Introduzione all'uso combinato di Gezel con un simulatore VHDL

Esercitazione 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

in questa esercitazione si trattano:

- modelli di hardware in Gezel: caratteristiche, limiti, usi pratici
- installazione del software Gezel
- traduzione di modelli Gezel in modelli VHDL
- installazione del software Quartus
- compilazione, analisi e tuning di modelli VHDL in Quartus
- creazione di forme d'onda di test in Quartus
- simulazione funzionale con il ModelSim di Quartus

circuiti digitali sincroni a singolo clock, composti da interconnessione di:

- reti combinatorie
- flip-flop

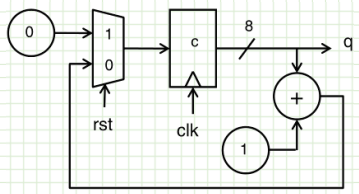
dunque anche componenti di largo uso quali: registri, addizionatori, moltiplicatori ecc.

livello di astrazione: cicli di clock, modelli RTL

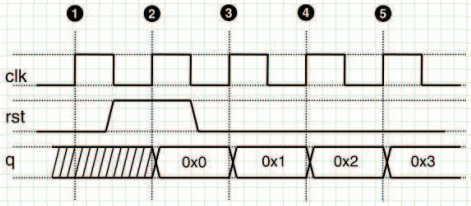
non si possono modellare: circuiti asincroni, con latch, con clock multifase ecc.

i modelli RTL sono comunque sufficienti in gran parte di casi pratici, per descrivere realizzazioni hardware di algoritmi

esempio:

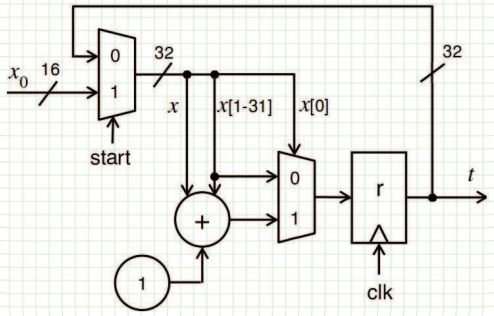



```
dp count(in  rst : ns(1);
         out q   : ns(8)) {
  reg c : ns(8);
  always {
    q = c;
    c = rst ? 0 : c + 1;
  }
}
```



Schaumont, Fig. 1.1 - Modelli e comportamento di un componente hardware

riconsideriamo l'esempio visto nella prima lezione:




```
dp collatz ( in start : ns(1) ; in x0 : ns(16) ;
             out t : ns(32)) {
  reg r : ns(32) ;
  sig x : ns(32) ;
  always {
      t = r ;
      x = start ? x0 : r ;
      r = x[0] ? x + (x >> 1) + 1 : x >> 1 ;
  }
}
```

che uso possiamo farne?

e.g., come possiamo visualizzare il suo comportamento?

gli strumenti di sviluppo industriali richiedono descrizioni in linguaggi standard, quali VHDL o Verilog...

il generatore di codice della piattaforma Gezel produce una traduzione in VHDL *sintetizzabile*

esperienza di laboratorio:

1. installare il software di base Gezel e il suo generatore di codice VHDL
2. reperire il file sorgente collatz.fdl contenente la descrizione Gezel dell'esempio
3. eseguire la traduzione da linea di comando: fdlvhd collatz.fdl
4. installare Quartus Prime Lite 16.1 prodotto da Intel Corp. e lanciarlo, quindi in tale sistema:
5. creare un nuovo progetto di nome collatz
6. copiare i file .vhd prodotti al passo 3 nella cartella del progetto
7. assegnare i file suddetti al progetto e compilare
8. controllare eventuali messaggi di errore o warning
9. impostare il clock a una frequenza che assicuri un valore positivo per lo slack di caso peggiore
10. creare forme d'onda di test per il circuito, con un input di clock corrispondente alla frequenza del passo precedente e il valore 27 per l'inizio della traiettoria
11. lanciare l'esecuzione della simulazione funzionale
12. ripetere la simulazione per differenti valori di inizio della traiettoria

alcune note per l'esecuzione dell'esperienza su Ubuntu 16.04:

le note che seguono, in inglese, indicano alcuni accorgimenti pratici per superare piccoli problemi che altrimenti possono presentarsi nell'esecuzione dell'esperienza di laboratorio

- si può tentare l'installazione su una versione più recente della distribuzione Ubuntu: questo richiede la ricompilazione dei sorgenti Gezel, ma non è stato collaudato; qualora il tentativo fallisse si possono reperire nell'area riservata di laboratorio i file VHDL prodotti dalla traduzione ed eseguire l'esperienza a partire dal punto 4
- su un sistema operativo diverso si può installare una macchina virtuale Ubuntu o altrimenti reperire i file VHDL ed eseguire l'esperienza a partire dal punto 4, come indicato sopra
- è disponibile il download dell'archivio ZIP di tutte le note

1. note per l'installazione del generatore Gezel di codice VHDL
2. note per l'installazione e lancio di Quartus Prime Lite 16.1 su Ubuntu 16.04
3. note sugli assegnamenti a un progetto Quartus
4. note sulla regolazione del clock con Quartus TimeQuest Analysis
5. note sull'uso di Quartus ModelSim

letture raccomandate:

Schaumont (2012) Cap. 1, Sez. 1.1.1

Quartus Prime Introduction Using VHDL Designs - For Quartus Prime 16.1; Intel FPGA University Program

Using TimeQuest Timing Analyzer - For Quartus Prime 16.1, Sez. 1-2, 4; Intel FPGA University Program

Introduction to Simulation of VHDL Designs - For Quartus Prime 16.1; Intel FPGA University Program

per ulteriore consultazione:

Schaumont (2012) App. A.1

altri materiali utili per l'esperienza di laboratorio proposta:

Quartus Prime Lite 16.1 download

Intel FPGA University Program Installer