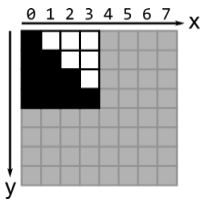
10. vaja: Grafika

Izdelali bomo komponento vezja za prikazovanje grafike na računalniškem monitorju. Grafiko prikazujemo v obliki zaporedja točk, ki jih digitalni sistem pošilja na monitor. Naloga vezja bo določitev 3-bitne vrednosti (barve) posamezne točke na podlagi trenutne koordinate (x, y).

Branje točk iz pomnilnika



Zaradi lažje izvedbe simulacije bomo najprej naredili vezje za prikaz podatkov iz pomnilnika ROM, ki vsebuje 16 eno-bitnih vrednosti. Podatke bomo razdelili na 4 vrstice po 4 bite in bodo predstavljali binarno sliko velikosti 4x4 točke, ki jo bomo postavili na mrežo velikosti 8x8 točk.

Za pomnilnik deklariramo 16-bitni vektor in mu določimo konstantno vrednost. Pomnilnik opisuje stavek, ki izbere eno-bitni izhod (**data**) ob 4-bitnem naslovu (**adr**):

spletno orodje

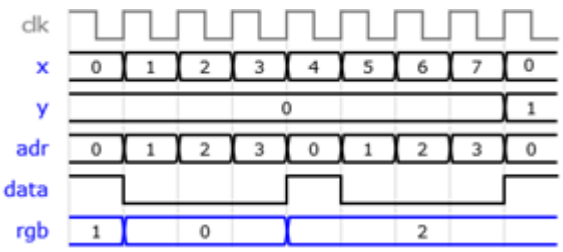
Deklaracija konstante:

```
rom : u16;
rom = 0b1111011100110001

Opis branja ROM:

data = rom(adr)
```

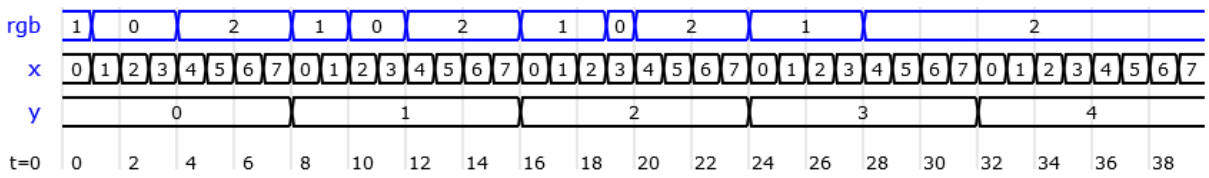
VHDL

```
constant rom : unsigned(15 downto 0) :=
"1111" &
"0111" &
"0011" &
"0001";

data <= rom(to_integer(adr));
```

1. Model vezja naj vsebuje 3-bitna števca **x** in **y**. Števec **x** naj se povečuje ob vsakem ciklu ure, števec **y** pa le, kadar je x = 7. Opiši vezje s števcema, pomnilnikom in 3-bitnim izhodom **rgb**. Naslov pomnilnika naj določata prva dva bita števca x. Kadar je števec manjši od 4, naj **rgb** dobi vrednost data, sicer pa vrednost 2.

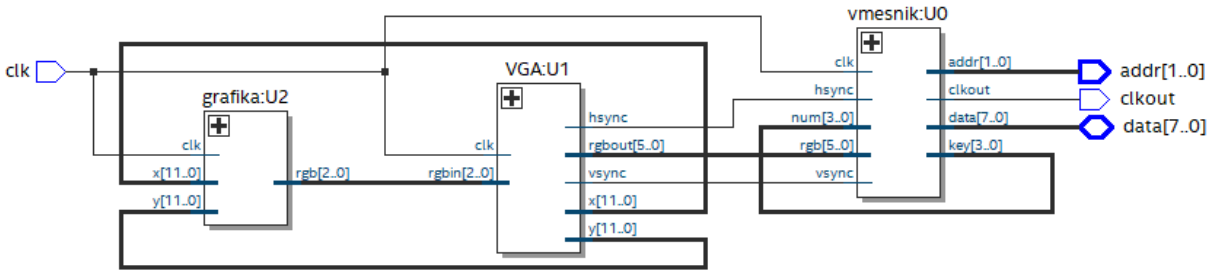


2. Na izhod **rgb** pošiljamo zaporedne točke, najprej iz prve vrstice, nato druge, itd. Ugotovi kako je potrebno naslavljati pomnilnik, da bomo prebrali celo sliko:



3. Ugotovi, kaj je potrebno spremeniti v kodi, da bi bila sličica iz pomnilnika ROM zamaknjena za dve mesti v desno. (namig: spremeni naslov in pogoje za izhod).

Projekt za razvojno ploščo



1. Odpri projekt B18vga.qar v programu Quartus. V projektu je digitalni sistem iz treh komponent: vmesnik za razvojno ploščo, generator signalov za monitor VGA in grafika. Generator VGA skrbi za časovni potek prikazovanja slike ločljivosti **800 x 600** točk s 6-bitno barvno globino. Na komponento grafika sta povezani 12-bitni koordinati trenutne točke (**x**, **y**) iz komponente pa pride 3-bitni podatek o barvi točke **rgb** (rdeča, zelena in modra).

2. Uredi komponento grafika, ki vsebuje 1024-bitno konstanto rom za prikaz slike kroga iz 32x32 točk:

```
constant rom: unsigned((32*32)-1 downto 0) := (
"00000000000000000000000000000000" &
"00000000001111111111100000000000" &
"00000001111111111111111000000000" &
```

   Dodaj VHDL proces in stavke, ki iz pomnilnika preberejo vrednost in jo pošljejo na **rgb,** ko sta koordinati x in y med vključno 0 in 31, sicer pa naj gre na **rgb** vrednost 2. Prevedi projekt in preizkusi delovanje na razvojni plošči.

3. Spremeni logiko za določanje barv, tako da bo celotno ozadje slike modro, prikazan krog pa rumene barve.

4. Dodaj stavke, ki prikažejo premaknjen krog, tako da se začne prikaz na koordinatah (100, 50).

5. Dodaj še stavke, ki prikažejo še en krog v drugačni barvi in na drugem mestu na zaslonu. Razmisli, katere signale je potrebno definirati in kateri ostanejo enaki.