Univerza v Ljubljani
Fakulteta za elektrotehniko

Dodali bomo vhodno logiko mikroprocesorja, da bo na podlagi vhodne informacije krmilil enostavno igro.

7.1 Zaznavanje trka

Z mikroprocesorjem želimo brati dogodek, ko pride do trka ali ko žogica doseže določeno koordinato. To so hitri dogodki, ki bi jih običajno povezali na prekinitveni vhod procesorja. Ker naš CPU nima prekinitev, bomo dodali nekaj kode, ki predstavlja vhodno logiko - zaznavanje prehoda signala iz 0 na 1, postavljanje 'prekinitvene' zastavice in brisanje zastavice ob pisanju na izhodna vrata.

a. V **proc.vhd** dodaj enobitni vhod **trk** in dva enobitna notranja signala: **trk1** in **int1.** Napiši sinhroni stavek, ki zazna spremembo vhoda **trk** na '1' in ob tem postavi **int1** na '1'. Kadar procesor piše na vrata z naslovom 2, pa naj ta signal resetira. Zaznavo trka naredimo v nekaj korakih v procesu ob naraščajoči fronti ure clk:
   - vhod zakasnimo za eno periodo: trk1 <= trk;
   - ob pogoju trk1='0' and trk='1' postavimo: int1 <= '1';
   - ob pogoju we_o='1' and ce='1' and adr_o=x"02" postavi: int1 <= '0';

   Signal int1 sedaj poveži na vhod procesorja ob naslovu 0:

```
dat_i <= "00000000000"&int1 when adr_o=x"00" else pin;
```

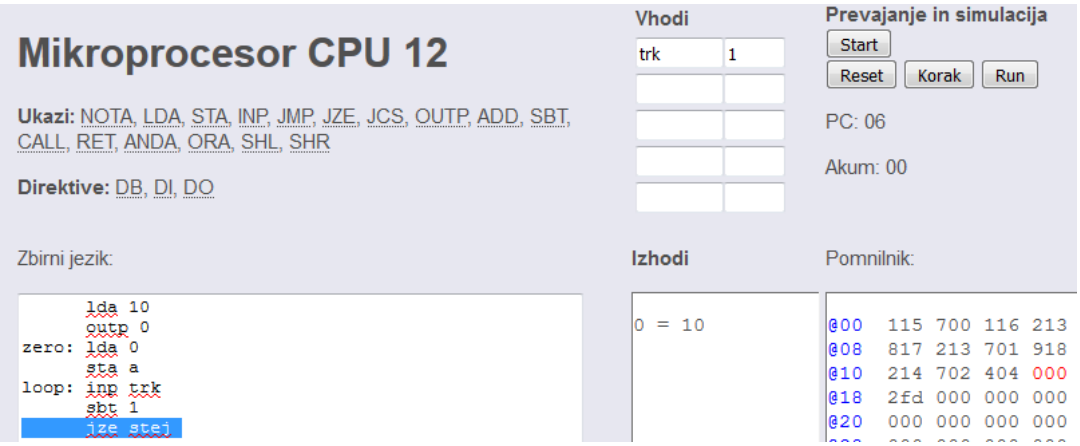
b. V sistemu poveži signal **trkz**, na vhod **trk** komponente proc. Procesorski izhod **pout2** pa poveži na zunanje **led** namesto pout1. Dopolni program, ki ob zaznavi trka poveča vrednost na izhodnih vratih 2 in preizkusi delovanje:

```
      lda 30
      outp 0
zero: lda 0
      sta a
loop: inp trk
      sbt 1
      jze stej
      lda a
      add 1
      sta a
      outp 1
      sbt 765
      jze zero
      jmp loop
stej: lda n
      add 1
      sta n
      outp 2
      jmp loop
a     db 0
n     db 0
trk   di 0
```

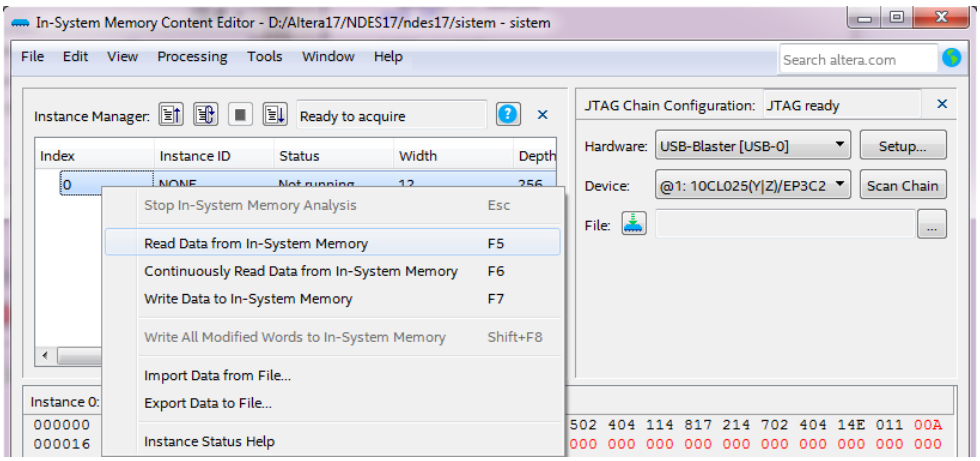
c. Spremeni logiko sistema, da bo namesto štetja trkov štel, kolikokrat je žogica zgrešila lopar. Izbriši stavek, ki spremeni smer žogice, ko pride do spodnje meje zaslona (yz = 600). Sedaj žogica ne bo prikazana v času, ko števec šteje od 600 do 4095, nato pa se spet pojavi na vertikalni koordinati 0. Pogoj, da je koordinata žogice dosegla spodnji rob zaslona (yz=600) naj bo sedaj sedaj povezan na vhod procesorja za zaznavo trka.

7.2 Programiranje in razhroščevanje

Programsko kodo mikroprocesorja razvijamo v spletnem prevajalniku cpu.html, kjer program najprej prevedemo in preizkusimo na simulaciji. Simulacijo vhodnih signalov procesorja naredimo tako, da zapišemo ime signala in vrednost v okence vhodi. Če želimo simulirati izvajanje kode, ko je signal trk na 1, nastavimo:



Prevedeno kodo, ki jo dobimo ob kliku na Start prenesemo iz okna Izhodi v datoteko **program.mif**. Če želimo spremeniti programsko kodo brez prevajanja celotnega sistema, moramo najprej v programu Quartus izbrati: Tools > In-System Memory Content Editor, ki zna spremeniti vsebino blokovnega pomnilnika prek vmesnika JTAG. V zavihku Hardware izberemo USB-Blaster in program bo avtomatsko našel instanco pomnilnika. Pomnilnik lahko beremo, urejamo ali vpišemo nove vrednosti iz datoteke **program.mif**. Vse operacije dobimo z desnim klikom na inštanco:



Mikroprocesorski sistem vsebuje tudi nekaj možnosti razhroščevanja, če povežemo spodnjo ploščo z USB/UART pretvornikom na računalnik. V terminalskem programu, npr. **Putty** nastavimo frekvenco 9600 baudov. Na voljo so ukazi:

- s - ustavi delovanje procesorja
- r - resetiraj procesor
- e - omogoči delovanje ustavljenega procesorja
- k - izvedi en ukaz (izvajanje po korakih) in
- p - izpiši vsebino programskega števca (v šestnajstiški obliki)