



Krmilni del skrbi za prehajanje stanj ter naslovne in krmilne signale za pomnilnik. Predstavili ga bomo v obliki algoritma, ki se veji glede na stanje:

- če je  $st=zajemi$ , bo v naslednjem ciklu:
  - $st \leq izvedi$
  - $pc \leq adr+1$ ;  $adr \leq datain(7 \text{ downto } 0)$
  - če je na  $datain$  ukaz  $jmp$ :
    - $st \leq zajemi$
  - če je na  $datain$  ukaz  $lda$ :  $rd \leq '1'$
  - če je na  $datain$  ukaz  $sta$ :  $wr \leq '1'$
- če je  $st=izvedi$ , bo v naslednjem ciklu:
  - $adr \leq pc$

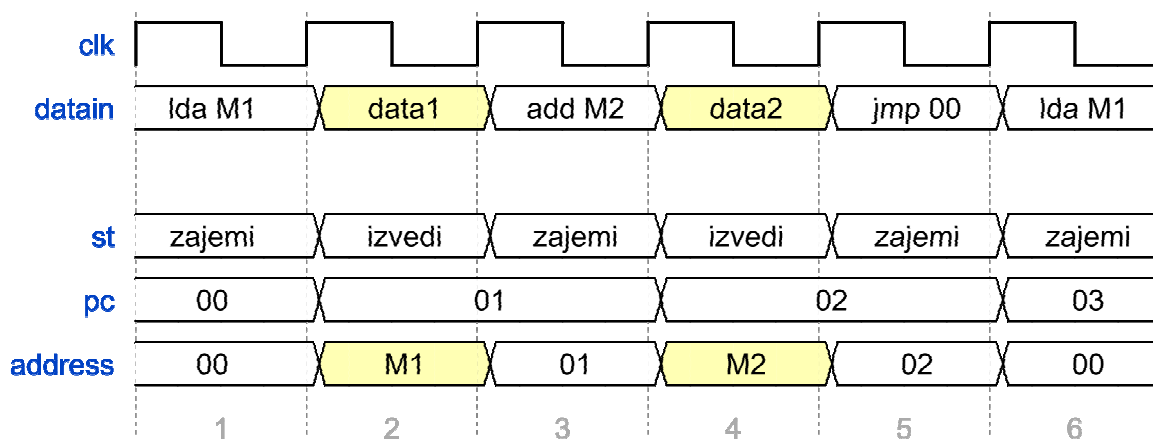
- a) Naredi VHDL model procesorja CPU12 in preveri sintakso. Simbolične konstante ukazov naj bodo definirane v paketu `procpak` knjižnice **work**, ki ga vključi v opis vezja.

Pomnilnik bomo naredili v obliki statičnega pomnilnika z ločenim podatkovnim vhodom in izhodom ter sinhronim vpisom, ki je na voljo v programirljivih vezjih. Programsko kodo bomo pisali z uporabo konstant iz paketa `procpak`, zato je inicializacija pomnilnika podobna pravi zbirniški kodi:

```
type memory is array(0 to 4) of unsigned(11 downto 0);
```

```
signal m : memory := (
  lda & x"04",
  add & x"04",
  sta & x"04",
  jmp & x"00",
  x"005"
);
```

Poglejmo si potek izvajanja kratkega programa s tremi ukazi: najprej naloži v akumulator vrednost ( $data1$ ) iz pomnilniške lokacije M1, nato prišteje vrednost ( $data2$ ) iz lokacije M2, tretji ukaz pa je brezpogojni skok na začetek programa (naslov 00):



- b) Naredi model statičnega pomnilnika v katerem bo programska koda za procesor. Uporabi sinhroni proces za vpis podatka iz vhoda  $din$  ob fronti ure in pogoju  $wr='1'$ . Branje podatkov naj bo asinhrono, na  $dout$  se pojavi vrednost, ki jo določa naslov  $adr$ . Naredi testno strukturo, ki naj povezuje procesor in pomnilnik ter določa časovni potek ure in signala  $rst$ . S testno strukturo preizkusi delovanje procesorja na simulatorju.