8. vaja: Mikroprocesor (2. del)

1 Vhodna vrata

Dodaj v model mikroprocesorja ukaz INP, ki naloži podatek iz vhodne enote v akumulator.

a) Dodaj v paket (mpack.vhdl) konstanto ukaza INP (0011).
b) Dodaj v datoteko mikro.vhd stavek, ki ob ukazu INP prenese vrednost iz vhoda **pin** v akumulator.
c) Popravi program (prog.vhdl), da bo prvi ukaz bral iz vhoda in preveri delovanje s simulacijo.

```
0=> x"3000", -- INP
1=> x"8006", -- ADD 4
2=> x"8007", -- ADD 1
3=> x"7000", -- OUTP
4=> x"4002", -- JMP 2
5=> to_unsigned(10,16),
6=> to_unsigned(4,16),
7=> to_unsigned(1,16),
```

2 Pogojni skok

Pogojni skok je ukaz, ki ob izpolnjenem pogoju izvede skok na drug del programa (podobno kot JMP), če pogoj ni izpolnjen pa se program nadaljuje z naslednjim ukazom. Pogojni skoki omogočajo izvedbo vejitev in zank v programski opremi. Učni mikroprocesor pozna dva pogojna skoka:

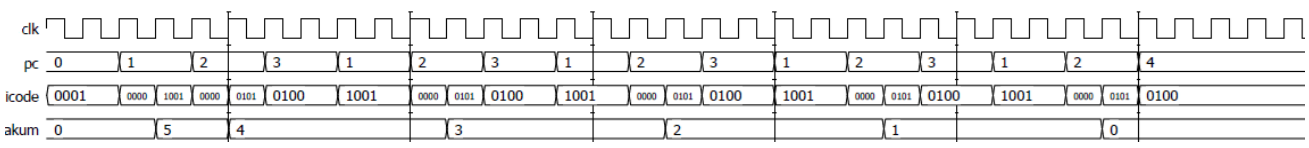
- JZE (angl. jump iz zero) izvede skok, kadar je vrednost akumulatorja enaka 0,
- JCS (angl. jump if carry set) pa izvede skok, če je rezultat zadnje računske operacije nastavil prenos.

V model mikroprocesorja (mikro.vhd) dodaj logiko za izvedbo ukaza JZE.

a) V spletnem prevajalniku za učni mikroprocesor CPE 16 napiši kratek zbirniški program, ki demonstrira delovanje ukaza JZE. Program od vrednosti spremenljivke n odšteva 1, dokler ni v akumulatorju vrednost 0.

```
      lda n
zanka: sbt 1
      jze konec
      jmp zanka
konec: jmp konec
n     db 5
```

b) Dodaj v paket konstanti za ukaz SBT (1001) in JZE (0101).
c) V opisu mikroprocesorja dodaj za izvedbo obeh novih ukazov. Odštevanje (SBT) naj bo narejeno po vzoru prištevanja (ADD), pogojni skok pa naredi po vzoru skočnega ukaza (JMP).
d) Prenesi nov program v model programskega pomnilnika in preveri delovanje s simulacijo.



3 Preizkus na FPGA

Delovanje mikroprocesorja preveri na razvojni plošči Altera DE0-nano. Uporabi vzorčni projekt (Quartus blink) v katerem naredi delilnik ure, da bomo lahko s počasno uro opazovaki delovanje procesorja. Program v procesoru naj izvaja štetje od 5 do 0 in izpisuje vrednosti na izhod.

```
start: inp 0
zanka: outp 0
       sbt 1
       jze nic
       jmp zanka
nic:   jmp start
n      db 5
```

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;

entity sistem is
 port (
   clk : in std_logic;
   led : out unsigned(7 downto 0) );
end sistem;

architecture RTL of sistem is
 signal div : unsigned(24 downto 0) := to_unsigned(0,25);
 signal clk2: std_logic := '0';
 signal pin : unsigned(15 downto 0) := x"0000";
 signal pout : unsigned(15 downto 0);
begin

 process(clk)
 begin
  if rising_edge(clk) then
   if div=2000000 then
         div <= (others=>'0');
         clk2 <= not clk2;
       else
         div <= div + 1;
       end if;
  end if;
 end process;

u1: entity work.mikro port map (
  clk => clk2,
  pin => pin,
  pout => pout );

 led <= pout(7 downto 0);

end RTL;
```