# Punto di Vista di...


## Nick Martin

Fondatore e Ceo - Altium


# Progettare e realizzare sistemi embedded all'interno di un Fpga

Quando, negli anni '70, i microprocessori a 8 bit sono divenuti dispositivi commodity, il mondo della progettazione elettronica ha subito una rapida trasformazione. La possibilità di realizzare parte di un progetto via software - mediante la programmazione del processore - ha avuto un impatto notevole. Al giorno d'oggi, qualsiasi porzione di un progetto che può essere implementata via software può essere modificata anche una volta che il prodotto è stato costruito. Nel corso degli anni questi dispositivi programmabili hanno raggiunto livelli di prezzi tali da consentirne l'impiego nelle più svariate applicazioni.

La semplicità, in fase di sviluppo e di collaudo, offerta da questo ambiente software ha portato a sensibili incrementi in termini di produttività, a un notevole miglioramento delle funzionalità che in ultima analisi si è trasformata nella possibilità di realizzare prodotti un tempo impensabili.

Nei decenni scorsi società come Xilinx, Altera e Actel hanno investito centinaia di milioni di dollari nello sviluppo di Fpga, ovvero dispositivi hardware programmabili. Verso la fine degli anni '90 l'accesa competizione tra i principali protagonisti di questo mercato ha contribuito da un lato a migliorarne le capacità e dall'altro a diminuirne il prezzo.

La sfida si sta ora concentrando sugli Fpga a elevata densità e a basso costo realizzati con processi da 90/130 nm: tali componenti forniscono opportunità del tutto analoghe a quelle offerte in passato dai micro a 8 bit. Attualmente è possibile integrare un sistema embedded completo all'interno di un Fpga a un costo inferiore a 20 dollari. L'avvento di tali Fpga consente di trasferire la maggior parte di un progetto in un ambiente hardware riprogrammabile via software.

Non va comunque dimenticato che affinché tali sistemi programmabili possano affermarsi come una piattaforma tecnologica di più ampia diffusione è necessario superare parecchi ostacoli che impediscono l'implementazione di un sistema in questo nuovo "spazio nanometrico" programmabile.

Prima di affrontare i problemi legati alla progettazione di questi "sistemi su Fpga" è utile esaminare pregi e difetti legati al trasferimento di questi sistemi su piattaforme programmabili al fine di valutare la razionalità di tale operazione. Sulla base di esperienze personali, un elemento da tenere in considerazione nel momento in cui si lavora su un progetto basato su Fpga è la possibilità di sviluppare buona parte del design prima di prendere la decisione finale per quel che riguarda il processore, le periferiche e la suddivisione tra l'implementazione hardware e software di una data funzione. Si tratta, per certi versi, di un approccio radicalmente diverso rispetto a quello di tipo top down.

Anche dopo che il progetto è stato trasportato sull'hardware, è possibile modificare la suddivisione tra le componenti hardware e software, con conseguente aggiornamento del progetto hardware. Con questo modello l'obiettivo è trasferire la maggior parte del progetto sull'Fpga per garantire la massima flessibilità nel momento in cui vengono apportate modifiche al progetto.

Un altro significativo vantaggio dei sistemi embedded basati su Fpga è la possibilità di variare la definizione del core del processore in modo da permettere il trasferimento di determinati compiti, eseguiti via software dal processore principale, a un coprocessore hardware.

Si tratta di compromessi che possono venire modificati nel corso dello sviluppo di un sistema, in modo da permettere l'ottimizzazione delle prestazioni sulla base delle osservazioni di sistemi che operano nel mondo reale.

L'adozione di un modello di sviluppo indipendente dal dispositivo e dal costruttore permette di procedere all'aggiornamento di un prodotto utilizzando un nuovo Fpga al fine di ottenere maggiori prestazioni o conseguire risparmi in termini di costi, senza per questo dover apportare variazioni a livello di astrazione del design.

La possibilità di utilizzare una differente versione di un processore, o addirittura un processore completamente diverso, rappresenta un ulteriore potenziale vantaggio delle piattaforme riprogrammabili. Questo processo può essere effettuato in

## Punto di Vista di...

**Nick Martin**

maniera quasi completamente trasparente dal punto di vista software, e in modo del tutto trasparente a livello hardware.
Una volta che l'applicazione basata su Fpga si sposta verso il mondo dei 32 bit, il costo può iniziare a divenire un elemento critico. Nel momento in cui gli Fpga a elevata densità diventano più economici, le soluzioni di tipo ibrido e "soft core" rientrano nella medesima fascia di prezzo dei processori dedicati. Il concetto che sta alla base è che, una volta pagato l'Fpga, ogni IP aggiuntivo integrato nel dispositivo si trasforma in una funzionalità gratuita. Occorre poi considerare i rischi, percepiti o reali, che ostacolano l'utilizzo degli Fpga. La mia personale opinione è che sarebbe utile poter disporre di un modello semplice e intuitivo al posto delle complesse metodologie di progetto (basate sulla verifica) adottate nei progetti di Asic e Soc. Un approccio alternativo diminuirebbe sensibilmente i rischi legati allo sviluppo di un sistema basato su componenti programmabili e consentirebbe di apprezzare "in toto" i vantaggi legati all'adozione di una piattaforma di natura configurabile.
Quando all'inizio è stato descritto l'avvento del microprocessore come un cambiamento di notevole entità per il mondo dell'elettronica mi riferivo a una tecnologia che aveva consentito di spostare problematiche complesse a un livello di astrazione più alto. Il solo fatto di integrare una logica complessa in un singolo dispositivo ha consentito di utilizzare quest'ultimo senza doverne conoscere necessariamente i dettagli. Nel giro di pochi anni gli ingegneri hanno potuto lavorare con parecchi milioni di elementi logici integrati in sistemi parecchio complessi considerando solamente la piedinatura, a livello quindi di componente: una simile opportunità esiste ora per i sistemi basati su Fpga.
I componenti di un sistema - processore, dispositivi periferici e altri elementi discreti - sono elementi ben noti. Quando si tratta di trasferirli all'interno di un Asic o di un Fpga, l'assemblaggio a livello di sistema si complica. In primo luogo vi è il problema legato al progetto stesso. Esso di solito viene rappresentato a livello Rtl utilizzando linguaggi come Vhdl o Verilog, efficaci quando si tratta di descrivere il comportamento a livello di componente ma che divengono scomodi e prolissi nel momento in cui la descrizione si estende all'intero sistema. Una tale metodologia di design richiede la simulazione e la verifica a livello di dispositivo come singola entità mediante appositi tool di verifica. La realizzazione e il debug di questi progetti è difficile e richiede parecchio tempo una volta superata una soglia di complessità non particolarmente elevata.
A livello di componenti il progettista si trova a dover affrontare problemi di altro tipo. I tentativi di rendere disponibili standard a livello industriale per i blocchi IP, le periferiche e i core processore di supporto non hanno fornito risultati concreti. In parecchi casi è considerato più sicuro e semplice partire "da zero" piuttosto che effettuare il debug e l'ottimizzazione manuale del codice scritto da qualcun altro per una particolare applicazione. Anche nel mondo degli Fpga, ogni famiglia di dispositivi è caratterizzata da una propria architettura funzionale che deve essere designata "ad hoc". Così, un core di processore che risulta idoneo per una data applicazione, può risultare inadatto per un'altra. La soluzione è trasformare questi componenti soft in entità che si comportano e possono essere manipolate come i loro omologhi hardware. Una volta eseguita tale operazione, essi possono essere utilizzati allo stesso modo dei componenti tradizionali, senza doversi preoccupare della loro struttura interna. Questi componenti soft possono essere rappresentati in maniera simbolica in un sistema di progettazione e assemblati sullo schema circuitale come se si trattassero di elementi hardware tradizionali. In questo schema, i simboli generici vengono collegati agli IP target automaticamente dal sistema di progetto, in modo da rendere disponibile al progettista una soluzione indipendente dal target.
Per quanto riguarda il problema del debugging, è possibile utilizzare un approccio originale che sfrutta la riprogramabilità. In passato, era possibile cercare di simulare il progetto oppure realizzare un prototipo per verificarne le prestazioni. Nessuna di queste opzioni rappresenta la soluzione ottimale. Una volta determinato il dispositivo programmabile più idoneo per un determinato sistema, lo scenario muta radicalmente. Sfruttando le doti di riprogrammabilità di questi dispositivi, è possibile sperimentare senza alcun rischio un numero qualunque di modifiche. È anche possibile compiere un ulteriore passo avanti integrando l'intero flusso di progetto - dallo schema circuitale fino alla programmazione del dispositivo - e rendendolo disponibile sul Pcb, con ogni funzionalità "off chip" richiesta implementata in hardware. Grazie a una scheda di sviluppo per Fpga, è possibile utilizzare strumenti virtuali per esplorare "in diretta" il progetto attraverso porte Jtag ed eseguire il debug "al volo". È anche possibile procedere a un ulteriore ampliamento integrando nel medesimo ambiente la parte relativa allo sviluppo del software embedded. A questo punto è possibile programmare e far girare i processori embedded in tempo reale, molto tempo prima che il prototipo sia disponibile.
Quando si riuniscono tutti questi elementi all'interno di un unico ambiente di progetto si viene a determinare un innovativo approccio che in Altium abbiamo denominato "live design". Penso che questo approccio sia la chiave di volta per consentire la diffusione di questi Fpga "system.capable" (ovvero che integrano a bordo un intero sistema) in una gamma sempre più vasta di applicazioni.