Compito di linguaggi di descrizione dell'hardware

Esercizio 1

Si realizzi un modello comportamentale in VHDL di un componente che ha due ingressi d, clk e un uscita y. Come parametri *generic* la rete riceve t_R e δ. In presenza di un fronte di salita di clk (che supporremo avvenire al tempo t_0) la rete campiona d, che viene poi campionato di nuovo a $t_0 + \delta$ e a $t_0 + 2\delta$. L'uscita y viene calcolata con un operazione di voting sui 3 valori campionati. Un nuovo valore di uscita viene prodotto con un ritardo t_R.

```
library ieee;
use ieee.std_logic_1164.all;

entity rz is
  generic(tr,del: time);
  port(d,clk: in std_logic;
       y: out std_logic);
end entity rz;

architecture behav of rz is
signal clk1,clk2: std_logic;  -- delayed clock signals
signal y0,y1,y2: std_logic;   -- sampled values
begin
  process(clk,clk1,clk2)
  begin
    if (rising_edge(clk)) then
      y0<=d;
    elsif (rising_edge(clk1)) then
        y1<=d;
      elsif (rising_edge(clk2)) then
          y2<=d;
    end if;
  end process;
  clk1<=clk after del;
  clk2<=clk after 2*del;

  y<=(y0 and y1) or (y0 and y2) or (y1 and y2) after tr;

end architecture behav;
```

Esercizio 2

Si descriva al livello comportamentale in VHDL una rete combinatoria che riceve

in ingresso due parole a di 8 bit e b di 4 bit. La rete produce il valore di uscita 1 se la parola b ricorre almeno una volta in a, altrimenti si produce il valore 0.

```
library ieee;
use ieee.std_logic_1164.all;

entity comp is
  port(a: in std_logic_vector(7 downto 0);
       b: in std_logic_vector(3 downto 0);
       y: out std_logic);
end entity comp;

architecture behav of comp is
begin
  process(a,b)
  begin
    if (b=a(7 downto 4) or
        b=a(6 downto 3) or
        b=a(5 downto 2) or
        b=a(4 downto 1) or
        b=a(3 downto 0)) then
      y<='1';
    else
      y<='0';
    end if;
  end process;
end architecture behav;
```

Esercizio 3

Si consideri il seguente algoritmo:

```
0. u0:=a+b;
1. u1:=c+d;
2. u2:=f+g;
3. u3:=u1*e;
4. u4:=u2*h;
5. u5:=u0*u1;
6. u6:=u3+i;
7. u7:=u6+u4;
8. u8:=u5+u7;
```

Si tracci il DFG e si determini uno scheduling a latenza minima che minimizza il numero di risorse utilizzate (nell'ipotesi di ciclo singolo).

Si considerino poi i seguenti dati:

- ritardi di multiplexer e flip-flop trascurabili;
- ritardo massimo di un sommatore = $0.9ns$;
- ritardo massimo di un moltiplicatore = $4.5ns$.

Si determini prima se la rete puó funzionare nell'ipotesi di ciclo singolo con una frequenza di clock di 200MHz e si calcoli la latenza in ns. Si supponga poi che la frequenza di clock sia di 400MHz e si individui una tecnica che consente il corretto funzionamento della rete in questo caso. Utilizzando tale tecnica, si tracci il DFG con il nuovo scheduling calcolando la latenza in ns.