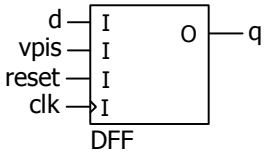
4. Vaja: Sekvenčni gradniki

```
P: process(clk, reset)
begin
  if reset='1' then
    izhodi <= "00000...";
  elsif rising_edge(clk) then
    ... logika ...
  end if;
end process;
```

SINHRONI PROCES Z RESETOM

Opis sinhronih gradnikov začnemo s pogojem za reset, ki mu sledi pogoj za fronto ure. Vsi stavki, ki opisujejo delovanje gradnika, so zapisani znotraj tega pogoja za fronto ure.

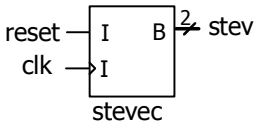
1. D flip-flop



a) Naredi opis D flip-flopa z asinhronim reset signalom in vhodom za omogočanje vpisa. Flip-flop prenese podatek iz vhoda (d) na izhod (q) ob fronti ure in pogoju, da je signal vpis na '1'. Ob aktivnem resetu naj gre izhod takoj na '0'.

b) Spremeni opis flip-flopa, tako da bo signal reset deloval sinhrono s fonto ure. Ob aktivnem resetu in fronti ure naj gre izhod na '0', ne glede na stanje signala vpis.

2. Binarni števec



Števci so sekvenčna vezja s povratno zanko, ki jih opisujemo s sinhronim procesom. Povratna zanka povzroči, da je naslednje stanje izhoda odvisno od trenutnega stanja (npr. izhod števca je za 1 večji od trenutnega izhoda: `stev <= stev + 1;`). Če se zunanji signal pojavlja na levi in na desni strani, ga deklariramo kot **buffer**.

a) Naredi opis dvobitnega binarnega števca s signalom reset in signalom za omogočanje štetja. Števec naj ob vsaki naraščajoči fonti ure poveča izhodno vrednost za 1.

b) Kako bi definiral začetno stanje števca brez uporabe reset signala?

c) Dodaj v opis še en števec (del), ki se obnaša kot delilnik ure za prvotni števec. Vsakokrat, ko del prišteje do 100.000, naj se postavi na 0 in hkrati omogoči povečanje prvotnega dvobitnega števca.