

Introduzione al codesign di sistemi dedicati

Lezione 01 di Sistemi dedicati

Docente: Giuseppe Scollo

Università di Catania
Dipartimento di Matematica e Informatica
Corso di Laurea Magistrale in Informatica, AA 2019-20

Indice

1. Introduzione al codesign di sistemi dedicati
2. Obiettivi dell'introduzione
3. Cosa è il codesign HW/SW?
4. Perché il codesign HW/SW? Hardware flessibile ...
5. ... sistemi dedicati, sistemi embedded
6. Fattori tecnologici rilevanti del codesign HW/SW
7. Fattori economici rilevanti del codesign HW/SW
8. Spazio di progetto di architetture dedicate
9. Livelli di astrazione di modelli per il codesign
10. Linguaggi di descrizione dell'hardware
11. Un piccolo esempio in GEZEL
12. Progetto di System-on-Chip (SoC)
13. Interfacce HW/SW
14. Piattaforme per il codesign
15. Cosimulazione di sistemi HW/SW
16. Letture consigliate
17. Siti web di interesse

Motivazione e concetti fondamentali di:

- modellazione, progettazione e realizzazione ottimale di sistemi di elaborazione dedicati a una specifica applicazione
- uso di strumenti hardware e software, quali piattaforme di sviluppo e di cosimulazione, per progettare e realizzare sistemi dedicati a una specifica applicazione

Cosa è il codesign HW/SW?

Una definizione "tradizionale":

the design of cooperating hardware components and software components in a single design effort
(Schaumont, p. 11)

Una definizione "non tradizionale":

the partitioning and design of an application in terms of fixed and flexible components
(Schaumont, p. 12)

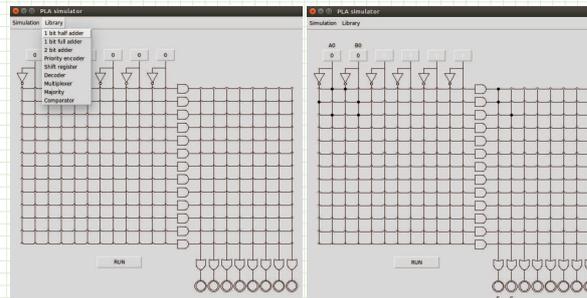
Qual è la differenza?

- progetto di un'applicazione piuttosto che di componenti
- il **partizionamento** dell'applicazione è un'attività del progetto
- **componenti**: hardware → **fissi**, software → **flessibili**

Perché questa differenza? V. appresso...

Hardware flessibile, in varia misura:

- caso prototipico: PLA - v. il simulatore di PLA di Alice Plebe:



con Library estesa e supporto di Python 3 nella versione 2.0 da Matteo Cavallaro

- odierno caso pratico: FPGA
il "programma" è una netlist di elementi logici e connessioni fra essi, specificata dall'utente
- hardware soft: un *soft-core* è un processore realizzato nella netlist di una FPGA

... sistemi dedicati, sistemi embedded

I due termini non sono sinonimi:

- Un sistema *dedicato* è progettato, in tutti i suoi aspetti, per realizzare un'applicazione specifica
- Un sistema *embedded* è un sistema dedicato che realizza un'applicazione specifica nel contesto di, e interagendo con, un più ampio sistema fisico
i due assieme formano un *sistema ciber-fisico*

Sistemi embedded: grande varietà, mercati in rapida crescita:

autoveicoli, aeromobili, controllo del traffico, telefonia mobile, fotografia digitale, televisione, domotica, robotica ...

Sistemi dedicati sono anche componenti o sottosistemi di sistemi di elaborazione dell'informazione *general-purpose*:

coprocessori aritmetici, coprocessori crittografici, schede videografiche, codec A/V, controllori di I/O DMA, sottosistemi GPU ...

Fattori tecnologici giocano a favore di più hardware:

- **Prestazione computazionale**
lavoro prodotto nell'unità di tempo, o per ciclo di clock: il parallelismo dell'hardware o acceleratori hardware dedicati migliorano la prestazione computazionale
- **Efficienza energetica**
può variare per parecchi ordini di grandezza, per esempio (Schaumont, p. 14)

Energy efficiency of AES encryption implementations					
Gb/J:	10^{-6}	10^{-3}	10^{-2}	10^0	10^1
platform:	Java KVM Sparc	C Sparc	Asm Pentium-III	Virtex-II FPGA	0.18µm CMOS ASIC

- **Dissipazione di potenza**
l'aumento di prestazione computazionale da più alta frequenza di clock è limitato dall'aumento proporzionale della potenza dissipata, dunque da limiti e costi della tecnologia di raffreddamento → *architetture parallele*

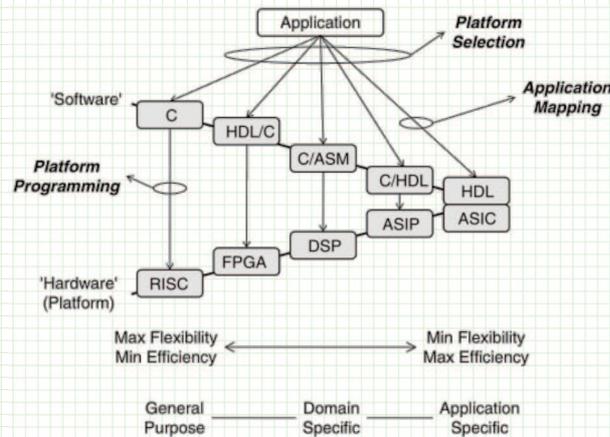
Supporti ottimali al codesign HW/SW: piattaforme di calcolo parallelo

- multiprocessori a memoria condivisa, acceleratori su FPGA, GPU, architetture multi-core ...
un esempio di piattaforma open HW/SW ad alta efficienza energetica: la scheda di sviluppo Parallela

Fattori economici giocano a favore di più software:

- **Costo del progetto**
Il progetto di un chip ha costi fissi molto alti;
chip riprogrammabili, che permettono il riuso mediante riprogrammazione, spalmano il costo di progetto del chip su più prodotti o versioni di un prodotto;
tuttavia, la riprogrammabilità può realizzarsi in molte forme diverse
- **Tempo di sviluppo**
Non solo il costo di progetto ma anche il tempo di sviluppo di un nuovo chip è considerevole;
uno sviluppo rapido, invece, permette un ingresso tempestivo nel mercato,
questo dà ricavi più alti, il che è specialmente significativo per prodotti innovativi
- **Complessità del progetto**
hardware fisso significa decisioni di progetto fissate;
la flessibilità del software permette ai progettisti di:
 - sviluppare l'applicazione a un più alto livello di astrazione, e
 - mantenere l'applicazione attraverso le modifiche necessarie per correggere errori o per far fronte all'evoluzione di requisiti

La collezione strutturata di tutte le possibili realizzazioni di una data applicazione



Schaumont, Fig. 1.7 - The hardware-software codesign space

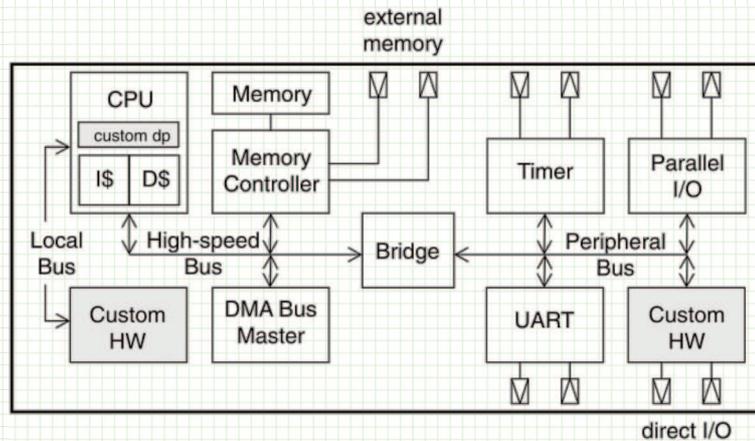
Livelli di astrazione di modelli per il codesign

Definibili dalla *granularità temporale* delle azioni elementari (atomiche)

A partire dal livello più basso di astrazione:

- segnali continui
 - modelli: sistemi di equazioni differenziali; utili per sistemi ibridi con componenti analogici*
 - non di uso pratico per tipici sistemi HW/SW*
- eventi discreti
 - variazioni di segnali a istanti di tempo irregolarmente distribuiti: astrazione minima per hardware digitale*
- cicli di clock
 - eventi discreti osservati a intervalli di tempo regolari*
 - modelli RTL (register-transfer level), utili per hardware sincro a singolo clock*
- istruzioni macchina
 - livello utile per la simulazione di sistemi software complessi, per i quali la simulazione a livello di cicli di clock avrebbe un costo proibitivo; tuttavia non rivela la reale prestazione temporale del sistema*
- transazioni
 - modelli in termini di interazioni fra componenti del sistema; utile quando anche la simulazione a livello di istruzioni sia troppo costosa, come pure nelle prime fasi del progetto di un sistema*

Uno schema generico di progetto:



Schaumont, Fig. B.1 - Generic template for a system-on-chip

Interfacce HW/SW

Concetti di base:

> Sincronizzazione

granularità temporale: ciclo di clock, ciclo di bus, transazione
scambio di dati: astratto, scalare, composto
controllo: semafori, protocolli di handshake, bloccante o non bloccante ecc.

> Prestazione computazionale

analisi dei colli di bottiglia, per esempio:
 - canale a v bit/trasferimento, B cicli/trasferimento
 - coprocessore a w bit/esecuzione, H cicli/esecuzione
limitata dalla capacità di comunicazione: $v/B < w/H$
limitata dalla velocità di calcolo: $v/B > w/H$

> Accoppiamento HW/SW

accoppiamento stretto: interazione frequente, granularità fine
accoppiamento lasco: interazione infrequente, granularità grossa

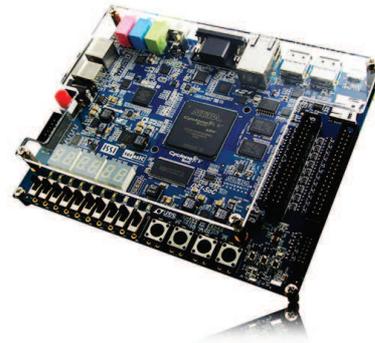
Piattaforme per il codesign

Collezioni di strumenti HW e SW per lo sviluppo e il collaudo in codesign

le schede di sviluppo di FPGA sono gli strumenti hardware di base a tal fine

sono accompagnate da sofisticati sistemi software per il codesign ad alto livello e per la cosimulazione

per esempio, la scheda di sviluppo Intel DE1-SoC (v. immagine), che monta un chip con FPGA Cyclone V, ha nello stesso chip un processore ARM Cortex-A9, può includere due processori softcore NIOS II sulla FPGA ed è supportata dal software Quartus Prime Lite, liberamente disponibile



Scheda di sviluppo Intel DE1-SoC con FPGA Cyclone V
fonte: Intel® FPGA University Program

Piattaforme di sviluppo open-hardware includono: Parallella, Arduino, Cosino ...

v. siti web d'interesse

Cosimulazione di sistemi HW/SW

La cosimulazione può anche essere condotta su piattaforma software, senza impiego di FPGA

una tale piattaforma tipicamente consta di:

- cross-compilatori e cross-assemblatori per un dato insieme di linguaggi di programmazione e di famiglie di processori, per la parte SW di un modello di codesign
- simulatori di HDL, per la parte custom del modello
- simulatori di insiemi di istruzioni di microprocessori, accurati al livello dei cicli di clock
- modelli software di interfacce hardware di microprocessori
- e possibilmente altro ancora ...

la cosimulazione al livello dei cicli di clock mette i progettisti in grado di stimare le prestazioni di soluzioni di codesign ben prima della loro realizzazione effettiva

un esempio di tale piattaforma è stato il software Gezel: una collezione di pacchetti Debian per l'installazione su Ubuntu, aggiornata a ogni nuova LTS fino alla 16.04, ma non più supportata

N.B. faremo uso di un componente di questo software in alcune esperienze di laboratorio, per la traduzione automatica di modelli Gezel in VHDL; istruzioni per l'installazione locale e documentazione pertinente sono reperibili nell'area riservata.

Lecture consigliate

Testi di riferimento

Schaumont, Ch. 1, Sect. 1.1.4 – 1.4, 1.6

Testi per consultazione

Vahid & Givargis, Ch. 1, Sect. 1.1 – 1.4

Brandolese, Fornaciari, Cap. 1

Siti web d'interesse

Corsi di codesign

Hardware/Software Codesign, Patrick Schaumont, VirginiaTech
rijndael.ece.vt.edu/ece4530f19

Hardware/Software Codesign with FPGAs, Jim Plusquellic, U. of New Mexico
ece-research.unm.edu/jimp/codesign

Cyber-physical system fundamentals, J.-J. Chen and A. Toma, TU Dortmund
ls12-www.cs.tu-dortmund.de/daes/en/lehre/courses/sommersemester-2019/cyber-physical-system-fundamentals-ss-2019.html

Introduction to Embedded Systems, Edward A. Lee and Sanjit A. Seshia, U. of Berkeley
bcourses.berkeley.edu/courses/1454183

Codesign platforms and tools

Intel® FPGA University Program - Educational Materials:
www.intel.com/content/www/us/en/programmable/support/training/university/materials.html

Xilinx University Program: www.xilinx.com/support/university.html

CUDA: developer.nvidia.com/cuda-zone

Parallella: www.parallella.org

Arduino: www.arduino.cc

Cosino: www.cosino.it