4. vaja: slika s pomnilnikom

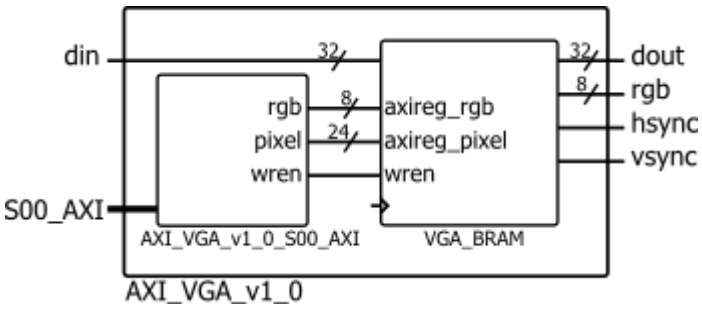
Pri vaji bomo:

- razvili komponento AXI, ki vsebuje generator sinhronizacijskih signalov za VGA monitor
- naredili VHDL model pomnilnika za slikovne točke v tej komponenti in povezave za nastavljanje barve točk in ozadja prek vodila AXI
- naredili sistem in demonstracijsko aplikacijo

4.1 Komponenta AXI z generatorjem VGA slike

- V programu Vivado odpri projekt, nastavi privzet jezik VHDL in naredi novo komponento vrste AXI Lite (**Tools > Create and Package IP**, **Create a new AXI4 peripheral...**).
  Nova komponenta naj se imenuje **AXI_VGA**, uporabi privzete nastavitve: Lite, Slave, Data Width: 32, Number of registers: 4 in odpri projekt za urejanje (**Edit IP**).

- V projekt dodaj datoteko VGA_BRAM.vhd, ki vsebuje števce in logiko za prikaz slike na monitorju in slikovni tok, narejen tako kot pri komponenti VGAstream iz 2. vaje. Vezje VGA_BRAM vsebuje nekaj signalov, ki jih bomo povezali na registre vodila AXI in signale, ki jih prevežemo na priključke komponente AXI_VGA po shemi:



- Uredi notranjo komponento AXI_VGA_v1_0_S00_AXI. Najprej dodaj 3 izhodne priključke: 8-bitni **rgb,** 24-bitni **pixel** in enobitni **wren.** S prireditvenimi stavki na koncu opisa vezja poveži signal **rgb** na register **slv_reg0,** signal **pixel** pa na **slv_reg1.** Signal **wren** pa je kontrolni signal, ki naj se postavi na 1 ob vpisu nove vrednosti v register **slv_reg1**, kot prikazuje izsek kode:

```
208    process (S_AXI_ACLK)
209    variable loc_addr :std_logic_vector(OPT_MEM_ADDR_BITS downto 0);
210    begin
211      if rising_edge(S_AXI_ACLK) then
212        if S_AXI_ARESETN = '0' then
213          slv_reg0 <= (others => '0');
214          slv_reg1 <= (others => '0');
215          slv_reg2 <= (others => '0');
216          slv_reg3 <= (others => '0');
217          wren <= '0'; -- kontrolni signal za pisanje
218        else
219          wren <= '0'; -- kontrolni signal za pisanje
220          loc_addr := axi_awaddr(ADDR_LSB + OPT_MEM_ADDR_BITS downto ADDR_LSB);
221          if (slv_reg_wren = '1') then
222            case loc_addr is
223              when b"00" =>
224                for byte_index in 0 to (C_S_AXI_DATA_WIDTH/8-1) loop
225                  if ( S_AXI_WSTRB(byte_index) = '1' ) then
226                    -- Respective byte enables are asserted as per write strobes
227                    -- slave registor 0
228                    slv_reg0(byte_index*8+7 downto byte_index*8) <= S_AXI_WDATA(byte_index*8+7
229                  end if;
230                end loop;
231              when b"01" =>
232                wren <= '1'; -- kontrolni signal za pisanje
```

- Uredi komponento AXI_VGA_v1_0.vhd, v kateri dodaj zunanje priključke za signale **din, dout, rgb, hsync** in **vsync**, deklariraj in ustrezno poveži novo komponento AXI_VGA. Ne pozabi tudi deklarirati in povezati nove signale komponente AXI_VGA_v1_0_S00_AXI.

- Dopolni opis komponente VGA_BRAM, tako da bo vsebovala blokovni pomnilnik za 65536 8-bitnih slikovnih točk.

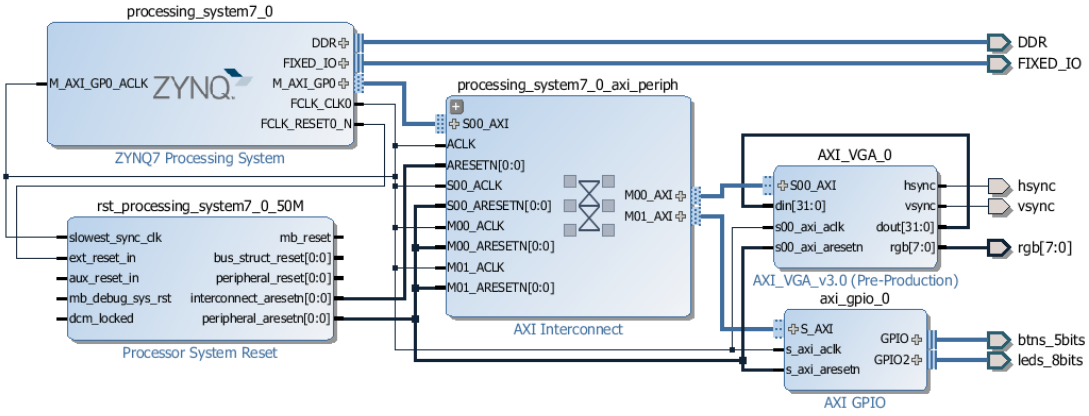
```
45 type mem is array(0 to 65535) of std_logic_vector(7 downto 0);
46 signal bram : mem;
```

- Slikovne točke naj se vpišejo v pomnilnik ob aktivnem signalu wren. Koordinate slikovnih točk predstavlja zgornjih 16 bitov signala **axireg_pix**, spodnjih 8 bitov pa določa barvo točke (zgornji biti so naslov pomnilnika, spodnji pa podatek, ki se vpiše v pomnilnik). Napiši sinhroni proces, ki ob uri bere podatke iz pomnilnika na naslovu, ki ga določa trenutna točka (**ty & tx**) ter vpisuje nove podatke ob aktivnem **wren.**

- Naredi sintezo in simulacijo vezja ter dokončaj pakiranje nove komponente (dvoklik na component.xml, **Merge Changes**...).

4.2 Testni sistem

- Izdelaj testni digitalni sistem na podlagi predloge iz projekta VGARAMtest.zip.



4.3 Testna aplikacija

- Preizkusi delovanje na razvojni plošči Zedboard z aplikacijo, ki nastavlja vrednost barve ozadja in poljubnih točk. V programu definiraj naslove komponente AXI in uporabi funkciji Xil_Out8() in Xil_Out32():

```
#define AXIRGB 0x43c00000
#define AXIPIX 0x43c00004

// nastavi barvo ozadja
Xil_Out8(AXIRGB, 50);

// nastavi točko (x,y) na barvo c
Xil_Out32(AXIPIX, (y << 16) | (x << 8) | c);
```

- Poskusi še testno aplikacijo, ki pobriše zaslon in nariše naključne črte: Test2.c.