

# Construirea ierarhică a arhitecturilor în Verilog

Probleme propuse

Oprîtoiu Flavius  
flavius.opritoiu@cs.upt.ro

October 16, 2024

# Problema 1

Construiți în Verilog unitatea Full Adder Cell (FAC) (numită *fac*). Modulul are 3 intrări de 1-bit  $x$ ,  $y$  și  $ci$ , precum și 2 ieșiri de 1-bit  $z$  și  $co$ . Scrieți codul într-un fișier numit *fac.v*.

## Problema 2

Construiți un testbench pentru verificarea exhaustivă a unității *fac* de mai sus, modul numit *fac\_tb*. Conținutul noului modul poate fi inclus în același fișier *fac.v*, împreună cu modulul *fac*.

Construiți un fișier script, numit *run\_fac.txt*, pentru compilare, lansare și rularea simulării unității testbench *fac\_tb*.

## Problema 3

Utilizând 2 instanțe *fac*, construiți un sumator pe 2 biți, numit *add2b*, având 2 intrări a câte 2-biți  $x$  și  $y$ , o intrare de 1-bit  $ci$ , respectiv având ieșirea  $o$  reprezentând suma pe 2 biți și transportul de ieșire,  $co$ . Scrieți codul în fișierul *add2b.v*.

Construiți un testbench pentru verificarea exhaustivă a implementării, numit *add2b\_tb* și fișierul script asociat, *run\_add2b.txt*

## Problema 4

Proiectați un modul *cmp2b* pentru compararea a 2 numere fără semn pe 2 biți și care 2 intrări  $x$  și  $y$  pentru cele 2 numere de comparat, respectiv are 3 ieșiri a câte 1-bit: *eq*(egal), *lt*(mai mic decât) și *gt*(mai mare decât). Scrieți codul în fișierul *cmp2b.v*.

## Problema 5

Folosind unitatea *cmp2b*, construiți un comparator pentru valori fără semn pe 4-biți, numit *cmp4b*, având intrările de 4-biți *x* și *y*, respectiv 3 ieșiri a câte 1-bit: *eq*, *lt* și *gt*. Scrieți codul în fișierul *cmp4b.v*.

Construiți un modul testbench pentru verificarea exghaustivă a implementării, numit *cmp4b\_tb* împreună cu fișierul script asociat, numit *run\_cmp4b.txt*.

## Problema 6

Realizați un modul pentru adunarea a 2 numere întregi reprezentate în codul C1, numit `c1_add4b`, care are 2 intrări a câte 4-biți  $x$  și  $y$  împreună cu intrarea de 1 bit  $ci$ , generând la ieșiri semnalele sumă  $z$  pe 4 biți (fără transport de ieșire, datorită *end around carry*).

Construiți un modul testbench pentru verificarea exghaustivă a implementării, numit `c1add4b_tb` împreună cu fișierul script asociat, numit `run_c1add4b.txt`.