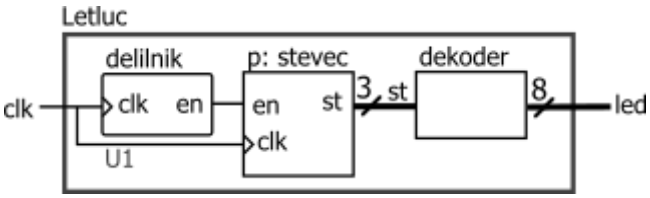
Opis strukture digitalnih vezij: vhdl_str

5 Leteča luč

Naredi vezje za zaporedno prižiganje LED s povezavo komponent v jeziku VHDL in preizkusi delovanje na razvojnem sistemu.



a. Odpakiraj in kopiraj datoteke projekta iz predloge za razvojno ploščo DE0 Nano: letluc.zip v mapo (npr: c:\proj\iv\ime\letluc). Z dvoklikom na **letluc.qpf** odpri program Quartus. V datoteki **letluc.vhd** deklariraj dva notranja signala: 3-bitni nepredznačen vektor **st** in enobitni **en.** Naredi opis 3-bitnega števca, ki se ob fronti ure clk in pogoju en='1' povečuje za 1.

b. Odpri datoteko delilnik.vhd, ki vsebuje delilnik ure s števcem po modulu M. Dodaj delilnik ure kot komponento v vezje **letluc.vhd** - komponento deklariraj med **architecture** in **begin**. Dodaj v opis vezja stavke za povezavo komponente in nastavi vrednost parametra M na 2. Preizkusi delovanje števca z delilnikom na simulaciji.

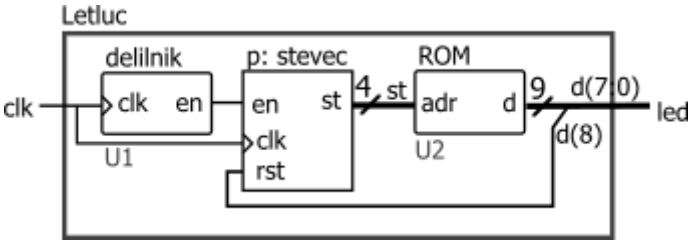
```
U1: delilnik generic map (M => 2)
             port map (clk => clk, en => en);
```

c. Naredi še eno novo datoteko z imenom **dekoder.vhd**, v kateri bo kombinacijski dekodirnik, ki pretvori 3-bitno binarno kombinacijo iz števca v 8-bitno kombinacijo na izhodu **led**. Dekodirnik naj deluje tako, da bo bomo dobili ob zaporednih vrednostih na vhodu vtis potovanja prižgane LED. Npr. kombinacija "000" se pretvori v "10000000", "001" v "01000000", ... Opiši dekodirnik s stavkom **with...select** (glej primer: vhdl_pretok.html#Dekod) in vključi v celotno vezje dekodirnik kot komponento.

d. Prevedi vezje in poglej shemo RTL prevedene kode (Tools > Netlist Viewers > RTL Viewer). Preizkusi delovanje vezja na simulaciji, nato pa spremenil modul štetja delilnika na 10000000, ponovno prevedi vezje in preizkusi delovanje na razvojnem sistemu.

6 Mikrosekvenčnik

Uporabi strukturo vezja iz 5. vaje in ga pretvori v mikro-programiran krmilnik oz. mikro-sekvenčnik. Delovanje mikrosekvenčnika določa koda v pomnilniku ROM.



a. Povečaj števec na 4 bite in dodaj notranji signal **rst,** ki povzroči, da se števec postavi na 0. Signal **rst** naj deluje sinhrono z deljeno uro !

b. Zamenjaj dekodirnik s pomnilnikom ROM, ki ima 4-bitni nepredznačen vhod **adr** in 9-bitni izhod **d**. Deklariraj podatkovni tip in notranji signal, ki bo predstavljal vsebino pomnilnika:

```
type memory is array (0 to 15) of unsigned(8 downto 0);
signal ROM: memory := ( "010000000", "001000000",... "100000000");
```

Kombinacije v pomnilniku naj bodo napisane tako, da se bo naredil reset po 14 ciklih:

adr	d	adr	d
0000	010000000	1000	000000010
0001	001000000	1001	000000100
0010	000100000	1010	000001000
0011	000010000	1011	000010000
0100	000001000	1100	000100000
0101	000000100	1101	101000000
0110	000000010	1110	100000000
0111	000000001	1111	100000000

V arhitekturnem opisu pomnilnika zapiši stavek, ki da na izhod podatek iz pomnilniške lokacije, ki jo določa naslov **adr**:

```
d <= ROM(to_integer(adr));
```

c. Dodaj pomnilnik v vezje **letluc** kot komponento in ga poveži. Spodnjih 8 bitov naj bo povezanih na izhodne LED, najvišji bit pa na **rst** od števca.
Prevedi vezje in preglej shemo RTL.

d. Preizkusi delovanje mikrosekvenčnika na simulaciji in na razvojnem sistemu.