G2. Prikaz sličice

V vezje za prikaz slike na monitorju bomo dodali logiko, ki bo prikazala v vogalu monitorja enobitno sličico. Prikaz sličice deluje tako, da se na osnovi trenutnih koordinat (**cx**, **cy**) in signala **en**, ki pove ali smo znotraj vidnega dela slike, odločamo kakšna bo vrednost barvnega izhoda **rgb**.

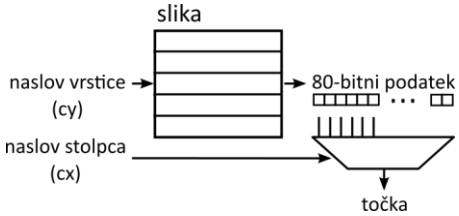
Slika v pomnilniku ROM

Slika bo v vezju shranjena v pomnilniku ROM, ki ga v jeziku VHDL deklariramo kot dvodimenzionalno zbirko. Deklarirajmo npr. ROM za logotip Red Pitaye v velikosti 80 x 20 točk:

```
type logo is array(0 to 19) of std_logic_vector(0 to 79);

signal slika: logo := (
 "00000000100000000000000000000000000000000000000000000000000000000000000000000000",
 "00000000100000000000000000000000000000000000000000000000000000000000000000000000",
 "00000001100000000000000000000000000000000000000000000000000000000000000000000000",
 "00000001000000000000000000000000000000000000000000000000000000000000000000000000",
 "00000001001000000000000000000000000000000000000000000000000000000000000000000000",
 "00000001001000000000000000000000000000100000000000000000000000000000000000000000",
 "00000001111000000000000000000000000000100000000001001000000000000000000000000000",
 "00001001111000000000000000000000000000100000000000010000000000000000000000000000",
 "00010011111001000001011001111100011111100011100010111110111111001000001011111110",
 "00011111111111000001100010000010100000100100001001001000000000010100000100000001",
 "00011111111111000001000010000010100000100100001001001000000000010100000100000001",
 "00011111111111000001000010000010100000100100001001001000001111110100000100111111",
 "01111110000000000001000011111000100000100100001001001000010000010100000101000001",
 "00111110011001100001000010000001000001001000010010010000100000101000001010000001",
 "00111100011001100001000010000001000001001000010010010000100000101000001010000001",
 "00011110000000000001000010000001000001001000010010010000100000101000001010000001",
 "00011111111111000001000001111100011111100111110001000111001111110011111100111111",
 "00011111111111000000000000000000000000010000000000000000000000000000000100000000",
 "00001111111111000000000000000000000000010000000000000000000000000000000100000000",
 "00000011111100000000000000000000000000010000000000000000000001111110000000000");
```

Deklaracijo dodaj v testno vezje iz prejšnje vaje (VGAtest), na mestu kjer so deklaracije notranjih signalov. Iz pomnilnika beremo točke, tako da določimo naslov vrstice (cy) in iz 80-bitne pomnilniške besede izluščimo posamezen bit (cx):



V jeziku VHDL naslovimo dvodimenzionalno zbirko podobno kot enodimenzinalni vektor. Indeks mora biti tipa integer, zato uporabimo funkcijo za pretvorbo:

```
tocka <= slika( to_integer(cy) )( to_integer(cx) );
```

Napiši proces, ki preverja ali je trenutna točka vidna in ali se nahaja v območju [0,0] in (80,20). Če smo notraj teh koordinat, naj preveri vrednost ustrezno ležeče točke iz pomnilnika in pri '1' naj da na izhod rgb3 belo barvo ("111"), pri '0' pa črno ("000"). Kadar smo izven območja prikaza slike, pa naj bo na izhodu vrednost ozadja, ki pride iz komponente VGA.

Opiši vezje in preveri delovanje na simulaciji in na testni strukturi. Poskusi narediti algoritem, ki prikaže sliko na poljubnem mestu na zaslonu, ki je definirano s konstantnimi koordinatami (x, y).