Linguaggi di descrizione dell'hardware: Gezel, VHDL, Verilog, SystemC

Esercitazione 03 di Sistemi dedicati

Docente: Giuseppe Scollo

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

Indice

argomenti dell'esercitazione

in questa esercitazione si trattano:

- concetti di base di Gezel
- primo approccio alla descrizione di hardware sincrono in VHDL
- cenni di Verilog e SystemC
- esperienza di laboratorio

Gezel: concetti di base

essenzialmente, due sorte di variabili: segnali e registri

- di uno o più bit, di tipo `ns` o `tc` ecc.
- i registri sono inizializzati a zero, i segnali non hanno memoria
- ingressi e uscite di un datapath sono sempre segnali
- l'uscita di un registro acquisisce il valore in ingresso solo sul fronte attivo del clock, dunque la semantica dell'assegnamento a registro (a intervalli di tempo discreti) è diversa da quella dell'assegnamento a segnale (istantanea)

l'ordine degli assegnamenti è irrilevante

> l'hardware è inerentemente parallelo

le espressioni a destra degli assegnamenti sono costruite da operatori che hanno una diretta interpretazione quali componenti hardware, v. Schaumont tabella 5.1 e sez. 5.1.3

incapsulamento in datapath e gerarchia strutturale, con riuso e clonazione di moduli, v. Schaumont sez. 5.2

introduzione rapida a VHDL

livello di astrazione: eventi discreti; semantica: coda degli eventi futuri

unità di specifica:

- *entità*: specifica di interfaccia (porte: nomi, tipi e direzioni dei segnali di I/O)
  può avere parametri: dichiarazioni *generic*
- *architettura*: specifica dell'interno dell'entità

più architetture associabili a una data entità, secondo stili di specifica differenti:

- *funzionale*: in termini di *processi* sequenziali, attivati da variazioni di certi segnali (*sensitivity list*)
- *comportamentale*: in termini di assegnamenti *concorrenti* (relazioni fra segnali)
- *strutturale*: in termini di *componenti* e connessioni delle porte di loro *istanze*

incapsulamento e gerarchia:

- componenti: incapsulamento simile a quello di entità, gerarchia strutturale
- processi: incapsulano funzionalità, ma non ne è consentita la nidificazione

v. Zwoliński Ch. 3-4 per una prima introduzione alla sintassi di costrutti VHDL e semplici esempi

cenni di Verilog e SystemC

livello di astrazione: eventi discreti (come VHDL)

Verilog, alcune caratteristiche:

- simile a VHDL per la descrizione di schemi di entità connesse
- più caratteristiche per la descrizione di basso livello (transistori)
- meno flessibile per la specifica top-down di sistemi
  SystemVerilog progettato a tal fine, ma meno supportato da strumenti di sintesi automatica
- v. Zwoliński, App. B per un'introduzione alla sintassi di Verilog e semplici esempi

SystemC, alcune caratteristiche:

- libreria di classi C++ con funzioni richieste per la descrizione di hardware
- processi concorrenti controllati da sensitivity list (v. VHDL)
- architettura stratificata del linguaggio, con canali di comunicazione, di modelli di computazione e canali per specifiche metodologie
- v. Marwedel sez. 2.7 per una introduzione più articolata a SystemC

esperienza di laboratorio

la traduzione in VHDL prodotta dal generatore di codice Gezel talvolta dà qualche sorpresa...

- creare il file sorgente delay_collatz.fdl contenente la descrizione Gezel della seconda versione dell'esempio Gezel presentato nella seconda lezione (il file è reperibile nell'area dedicata di laboratorio, cartella 3x+1)
- eseguire da linea di comando: fdlvhd delay_collatz.fdl
- lanciare Quartus e in tale sistema creare un nuovo progetto di nome delay_collatz
- copiare i file .vhd prodotti al passo 2 nella cartella del progetto e visualizzare il file delay_collatz.vhd nell'editor di Quartus
- assegnare i file suddetti al progetto, compilare e visualizzare i messaggi di errore che ne risultano, quindi, con doppio click sul primo dei messaggi di errore, identificare la linea di codice sorgente VHDL che causa l'errore
- analizzare l'errore in delay_collatz.vhd alla luce di Zwoliński, Sect. 4.1.2 e p. 63 di 4.2.3
- escogitare una modifica del sorgente Gezel che elimini l'errore nella sua traduzione VHDL, quindi produrne la revisione delay_collatz_rev.fdl e la sua traduzione delay_collatz_rev.vhd, e assegnare quest'ultima quale *top-entity* a una revisione del progetto
- ricompilare e verificare che non vi siano messaggi di errore
- impostare il clock a una frequenza che assicuri un valore positivo per lo slack di caso peggiore (v. note sulla regolazione del clock dalla prima esperienza di laboratorio)
- confrontare l'uso di risorse (n. di LE e registri) e slack di caso peggiore con quelli ottenuti dalla compilazione e analisi temporale condotte nella seconda esperienza di laboratorio
- copiare e modificare le forme d'onda di test dalla seconda esperienza di laboratorio (se necessario, regolare il clock in accordo con la frequenza determinata sopra)
- lanciare l'esecuzione della simulazione funzionale e verificare la corrispondenza del risultato alle attese

riferimenti

letture raccomandate:

Schaumont, Ch. 5, Sect. 5.1-5.2

Zwoliński, Ch. 3, Sect. 3.1-3-7; Ch. 4, Sect. 4.1-4.3

altre fonti per consultazione:

Brandolese, Fornaciari, App. B

Zwoliński, App. B

Marwedel, Ch. 2, Sect. 2.7

Vahid, Digital Design, Cap. 9, Hardware Description Languages (PDF slides)

Smith, VHDL & Verilog Compared & Contrasted