Compito di linguaggi di descrizione dell'hardware

Esercizio 1

Si realizzi un modello comportamentale in VHDL di un componente che deve verificare che i suoi due ingressi x e y non rimangano a 1 per un tempo maggiore di τ. In tale caso l'uscita, normalmente a 0, si porta a 1 e mantiene tale valore fino a quando un ulteriore ingresso (*reset*) non si porta a 1.
Soluzione

```
library IEEE;
use IEEE.std_logic_1164.all;

entity eser is
  generic(t: time);
  port(x,y: in std_logic;
       reset: in std_logic;
       w: out std_logic);
end entity eser;

architecture behav of eser is
signal and_tmp: std_logic;
begin
  and_tmp<=x and y after t;

  process(and_tmp,reset)
  begin
    if (reset='0') then
      if (and_tmp='1') then
        w <= '1';
      end if;
    elsif (reset='1') then
         w<='0';
    end if;
  end process;
end architecture behav;
```

Esercizio 2

Si realizzi un modello comportamentale in VHDL di una rete combinatoria che riceve in ingresso una parola di 15 bit $a_{14..0}$ che rappresenta un numero naturale j codificato con un codice termometrico: $a_i = 1\ \forall i < j \wedge b_i = 0\ \forall i \geq j$ (nel caso particolare in cui $j = 0$ a é composta di tutti 0). Le uscite della rete sono v e $b_{3..0}$. Compito della rete é: 1) verificare che la parola di ingresso sia priva di errori e produrre $v = 1$, mentre in presenza di errori si ha $v = 0$; 2) produrre su b la codifica binaria di a. Si raccomanda di realizzare un codice compatto.

$a_{14}a_{13}....a_1a_0$	j
00000000000000	0
00000000000001	1
00000000000011	2
00000000000111	3
....	....
11111111111111	14
00000000000010	errore
....	....
00000100111111	errore
....	....

Soluzione

```
library IEEE;
use IEEE.std_logic_1164.all,ieee.numeric_std.all;

entity eser is
  port(a: in std_logic_vector(14 downto 0);
       b: out std_logic_vector(3 downto 0);
       v: out std_logic);
end entity eser;

architecture behav of eser is
begin

  process(a)
  variable n: unsigned(3 downto 0);
  variable valid: std_logic;
  begin
    n:="0000";
    valid:='1';
    for i in 14 downto 0 loop
      if (i>0) then
      if (a(i)='1') and (a(i-1)='0') then
        valid:='0';
      end if;
      end if;
      if (a(i)='1') then
        n:=n+"0001";
      end if;
    end loop;
    v<=valid;
    b<=std_logic_vector(n);
  end process;
end architecture behav;
```

Esercizio 3

Si consideri il seguente algoritmo:

```
read a;
read b;
read c;
read d;
read e;
read f;
u0:=a*b;
u1:=c+d;
u2:=e*f;
u3:=a+e;
u4:=b*c;
u5:=u1+u0;
u6:=u2*u1;
write u6;
u7:=u3*u4;
u8:=u6+u3;
u9:=u8+u7;
u10:=u5+u9;
write u10;
```

si tracci il DFG e si determinino lo scheduling ASAP o quello ALAP nell'ipotesi di ciclo singolo. Nell'ipotesi che siano disponibili 2 adder, 1 moltiplicatore e 2 bus, si determini poi uno scheduling che minimizza la latenza. Si indichino chiaramente la latenza relativa dello e il binding per tale scheduling.