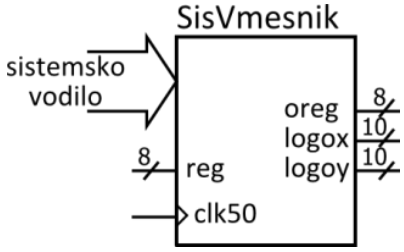
8. vaja: Sistemski vmesnik

Naredi vmesnik za sistemsko vodilo na Red Pitayi. Poenostavljeno sistemsko vodilo ima priključke:

```
clk_i, rstn_i: in STD_LOGIC;                    -- ura in negativni reset signal
adr_i: in STD_LOGIC_VECTOR (31 downto 0);       -- 32 bitni naslov, dekodiramo spodnjih 20 bitov

wdata_i: in STD_LOGIC_VECTOR (31 downto 0);     -- 32 bitni vhodni podatek
wen_i: in STD_LOGIC;                            -- signal za vpis v periferne enote

ren_i: in STD_LOGIC;                            -- signal za branje periferije
rdata_o: out STD_LOGIC_VECTOR (31 downto 0);    -- 32 bitni izhodni podatek

err_o: out STD_LOGIC;                           -- javljanje napake na vodilu (logična 0)
ack_o: out STD_LOGIC;                           -- potrditev prenosa (logična 1 za potrditev)
```

Vmesnik naj dekodira naslednje naslove:

naslov (spodnjih 20 bitov)	register	število bitov	R/W
0x00000	reg (vmesnik iz 1. vaje)	8	R
0x00004	oreg (vmesnik iz 1. vaje)	8	R/W (notranji rego)
0x00008	logox	10	R/W (notranji regx)
0x0000C	logoy	10	R/W (notranji regy)
0x10000 – 0x103FC	ram(0 to 255)	255 x 32	W



Vmesnik naj ima poleg signalov sistemskega vodila še vhodno uro **clk50,** 8-bitni vhodni signal **reg**, 8-bitni izhod **oreg** in dva 10 bitna izhoda **logox** in **logoy**.

8.1 Vpis v registre

Periferni registri dobijo vrednost iz sistemskega vodila ob sistemski uri (**clk_i**). Stanje registrov bomo uporabili za krmiljenje VGA grafike, ki uporablja uro **clk50**, zato bomo uporabili v vezju podvojene registre in dva procesa.

Prvi sinhroni proces dela z uro **clk_i** in ob resetu postavi notranje registre (**regx, regy, rego**) na 0, ob aktivnem signalu **wen_i** pa vpiše podatek iz **wdata_i** v:

- **rego**, kadar je spodnjih 20 bitov naslova X"00004",
- **regx**, ob naslovu X"00008"
- **regy**, ob naslovu X"0000C"

Drugi sinhroni proces dela z uro **clk50** in stalno prepisuje vrednost iz notranjih registrov v izhodne registre **oreg, logox** in **logoy.**

8.2 Vpis v pomnilniški blok

Vmesnik naj vsebuje tudi 1kB pomnilnika v obliki 256 vektorjev velikosti 32-bitov. V vezju deklariramo zbirko in signal za pomnilnik:

```
type ram1k is array(0 to 255) of std_logic_vector(31 downto 0);
signal ram: ram1k;
```

V pomnilnik pišemo z uro **clk_i**, kadar je aktiven signalu **wen_i** in naslov adr_i >= X"10000", s stavkom:

```
ram( to_integer( unsigned( adr_i(9 downto 2) ) ) ) <= wdata_i;
```

8.3 Branje notranjih registrov in vhodov

Naredi še logiko za branje registrov, ki se izvaja v sinhronem procesu ob **clk_i** in aktivnem signalu **ren_i.** Preizkusi delovanje vmesnika s simulacijo, kjer nastavi sistemsko podatkovno vodilo, naslov in impulz na kontrolnem vhodu **wen_i** ali **ren_i.**