi un collegamento longline-longline o single-line-longline attraverso i Programmable Interconnect Point del General Routing Matrix non è possibile.

Un fascio di interconnessioni che circonda il die alla sua periferia e che prende il nome di Versaring, è la soluzione ai problemi di interconnessione della logica interna ai pad di uscita nei casi di pin locking.

Una volta descritta la struttura a routing segmentato di questo tipo di famiglia quello delle altre è abbastanza simile. Nella famiglia XC4000 della Xilinx non c'è il LIM ma continuano a sussistere le Direct Connect che evitano di effettuare il routing mediante matrice di interconnesione che in questo caso si chiama Programmable Switch Matrix (PSM). Questa matrice connette segmenti del tipo Single, Double, Quad cioè segmenti che scavalcano tre PSM prima di entrare in una quarta. Sono inoltre presenti le longline e le global line. Il numero di tutti questi segmenti che affluiscono in un PSM dipende dalla grandezza del componente scelto.

La stessa struttura gerarchica, con modifiche di tipo evolutivo, si ha nella VIRTEX. Viene ripresentata la struttura di interconnessione locale simile al LIM con il nome Local Feedback. Le Direct Connect connettono solo CLB adiacenti sulla medesima riga e non più con disposizione a croce. Al posto delle Double e Quad Line ci sono le Hex Line, restando ancora valide le Single Line. Il PSM, che qui prende il nome di Switch Matrix, subisce una evoluzione, passando dalla modalità di commutazione denominata Planar Pipulation a quella Non Planar Pipulation. Questa evoluzione permette il routing, non più tra segmenti di una medesima gerarchia, bensì tra gerarchie diverse. Così le Longline possono confluire in Hex Line, le Hex Line in Hex o Single Line.

8. CONFRONTI

Non utilizzando strutture di switch matrix gli FPGA basati sulle FastTrack Interconnect hanno minore complessità nella struttura del routing sia in termini costruttivi che in termini di complessità del software che deve effettuare il routing stesso. Inoltre le prestazioni sono predicibili con maggiore accuratezza. Volendo ad esempio connettere dei segnali uscenti da flip flop ad un altro flip flop la cui parte combinatoria è confinata all'interno di un solo LAB, nel peggore dei casi il routing è riconducibile ad un attraversamento di una colonna, una riga e una struttura di interconnessione locale.

Gli FPGA basati sulla struttura a routing segmentato utilizzano meno silicio dedicato ai collegamenti, perchè le righe o le colonne sono sostituite da segmenti ed inoltre consentono di avere consumi più contenuti. Infatti, a parità di tensione di alimentazione e frequenza di funzionamento, il carico capacitivo, superiore per linee lunghe utilizzate per intero anche quando non è necessario, aumenta la potenza assorbita dal componente ($P= C \times f \times V^2$).

9. CONFIGURAZIONE

La configurazione è il processo che si occupa di caricare i dati di programmazione specifici di un'applicazione, in uno o più FPGA disposti in daisy chain, definendo le operazioni funzionali dei blocchi interni e le interconnessioni necessarie. Ogni bit di configurazione definisce lo stato di una cella di memoria static ram che ha il compito di specializzare la look-up table alla realizzazione della funzione desiderata, selezionare l'ingresso di un multiplexer di controllo presente all'interno delle celle logiche, gestire un pass transistor per la realizzazione delle interconnessioni.

Un FPGA può essere programmato in diversi modi e i dati necessari alla programmazione possono essere immagazzinati in memorie OTP o riprogrammabili, seriali o parallele. La programmazione avviene tipicamente utilizzando cinque o sei segnali che gestiscono il processo. Appartengono a questi segnali i dati inviati verso l'FPGA, quelli che escono da questa (che servono per le operazioni di verifica o di programmazione in catena), il clock associato ai dati, un segnale che indica la fine della programmazione, un segnale che indica se si è verificato un errore nella programmazione, un segnale di riprogrammazione.

Se l'FPGA genera il clock e riceve i dati, in seriale o in parallelo, la si considera in modalità di programmazione Master (Master Serial o Master Parallel o, in altre dizioni, Active Serial o Active Parallel). Se riceve il clock e i dati la si suppone in modalità Slave (Slave Serial o Slave Parallel o, analogamente, Passive Serial o Passive Parallel). In una catena daisy chain in cui il primo elemento può essere Master o Slave, programmato in seriale o in parallelo, gli altri elementi che la compongono sono sempre configurati in modalità Slave. Esistono tre pin, definiti come pin di modalità di configurazione, che indicano all'FPGA, all'inizio della programmazione, come questo deve aspettarsi i dati e se deve generare o ricevere il clock.

Quando si hanno diversi FPGA da programmare in catena le possibilità di programmazione sono due. Accedendo ad una memoria dedicata, un FPGA che funge da Master genera il clock e preleva i dati dalla memoria. Questi dati serviranno alla sua programmazione e a quella delle altre che sono configurate in modalità Slave. In alternativa è possibile utilizzare una porta di I/O di un microprocessore con i dati di programmazione residenti in una porzione della memoria gestita dal micro stesso. Tutti gli FPGA sono configurati in modalità Slave e si aspettano di ricevere il clock e i dati dalla porta del micro. Il microprocessore dovrà

eseguire un opportuno programma di scrittura dei dati prelevati dall'opportuno spazio di memoria, di generazione del clock e di manipolazione dei segnali necessari allo svolgimento del protocollo di programmazione, come inizio, fine, errore di programmazione.

E' possibile programmare il componente con un cavo di download, ma questa opportunitàè valida solo per scopi di debug, perchè allo spegnimento della piastra la memoria di configurazione perde il suo contenuto. Sempre utilizzando questo cavo, è possibile compiere una operazione definita, in certi casi, readback. Con questa operazione si accede al contenuto della memoria di configurazione e si determina il livello dei nodi interni senza alterare le normali operazioni. Si è quindi in grado di conoscere il contenuto di tutti i flip flop o latch delle celle logiche o degli I/O e delle RAM interne.

10. SEQUENZA DI PROGRAMMAZIONE

La programmazione di un FPGA attraversa cinque fasi che si seguono in ordine temporale e che si possono così riassumere: test della tensione di alimentazione, azzeramento della memoria di configurazione, inizializzazione, caricamento dei dati, entrata in funzione o Start-Up.

Nel momento in cui si fornisce la tensione di alimentazione all'FPGA, un circuito interno inizia una fase di monitoraggio su questa, fino al raggiungimento della tensione minima di funzionamento. Raggiunto questo valore viene compiuta una scrittura e lettura di prova in una Static Configuration Memory Cell. L'esito positivo di questa operazione avvia un timer che ha lo scopo di attendere la stabilizzazione della tensione di alimentazione. Durante il normale funzionamento dell'FPGA viene costantemente compiuto un monitoraggio sulla tensione. Se questa scende oltre i limiti di minimo funzionamento viene riavviata la procedura come se ci fosse stata una accensione.

Una volta trascorso questo ritardo viene compiuta la fase di azzeramento della memoria di configurazione. Questo azzeramento è opportuno non solo in fase di accensione ma anche in fase di riprogrammazione per garantire che il nuovo gruppo di dati non si sovrapponga a quello giàpresente in memoria.

A questo punto si ha la fase di inizializzazione in cui, se ci sono diversi FPGA connessi in catena con differenti tempi di cancellazione della memoria, si attende il completamento delle operazioni da parte di tutti. Avvenuta questa operazione, ciascun componente campiona lo stato dei pin che definiscono la modalità di configurazione (Master, Slave, Parallel, Serial) e si predispone all'arrivo dei dati.

I dati, definiti anche bitstream, contengono un preambolo, per consentire un'operazione di allineamento, un campo che definisce il numero dei bit di configurazione, e un certo numero di trame contenenti i bit di configurazione. Queste sono precedute da uno start byte e seguite da un crc per il controllo, trama per trama, del corretto svolgimento delle operazioni. Con l'arrivo del postambolo e se non si sono verificati errori, l'FPGA segnala, su un pin dedicato allo scopo, che l'operazione di programmazione è andata a buon fine.

Le operazioni di programmazione si concludono con la sequenza di Start-Up. L'FPGA passa dal clock di programmazione a quello di scheda e i pin usati per la programmazione diventano, senza generare contese, pin di utente. Gli altri pin, che durante la programmazione erano in tri-state, diventano attivi. Tutti i flip flop interni tenuti in uno stato di global reset o set divengono operativi contemporaneamente.

FIGURA 1: Architettura di un FPGA. Si evidenziano blocchi di logica, I/O, area di interconnessione, memoria di configurazione composta da static configuration memory cell. Questa cella a sua volta e' composta da un pass transistor e due porte CMOS invertenti (Xilinx Data Book).

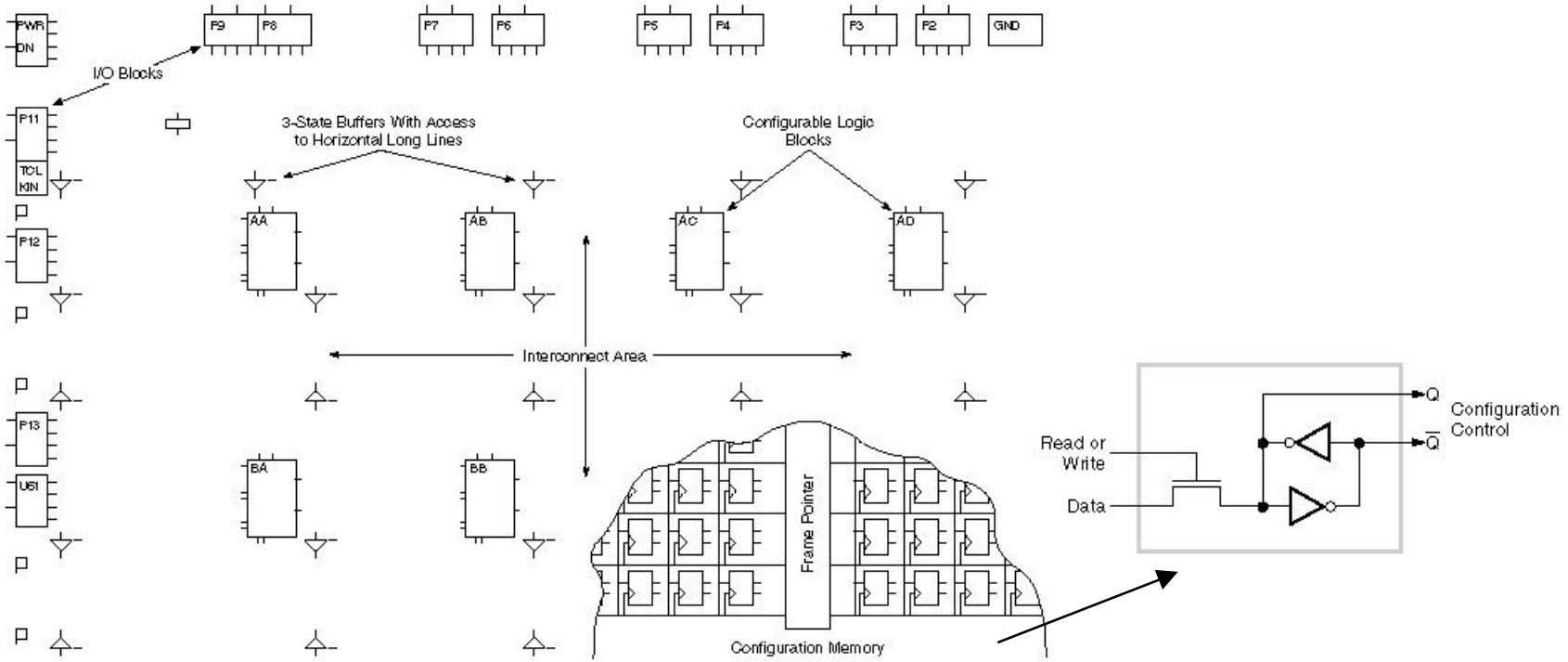


FIGURA 2: Cella logica composta da una LUT e un registro. E' presente un bypass di LUT e uno di Flip Flop (Altera FLEX10K).

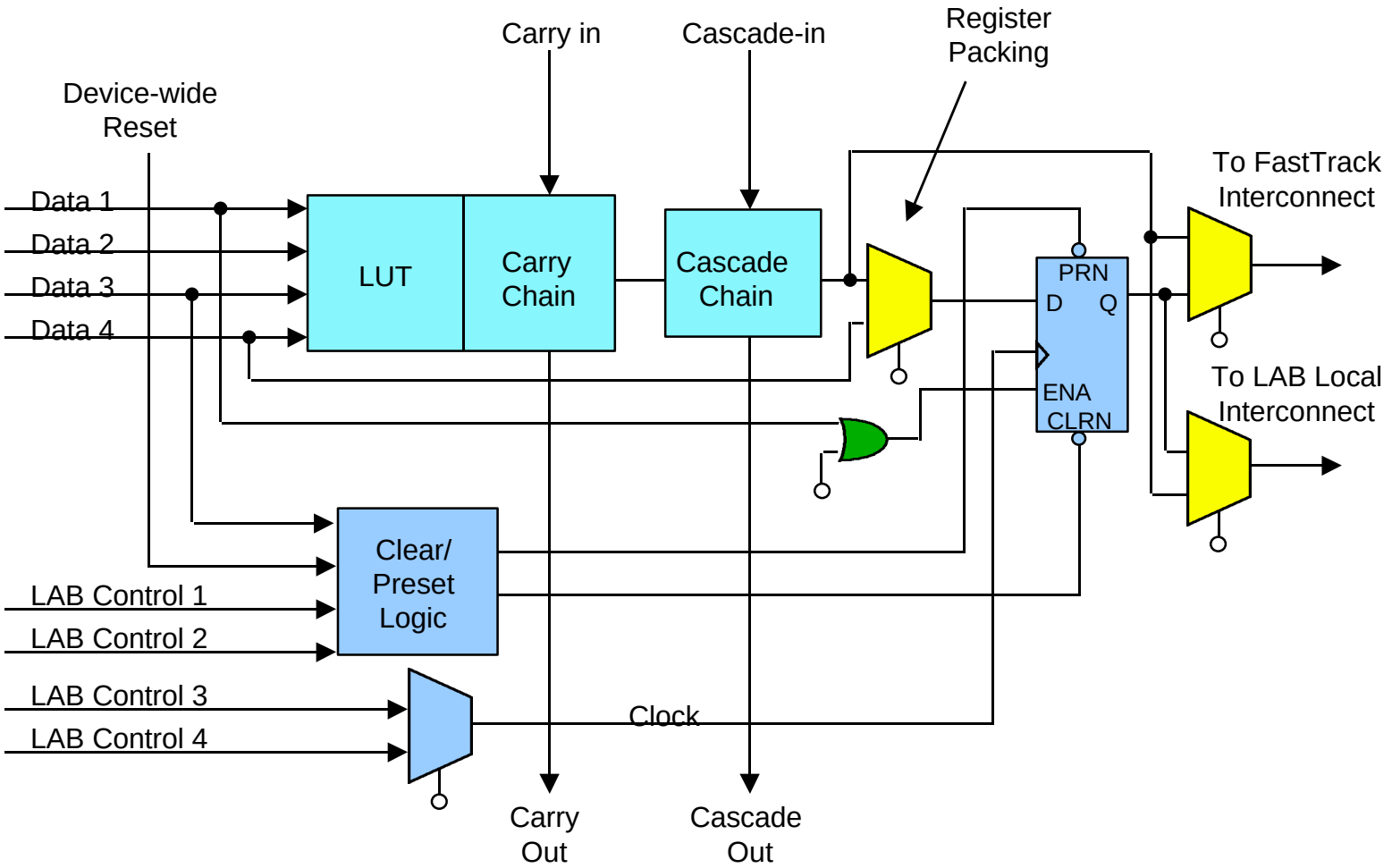


FIGURA 3: Interconnessioni del tipo FastTrack. Il Local Interconnect e' in questo caso condiviso da due LAB adiacenti (Altera Data Book).

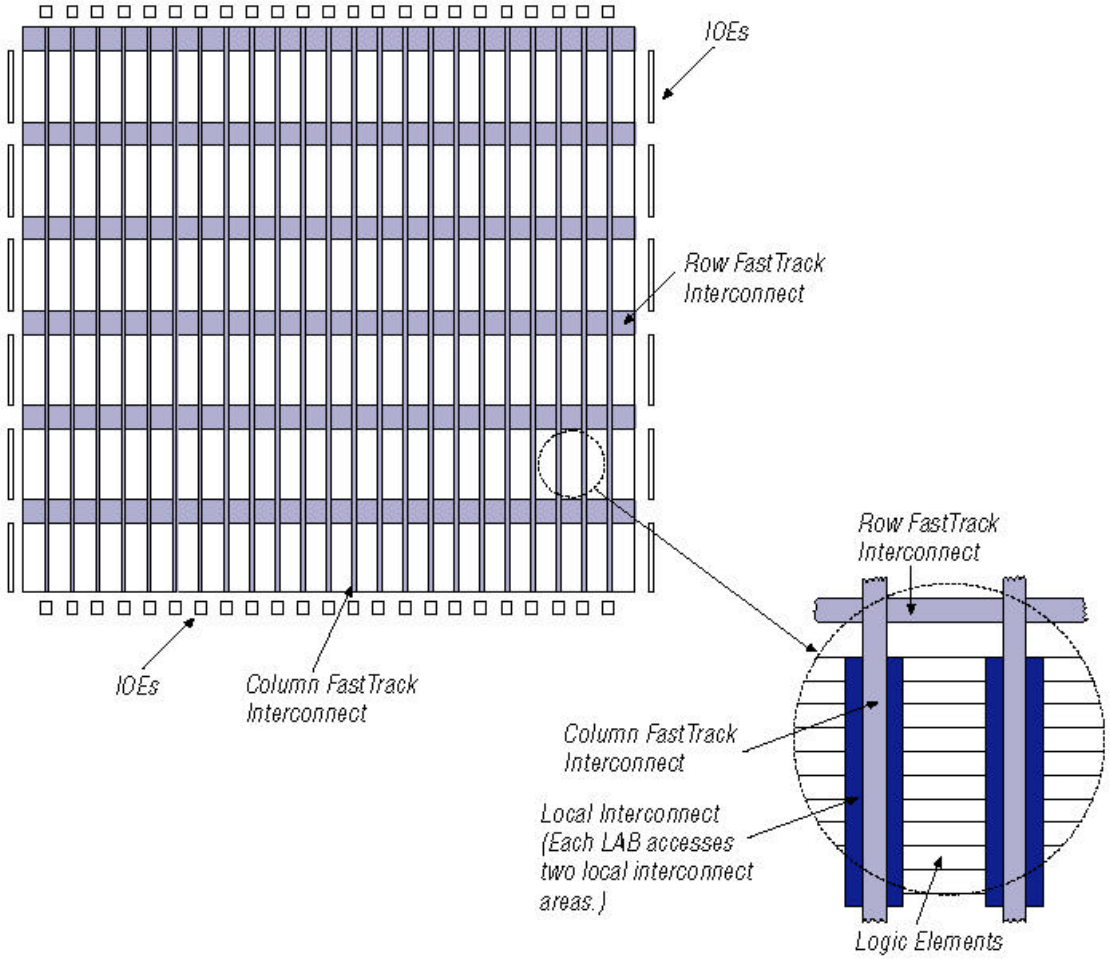


FIGURA 4: Struttura di un LAB e modalita' di interconnessione tra righe e colonne (Altera FLEX10K).

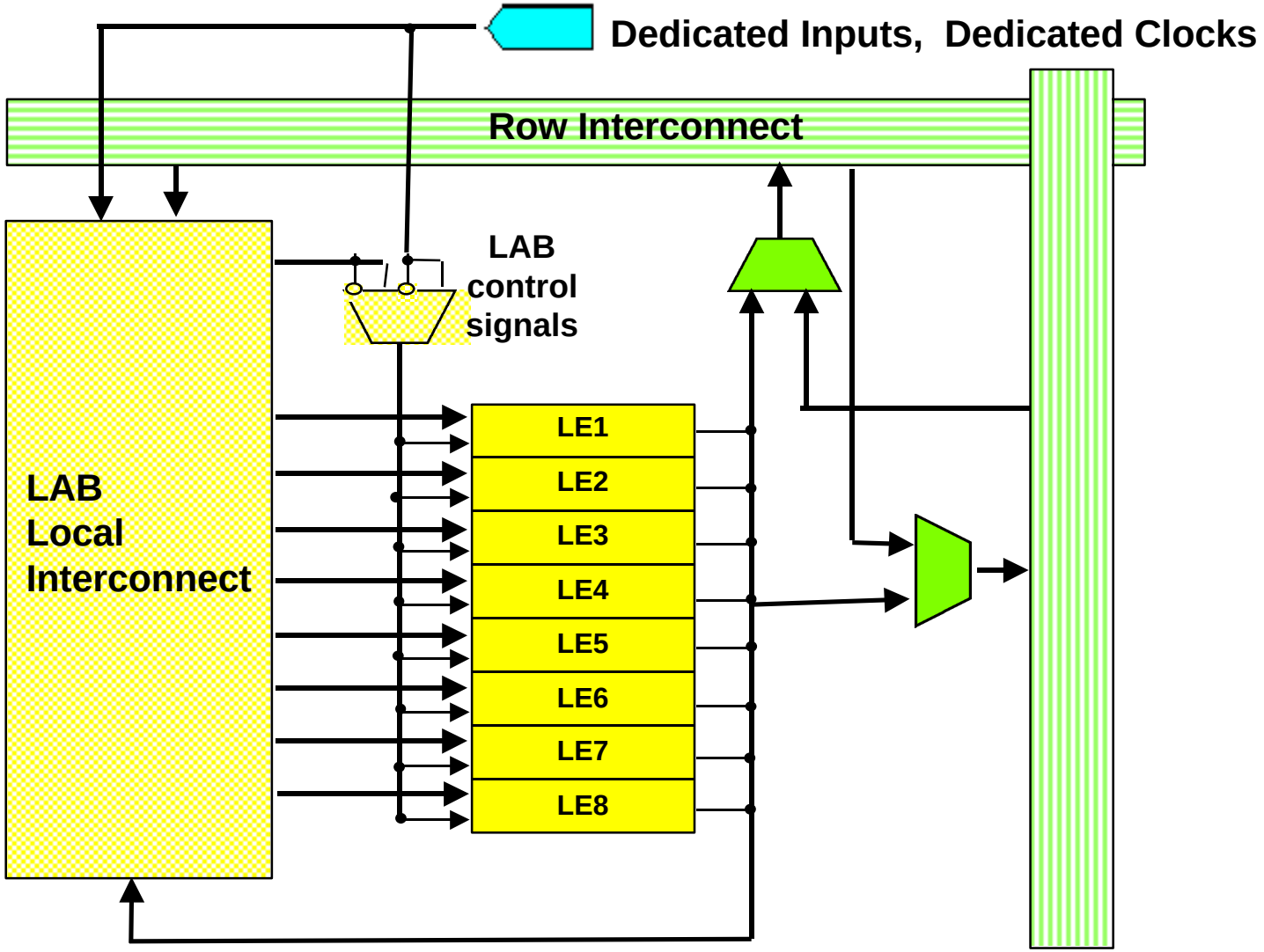
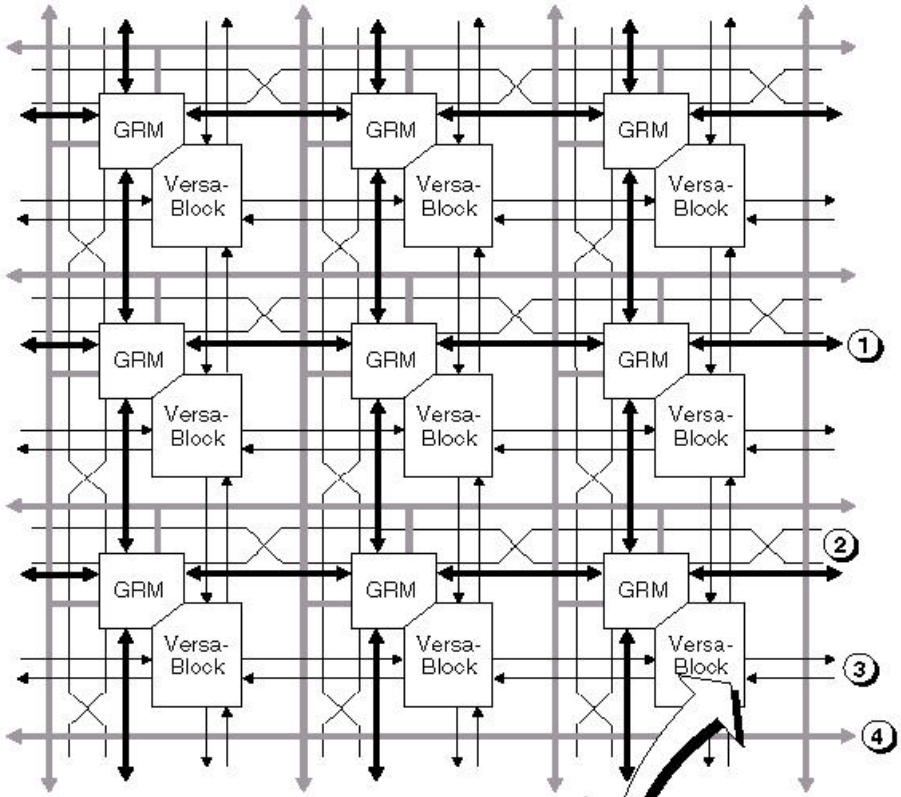


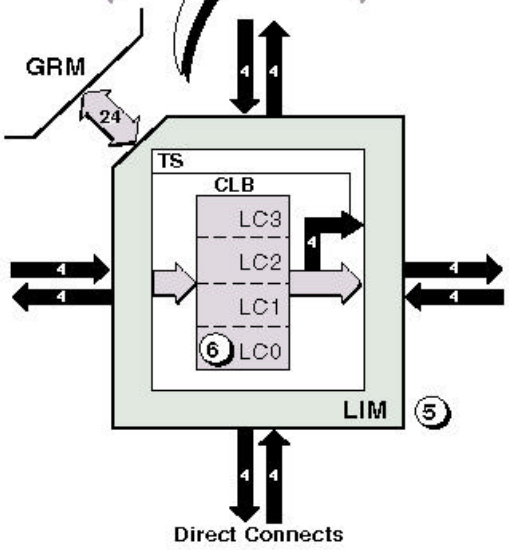
FIGURA 5: Routing segmentato gerarchico della famiglia XC5200 e Versaring per gli I/O (Xilinx Data Book).

la figura e’ nella pagina seguente



Six Levels of Routing Hierarchy

1	⟷	Single-length Lines
2	⨉	Double-length Lines
3	⟵⟶	Direct Connects
4	⟷	Longlines and Global Lines
5	LIM	Local Interconnect Matrix
6		Logic Cell Feedthrough Path (Contained within each Logic Cell)



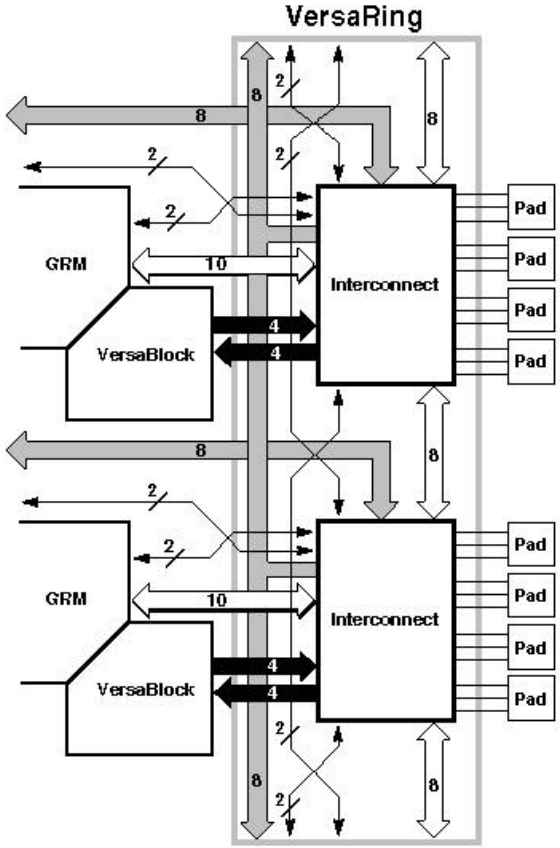


FIGURA 6: LUT usata come feedthrough (segmento blu) e come funzione combinatoria (segmento rosso) (Xilinx Data Book).

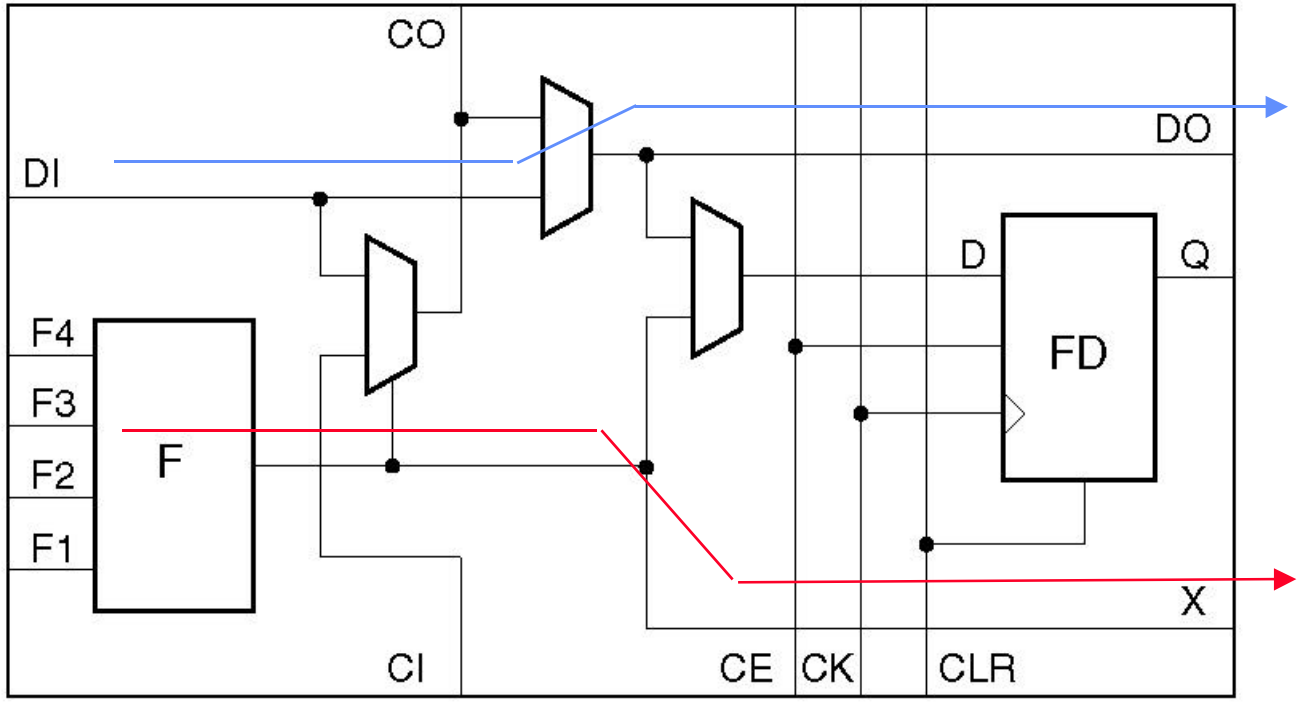


FIGURA 7: Daisy Chain con un FPGA Master, due Slave e una eprom seriale contenente il file di programmazione multiplo (Xilinx Data Book).