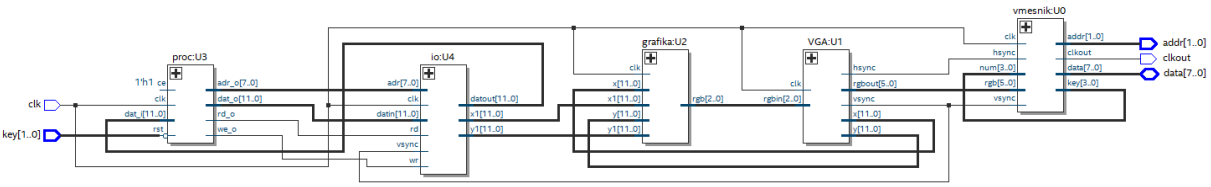
11. vaja: Grafični sistem s procesorjem

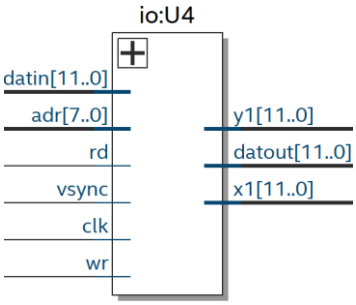
Vezju za prikaz grafike bomo dodali procesor z vmesnikom.



Vhodno-izhodni vmesnik

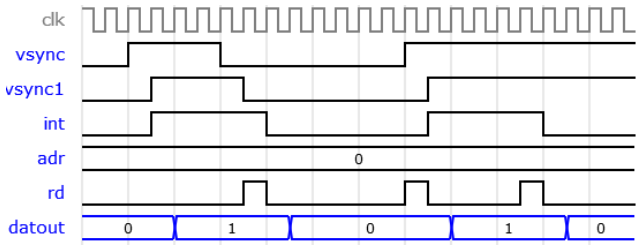
Naredi vhodno-izhodni vmesnik za 12-bitni procesor. Vmesnik naj nastavlja 2 izhodna vektorja **x1** in **y1** in bere stanje prekinitve, ki jo sproži signal **vsync.** To je signal iz vezja VGA, ki se aktivira 72x na sekundo (vsakokrat, ko se osveži slika na monitorju).

1. Naredi model vezja **io.vhd** s signali za povezavo s procesorjem in izhodi registrov.
   - **adr** je 8-bitno naslovno vodilo
   - **datin** je 12-bitni vhod na katerega procesor piše
   - **rd** in **wr** označujeta, da procesor bere oz. piše
   - **datout** je 12-bitni podatkovni izhod iz katerega procesor bere
   - **vsync** je 1-bitni sinhronizacijski impulz
   - **x1** in **y1** sta 12-bitna izhodna registra



Napiši logiko za nastavljanje izhodnih registrov. Registra shranita vrednost iz vhoda ob uri, pogoju wr=1 in ustreznem naslovu: **x1** ob adr=0 in **y1** ob adr=1.

2. Prekinitvena logika: signal vsync zakasni za en cikel ure (vsync1) in uporabi kombinacijo obeh signalov za postavitev enobitnega prekinitvenega signala **int** na 1. Prekinitev naj bo vezana na izhodno vodilo (**datout**). Kadar je prekinitev postavljena in procesor bere (**rd**) iz naslova 0 (**adr**), naj se prekinitev resetira:



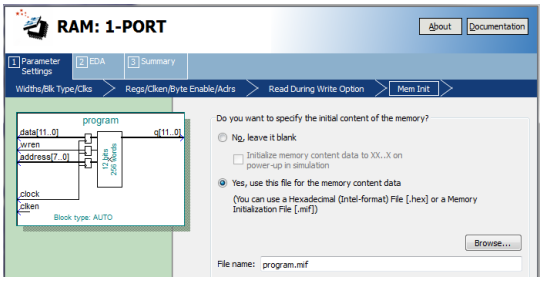
Preizkus na razvojni plošči

1. Odpri projekt B18sistem.qar v katerega dodaj svojo komponento za prikaz grafike in vhodno-izhodni vmesnik. V datoteki s programskim pomnilnikom je program, ki nastavi **x1** in **y1**, nato pa ob vsaki prekinitvi poveča **x1**. Preizkusi delovanje na razvojni plošči !

```
start:  lda x          |        lda x
        outp 0         |        add 1
        lda y          |        sta x
        outp 1         |        jmp start
loop:   inp 0          |  x     db 5
        jze loop       |  y     db 10
```

2. Spremeni program tako, da se bo sličica premikala čez zaslon dvakrat hitreje!

3. Dodaj opis pomnilnika v obliki komponente IP: **Tools > IP Catalog**, pod **Basic Functions**, **On Chip Memory** izberi: RAM: 1-PORT in določi ime: **program**.
   Odpre se okno za nastavitev komponente (MegaWizard):
   - v prvem zavihku nastavi širino pomnilnika 12 bits,
   - v drugem odstrani kljukico pri **q** in dodaj kljukico na **Create one clock enable signal**..., tretjega ni potrebno spremeniti,
   - v četrtem pa dodaj kljukico na: **Yes, use this file**... in z gumbom **Browse...** izberi program11.mif, dodaj še kljukico pri **Allow In-System Memory Content Editor** ter potrdi vnos IP v projekt.



Datoteka *.mif je tekstovna datoteka z opisom vsebine pomnilnika:

```
WIDTH=12;
DEPTH=256;
ADDRESS_RADIX=HEX;
DATA_RADIX=HEX;
CONTENT BEGIN
00 : 10a;
01 : 700;
02 : 10b;
03 : 701;
04 : 300;
05 : 504;
06 : 10a;
07 : 80c;
08 : 20a;
09 : 400;
0a : 005;
0b : 00a;
0c : 001;
END;
```

Vsebino pomnilnika lahko sedaj spreminjamo, ne da bi ponovno prevajali celotno vezje. Novo kodo iz prevajalnika zapišemo v datoteko in v Quartusu uporabimo orodje: **Tools > In System Memory Content Editor**, v katerem nastavimo Hardware: USB-Blaster, kliknemo na instanco pomnilnika (index 0), izberemo **Import Data from File** in **Write Data**...