



Metodologie di progetto hardware

All. Informatici e telecomunicazioni (A-ZZZ)

Fabrizio Ferrandi

a.a. 2009-2010



Design challenges

- I sistemi elettronici stanno diventando sempre più grandi mentre i tempi di sviluppo si riducono
 - 20 Mgate dimensione comune di un'attuale ASIC
 - 0.4 Mlinee di codice C per descrivere il comportamento
 - 5 Mlinee di codice RT VHDL/Verilog
 - Gruppi di progetto sempre più numerosi
 - diverse centinaia di persone
 - diverse capacità progettuali
 - lavoro parallelo a diversi livelli d'astrazione
 - la gestione della complessità e delle comunicazioni è veramente complicata
 - Gli strumenti anche se sempre più complessi sono ancora inadeguati
 - un tipico progettista deve utilizzare circa 50 strumenti per ciascun componente
 - Gli strumenti spesso hanno banchi interfacce non sempre semplici da usare ecc.
-



Design challenges

- Le decisioni relative allo spazio di progetto sono molto complicate
 - compromesso tra prestazioni costo e time-to-market
 - decisioni devono essere prese 2-3 anni prima della fine del progetto
 - scelte di progetto difficili senza arrivare alla realizzazione del dispositivo
 - scheduling del ciclo di vita del prodotto
 - Verifica funzionale
 - la simulazione è ancora il mezzo principale per la verifica funzionale ma inadeguata per le dimensioni dello spazio di progetto
 - un errore nel progetto hardware è veramente costoso da recuperare (diverso dal software ;-)
-

- 3 -



Design challenges

- Tradeoff tra i diversi livelli di astrazione nella modellizzazione:
 - dettaglio del modello vs. dimensione del modello da mantenere
 - modelli ad alto livello possono essere mantenuti da una due persone
 - modelli dettagliati richiedono progetti partizionati in più blocchi aumentando così il costo di comunicazione
 - accuratezza del modello vs. compattezza
 - modelli compatti omettendo dettagli danno solo una stima approssimata dell'implementazione
 - modelli dettagliati sono di dimensioni maggiori e richiedono grossi cambiamenti a seguito di modifiche delle scelte progettuali
 - velocità di simulazione vs. prestazioni hardware
 - modelli ad alto livello possono essere velocemente simulati ma non possono essere implementati efficientemente con mezzi automatici
 - modelli a basso livello possono essere velocemente realizzati ma non possono essere simulati altrettanto efficientemente
-

- 4 -



Approccio di progetto generale

- Come costruiscono i ponti, gli ingegneri ?
 - Divide et impera !!!!
 - partizionare lo spazio di progetto in diversi sotto problemi di dimensione ragionevole
 - definire un modello matematico per il sotto problema e trovare una soluzione algoritmica
 - Controllare le limitazioni del modello!!!!!!
 - realizzare gli algoritmi in singoli strumenti CAD definendo interfacce generali per lo scambio dati
 - realizzare strumenti in grado di verificare le condizioni al contorno
 - integrare gli strumenti realizzati in flussi di progetto
 - identificare cosa manca e ripartire
-

- 5 -



Automazione di progetto

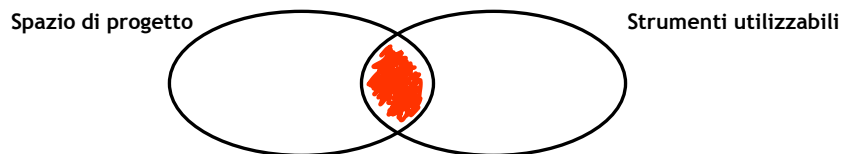
- L'automazione di progetto è una delle aree più avanzate dell'informatica
 - molti problemi richiedono sofisticati modelli matematici
 - molti algoritmi sono computazionalmente complessi e richiedono euristiche avanzate per poter risolvere problemi di dimensioni reali
 - le condizioni al contorno devono essere ben definite anche tra strumenti differenti
 - In questa area di ricerca spesso accade che
 - Il problema è alla ricerca di una soluzione:
 - La dimensione del problema è elevata, il modello è complesso o gli algoritmi non scalano
 - La dimensione del problema è ridotta, ma le soluzioni non sono sufficientemente buone
 - La soluzione è alla ricerca del problema:
 - Il modello è troppo semplificato perché il problema è troppo complesso con diverse condizioni al contorno
-

- 6 -



La chiave per arrivare al successo

- Una buona integrazione fra metodologie di progetto e strumenti comporta indirizzare la complessità algoritmica richiedendo:
 - partizionamento manuale del problema
 - guidare gli strumenti sfruttando l'esperienza dei progettisti
 - iterare nella ricerca delle soluzioni per arrivare all'ottimo
- tutto ciò rende i sistemi CAD e i flussi di progetto molto complessi e difficili da gestire



- 7 -



Esempi di divide et impera

- La simulazione RTL verifica funzionalmente il progetto, assume che le temporizzazioni siano corrette
 - combinando verifica statica delle temporizzazioni, verifica formale di equivalenza e simulazione funzionale si separano i problemi
 - la simulazione funzionale cycle-based riduce gli eventi da gestire aumentando così l'efficienza nella simulazione
- Tuttavia:
 - l'analisi timing è conservativa rispetto alle massime prestazioni raggiungibili dal sistema

- 8 -



Esempi di divide et impera

- L'analisi statica delle temporizzazioni utilizza modelli semplici per le porte logiche
 - la complessità dell'analisi diventa lineare (identificazione del percorso più lungo/corto nel circuito)
 - l'analisi incrementale diventa molto efficiente ed utilissima nella fasi di sintesi logica dipendente dalla tecnologia
 - Tuttavia:
 - il ritardo reale varia molto nella realtà
 - il modello spesso assume un fan-out medio invece del carico reale delle porte logiche
 - il modello dei ritardi assume segnali ideali
 - l'effetto di "skew" è trascurato
-

- 9 -



Esempi di divide et impera

- La sintesi logica assume porte logiche ideali indipendenti dall'implementazione fisica
 - la tecnologia standard cell ha reso la sintesi logica possibile
 - le porte logiche sono sovra-specificate per una vasta gamma di combinazioni (diverso numero di fan-out, diversi carichi per le interconnessioni)
 - le standard cell sono organizzate in righe e colonne per minimizzare effetti di latch-up, il consumo di potenza e la distribuzione del clock
 - Tuttavia:
 - l'implementazione del layout rimane sub-ottima perché le celle sono progettate per il caso pessimo e con un elevato margine rispetto alle condizioni al contorno
-

- 10 -



Esempi di divide et impera

- La sintesi logica usa modelli per stimare l'area molto grezzi
 - il numero dei letterali o delle tabelle di look-up nel dimensionamento delle porte consente veloci confronti fra differenti realizzazioni
- Tuttavia:
 - La dimensione reale delle porte logiche può variare fortemente al variare del carico dei requisiti di temporizzazione
 - area richiesta dalle interconnessioni è completamente trascurata

- 11 -



Esempi di dividi et impera

- La verifica formale di equivalenza assume la stessa codifica degli stati nel confronto di due progetti
 - riduce il problema generale della verifica di equivalenza al quello combinatorio che è computazionalmente molto meno complesso
 - sfruttando le similarità strutturali tra due progetti da confrontare gli strumenti riescono a confrontare progetti molto grandi (multi-million gate)
 - gli algoritmi per l'identificazione della corrispondenza fra registri consentono un'estensione della metodologia
- Tuttavia:
 - la verifica di circuiti combinatori non può essere sempre applicata al caso sequenziale

- 12 -



Struttura del corso

- Introduzione
 - Flusso di progetto hardware.
 - I livelli d'astrazione
 - La sintesi dei sistemi digitali
 - Ottimizzazione delle prestazioni
 - Low-power design
 - Sintesi ad alto livello
 - Il collaudo
 - Modelli
 - Algoritmi per il collaudo
 - Design for Testability.
 - La verifica dei sistemi digitali
 - Simulazione
 - Equivalence checking
-

- 13 -



Materiale didattico

- Materiale preparato dai docenti
 - Home page Ferrandi: www.dei.polimi.it/people/ferrandi
 - Home page Grassi: www.dei.polimi.it/people/grassi
 - Testi consigliati
 - G. De Micheli: "Synthesis and Optimization of Digital Circuits", Mc Graw-Hill International Editions, 1994.
 - M. Abramovici, M.A. Breuer, A.D. Friedman: "Digital Systems Testing and Testable Design", Computer Science Press, 1993.
 - G.D Hatchel, F. Somenzi, "Logic synthesis and verification algorithms", Kluwer Academic Publishers, 1996.
 - Docenti e collaboratori
 - Fabrizio Ferrandi - Dipartimento di Elettronica e Informazione, I piano, tel. 02 2399 3479, e-mail ferrandi@elet.polimi.it, ricevimento: giovedì 14.15-16.15
 - Paolo Roberto Grassi - Dipartimento di Elettronica e Informazione, tel. 02 2399 3494, e-mail grassi@elet.polimi.it
-

- 14 -



Modalità di svolgimento delle prove di verifica

- Prerequisito: nessuno

- Valutazione: la prova d'esame assegna 32 punti corrispondenti al 30 e Lode
 - Prova (inizio maggio???)
 - Esami (luglio - settembre - febbraio)
 - E' possibile recuperare le prove nelle sessioni disponibili
 - La prova scritta potrà essere seguita da una prova orale integrativa
 - La prova orale dovrà essere tenuta nella stessa sessione della prova scritta
 - **Obbligo iscrizione prove ed esami**