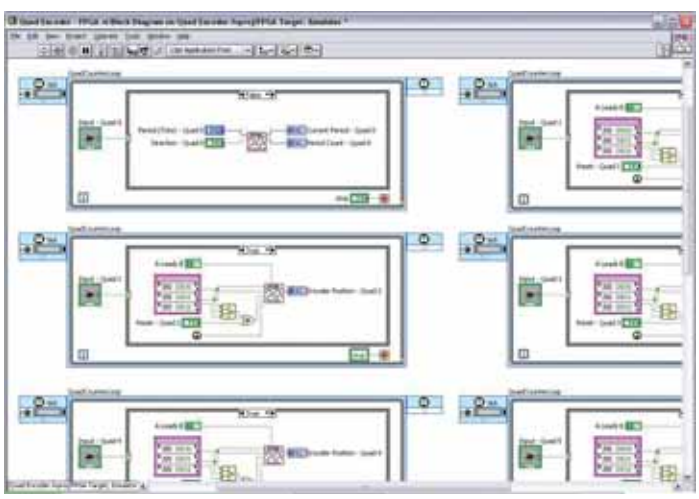
Progettazione grafica di sistemi

Unendo la progettazione basata su modelli a un ambiente di sviluppo visuale, il Graphical System Design permette di ridurre i tempi di sviluppo dei sistemi complessi



Fonte: National Instruments

Massimo Giussani

Livello di astrazione superiore

La complessità dei sistemi elettronici, in particolare dei sistemi dedicati (cosiddetti 'embedded'), cresce ogni giorno di più, grazie alla disponibilità di hardware sempre più potente e alla domanda di funzionalità sempre più avanzate, in grado di soddisfare nuove esigenze o di garantire la differenziazione dalle offerte concorrenti. Per poter contenere i tempi di sviluppo di sistemi complessi che richiedono l'intervento di specialisti in diverse discipline si rende necessario operare a un alto livello di astrazione, in maniera tale che ai progettisti non sia richiesta una formazione specifica nello sviluppo di hardware a basso livello. La progettazione grafica di sistemi, o Graphical System Design (GSD), si prefigge appunto di semplificare le fasi di sviluppo, prototipazione e realizzazione di interi sistemi utilizzando intuitive interfacce grafiche per la loro modellizzazione. Il modello del sistema può essere impiegato per effettuare simulazioni via soft-

La progettazione grafica di sistemi è composta da tre stadi: progettazione, prototipazione e trasferimento nella piattaforma target (Fonte: National Instruments)

Il passaggio dalla linea di comando alle interfacce grafiche ha aperto una nuova era nell'informatica di massa, rendendo accessibili a un più ampio bacino d'utenza i calcolatori e incrementando la produttività degli utenti meno versati alla programmazione. In maniera analoga, con l'introduzione degli strumenti di sviluppo visuale è migliorata la produttività dei progettisti di sistemi complessi, grazie anche alla possibilità data agli esperti della varie discipline coinvolte (fisica, meccanica, elettronica, automazione, robotica) di contribuire direttamente allo sviluppo senza doverne necessariamente conoscere i dettagli implementativi a basso livello. Gli strumenti di progettazione grafica basati sui modelli mettono a disposizione dei team di sviluppo una potente metodologia per creare e ottimizzare, grazie anche alla sperimentazione interattiva di nuove soluzioni, un'ampia varietà di sistemi.

ware con cui il progettista può sperimentare diverse scelte dei parametri progettuali, oppure per generare automaticamente tutto il codice necessario all'effettiva realizzazione hardware, prima di un prototipo che consenta di verificarne l'effettivo funzionamento nel mondo reale e poi del prodotto finale vero e proprio. Il fatto di operare a un livello di astrazione più alto rende più facile riutilizzare le soluzioni progettuali, potendo adattarle a diversi tipi di implementazioni di basso livello (ad esempio generando il codice specifico per diverse architetture). Il Graphical Systems Design è dunque un approccio alla progettazione che, generalmente tramite un unico ambiente di sviluppo visuale, rende accessibili a un più ampio bacino d'utenza tutti gli stadi della progettazione

L'integrazione della piattaforma software per la modellizzazione grafica del sistema e della piattaforma hardware che ne permette la prototipazione semplifica significativamente il percorso progettuale riducendo in ultima analisi il time-to-market (Fonte: National Instruments)

di un sistema complesso, riducendo significativamente il time-to-market e i costi associati.

Tre fasi

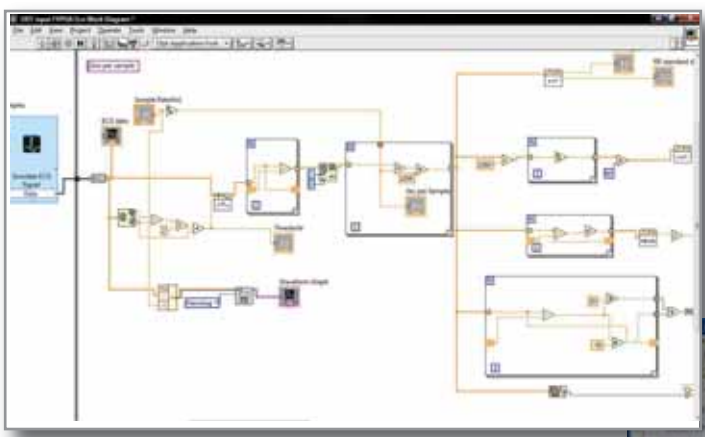
La progettazione grafica di sistemi può essere vista come una successione di tre fasi, o stadi: progettazione, prototipazione e trasferimento nella piattaforma target. La prima fase consiste nel realizzare, attraverso l'interfaccia grafica, un modello software che implementi le funzionalità e gli algoritmi che definiscono il sistema. L'utilizzatore lavora qui a un livello di astrazione superiore utilizzando strumenti del tutto analoghi, se non identici, a quelli che impiegherebbe nella progettazione 'con carta e penna': schemi a blocchi, flussi di dati, formule matematiche. Una volta realizzato il sistema, il software permette di simularne il comportamento dinamico, dando la possibilità al progettista di trovare i migliori compromessi ingegneristici. La fase successiva, quella di prototipazione, consiste nel tradurre in hardware quanto appena realizzato via software: l'ambiente di sviluppo provvede in questo caso a generare tutto il codice necessario alla configurazione delle logiche programmabili, alla programmazione del sistema e alla simulazione dell'interazione con l'ambiente. Lo scopo di questa fase è verificare il funzionamento del sistema nel mondo reale, risolvendo eventuali problemi connessi con la particolare scelta di componenti o le discrepanze tra il modello teorico e la pratica. Una volta che ci si è sincerati che la particolare implementazione hardware si comporta come previsto, si può passare alla fase di trasferimento (deployment) del codice ottimizzato nella piattaforma target. Per poter fare tutto questo un ambiente GSD ha bisogno di due anime: una piattaforma software in grado di interpretare le diverse rappresentazioni di modello del sistema e di tradurle nelle rispettive sequenze di codice (ad esempio codice HDL per la configurazione di Fpga, linguaggi IEC 16631-3 per la programmazione dei PLC, linguaggio macchina eseguibile dai microprocessori), ma anche una piattaforma hardware capace di ospitare tutti i moduli che faranno parte del sistema finale.
L'integrazione di queste due anime permette di ridurre significativamente le iterazioni progettuali necessarie per raggiungere la soluzione cercata.

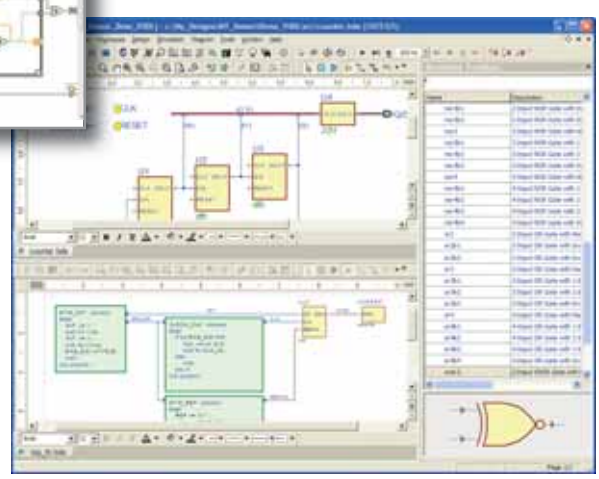
Progettazione per modelli

La descrizione dei diversi elementi e funzioni che costituiscono un sistema complesso richiede la conoscenza di varie forme di rappresentazione grafica o testuale. Ad esempio, il comportamento di un particolare sottosistema può essere convenientemente rappresentato da un blocco funzionale che mette in relazione tra loro opportune variabili di ingresso e uscita associate alle grandezze fisiche che influenzano o sono influenzate dal sistema stesso. Questa descrizione viene solitamente fatta ricorrendo a modelli matematici espressi per mezzo di equazioni differenziali e algebrico-differenziali che permettono, una volta noto lo stato del sistema a un certo istante e la configurazione iniziale delle variabili coinvolte, di determinare l'evoluzione dinamica nel tempo.
Gli ambienti di sviluppo grafico offrono una vasta libreria di blocchi funzionali cui il progettista può attingere per rappresentare gli elementi che compongono il sistema; l'utente non deve fare altro che selezionare la rappresentazione grafica del blocco, trascinarla sull'area di lavoro e configurarne i parametri. Nel caso di sottosistemi particolarmente complessi, il progettista può decidere di specificare direttamente l'algoritmo che lega gli ingressi e le uscite del blocco funzionale utilizzando una descrizione matematica in forma testuale: sostanzialmente specificando le equazioni differenziali che il software risolverà numericamente quando sarà necessario produrre i valori delle variabili di uscita.
Qualora non sia disponibile una descrizione matematica esatta del comportamento del sistema (perché non è nota o perché computazionalmente troppo complessa da trattare) il progettista può ricorrere a modelli empirici basati sull'interpolazione di dati sperimentali o, con un salto netto rispetto all'approccio algoritmico tradizionale, a reti neurali preventivamente 'addestrate'.

Rappresentazioni multiple

La rappresentazione mediante schemi a blocchi permette di stabilire le relazioni tra i vari elementi costituenti il sistema e risulta essere una scelta intuitiva per descrivere un generico sistema fisico (in particolare elettronico e meccatronico) e si presta naturalmente all'implementazione di meccanismi di controllo ad anello aperto o chiuso. In altre circostanze, tuttavia, è preferibile vedere il sistema sotto altri punti di vista. Ad esempio, il modo più naturale per modellizzare le funzioni di controllo è tramite automi agli stati finiti per mezzo di diagrammi stato-transizioni. Un ambiente GSD deve offrire al progettista la possibilità di scegliere il tipo di rappresentazione che più si addice al particolare problema da risolvere. Questo da un lato colloca lo strumento di sviluppo alla portata degli esperti delle varie discipline coinvolte, consentendo loro di utilizzare metodologie di progetto a loro familiari e nascondendo i dettagli implementativi di più basso livello. Dall'altro, la scelta della rappresentazione che meglio si presta a una data metodologia di progettazione contribuisce a semplificare la traduzione delle caratteristiche del sistema nelle istruzioni necessarie alla sua implementazione in hardware. Il principale vantaggio della progettazione basata sulla creazione di un modello del sistema risiede nella possibilità che esso offre di sperimentare nuove soluzioni a costo nullo e, con la potenza degli elaboratori attuali, praticamente in tempo reale. Diventa così più facile tarare le caratteristiche del sistema, ancorché nella sua rappresentazione tramite modello, per soddisfare le esigenze del prodotto finale. Il modello, tuttavia, non è il sistema: esistono numerose ragioni per cui il sistema reale potrebbe comportarsi in maniera diversa da quanto previsto. Le equazioni potrebbero ad esempio riferirsi a un modello semplificato, perché risultato della linearizzazione di un comportamento non lineare oppure perché è stata trascurata l'influenza di alcune variabili, o ancora perché in determinate circostanze (per via della rappresentazione dei numeri in formato binario o per l'intrinseca stabilità numerica degli algoritmi usati) la precisione di calcolo non è sufficiente.





L'ambiente di sviluppo grafico semplifica la creazione di un modello funzionale, permettendo ai progettisti di concentrarsi sulle funzionalità avanzate a livello di sistema e sull'esplorazione delle diverse opzioni di ottimizzazione (Fonti: NI, Aldel)

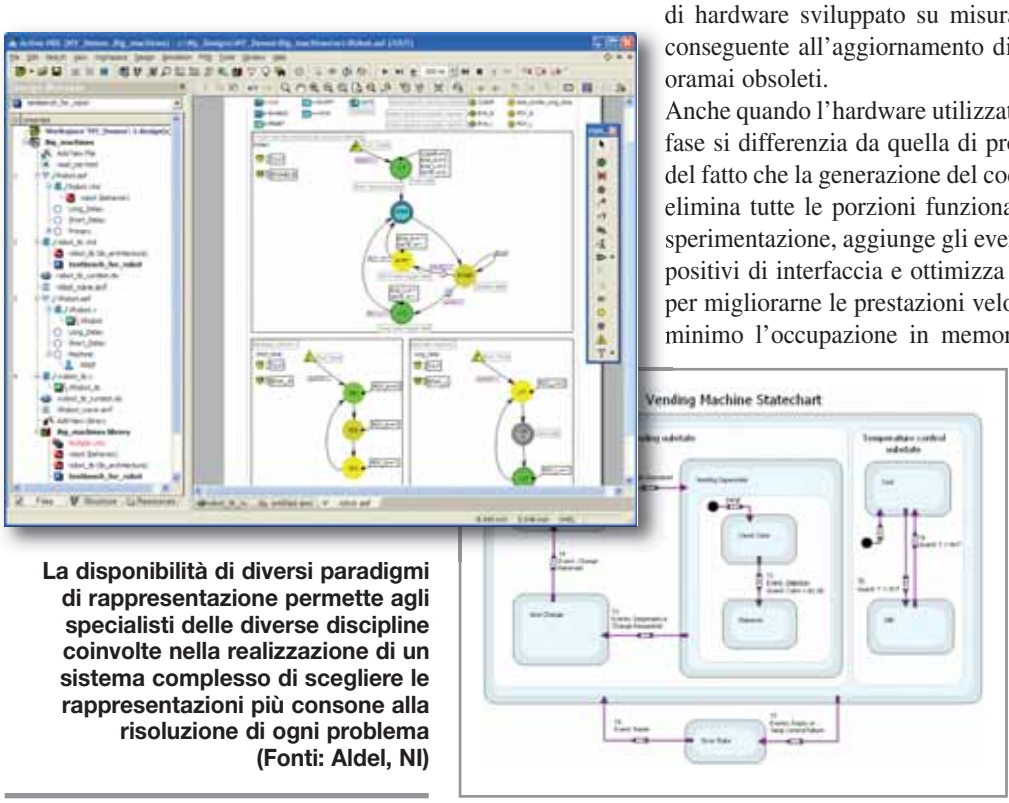
Prototipazione

Si impone pertanto un'ulteriore fase dello sviluppo che ha lo scopo di proiettare il progetto nel mondo reale per mezzo di una piattaforma hardware, compatibile con il software grafico di modellizzazione, che integri i componenti fisici che verranno utilizzati dal prodotto finale. Tra i componenti utilizzati dai sistemi dedicati figurano, ad esempio, microprocessori e DSP, logiche programmabili per la creazione di circuiteria ad hoc, la memoria necessaria per ospitare il codice che regge il funzionamento del sistema e interfacce di I/O in grado di adattarsi alle diverse possibilità di interfacciamento con il mondo esterno. Il prototipo richiede la conversione effettiva della descrizione del modello basata sulla rappresentazione grafica in istruzioni in grado di modificare o interagire con l'hardware del prototipo: codice HDL per configurare la logica Fpga, istruzioni nei linguaggi dello standard IEC 61131-3 per i PLC, programmi in C o linguaggio macchina che verranno eseguiti dai nuclei di elaborazione. In questa fase gli algoritmi che sono stati implementati ipotizzando la continuità delle variabili vengono convertiti e ottimizzati per essere eseguiti su un sistema digitale, quindi con grandezze discrete, rappresentate da valori con un numero limitato di cifre significative se non addirittura in formato a virgola fissa. La piattaforma GSD provvede a generare automaticamente tutto il codice necessario alla messa in funzione del prototipo con il duplice vantaggio di ridurre la probabilità di introdurre errori in questa fase dello sviluppo e di far risparmiare tempo ai progettisti.

Con i Cots è meglio

Oltre al codice necessario all'implementazione delle funzionalità previste dallo sviluppatore, una buona piattaforma GSD di deve preoccupare di generare anche il codice

necessario alla simulazione degli stimoli cui il sistema sotto studio è soggetto nel mondo reale: se si sta progettando un ricevitore sarà necessario simulare i segnali prodotti dal corrispondente trasmettitore e viceversa. A seconda delle funzionalità da testare, potrà bastare generare i segnali direttamente in corrispondenza delle interfacce con il mondo esterno, oppure potrebbe rendersi necessario inserire il prototipo completo dei moduli di I/O nell'ambiente nel quale è destinato a operare. Se vengono riscontrati errori o si ravvisa la possibilità di migliorare il prodotto, i progettisti possono tornare allo stadio precedente, modificare il modello e ripetere la fase di prototipazione in breve tempo e senza dover produrre fisicamente un nuovo prototipo. I tempi e i costi di progettazione possono essere ulteriormente ridotti prendendo in considerazione a questo stadio dello sviluppo il ricorso a componenti Cots (Components Off The Shelf). Se l'ambiente GSD supporta una piattaforma di prototipazione modulare basata su componenti 'da scaffale' si può saltare la fase di creazione del prototipo ex-novo, con conseguente risparmio di tempo e denaro.



La disponibilità di diversi paradigmi di rappresentazione permette agli specialisti delle diverse discipline coinvolte nella realizzazione di un sistema complesso di scegliere le rappresentazioni più consone alla risoluzione di ogni problema (Fonti: Aldel, NI)

Il ricorso ai moduli Cots semplifica il riutilizzo della proprietà intellettuale, ma anche dello stesso hardware, per altri progetti. Consente inoltre un approccio flessibile alla produzione: per quantitativi ridotti può risultare conveniente produrre il prodotto finale assemblando componenti commerciali, e solo quando l'economia di scala lo giustifica passare alla più costosa realizzazione di hardware su misura. Non ultimo, permette di creare un prototipo funzionante del sistema, pur se non fedele in tutto e per tutto al prodotto finale, prima ancora che siano disponibili i campioni dell'hardware. L'accorciamento dei tempi di sviluppo che ne consegue assume una valenza di cruciale importanza quando si lavora a prodotti con hardware allo stato dell'arte e si voglia arrivare sul mercato prima della concorrenza. Ritardare la fase di affinamento del software al momento in cui l'hardware sarà finalmente disponibile potrebbe rivelarsi fatale per il successo del proprio prodotto.

La fase finale

L'ultima fase è quella di trasferimento del codice generato dal sistema di progettazione grafica nel target vero e proprio, che si tratti di un assemblato di componenti Cots, di hardware sviluppato su misura o di una variante conseguente all'aggiornamento di alcuni componenti oramai obsoleti.

Anche quando l'hardware utilizzato è lo stesso, questa fase si differenzia da quella di prototipazione per via del fatto che la generazione del codice 'di produzione' elimina tutte le porzioni funzionali all'operazione di sperimentazione, aggiunge gli eventuali driver dei dispositivi di interfaccia e ottimizza il codice eseguibile per migliorarne le prestazioni velocistiche e ridurre al minimo l'occupazione in memoria. Se richiesto, in questa fase il codice può essere modificato per conformarsi allo stile utilizzato dall'azienda che ha commissionato il prodotto.

Nel complesso il Graphical System Design rappresenta un approccio alla progettazione di sistemi che sfrutta la programmazione grafica per innalzare il livello di astrazione cui operano gli sviluppatori. La diffusione di questi sistemi è una diretta conseguenza della richiesta di sistemi complessi che scaturisce dalla riduzione dei costi della logica digitale e dei moderni microprocessori. Per affrontare la crescente complessità nello sviluppo di questi sistemi, sempre più spesso dotati di architetture multiprocessore cui è richiesto di eseguire numerosi compiti in parallelo con un elevato grado di sincronizzazione, si è reso necessario alleggerire il carico sulle spalle dei progettisti delegando la generazione di codice alla piattaforma di sviluppo.

Il ricorso al paradigma grafico ha permesso ai progettisti di concentrarsi sulle complessità a livello di sistema, mettendoli in condizione di affrontare questioni, come la gestione del parallelismo e della sincronizzazione, che diventano presto problematiche quando sono affrontate con le metodologie tradizionali. ■