



Laboratorij za načrtovanje integriranih vezij

Univerza *v Ljubljani*
Fakulteta *za elektrotehniko*



Digitalni Elektronski Sistemi

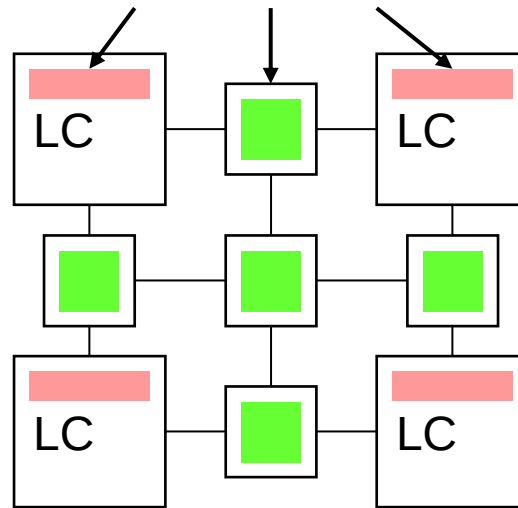
Programirljiva vezja in mikro-programiranje

mikroprogramirana vezja

Programirljiva vezja / mikroprocesorji

▶ programirljivo vezje

- ▶ programski biti določajo strukturo (konfiguracijo)



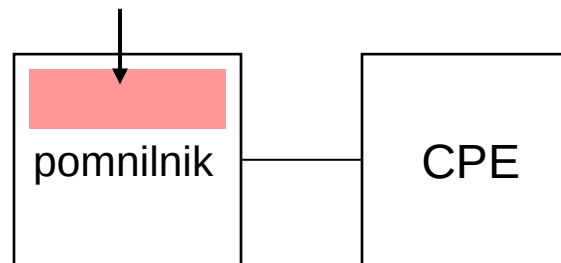
delovanje logičnih celic (LC)

povezave znotraj LC

povezave v povezovalni mreži

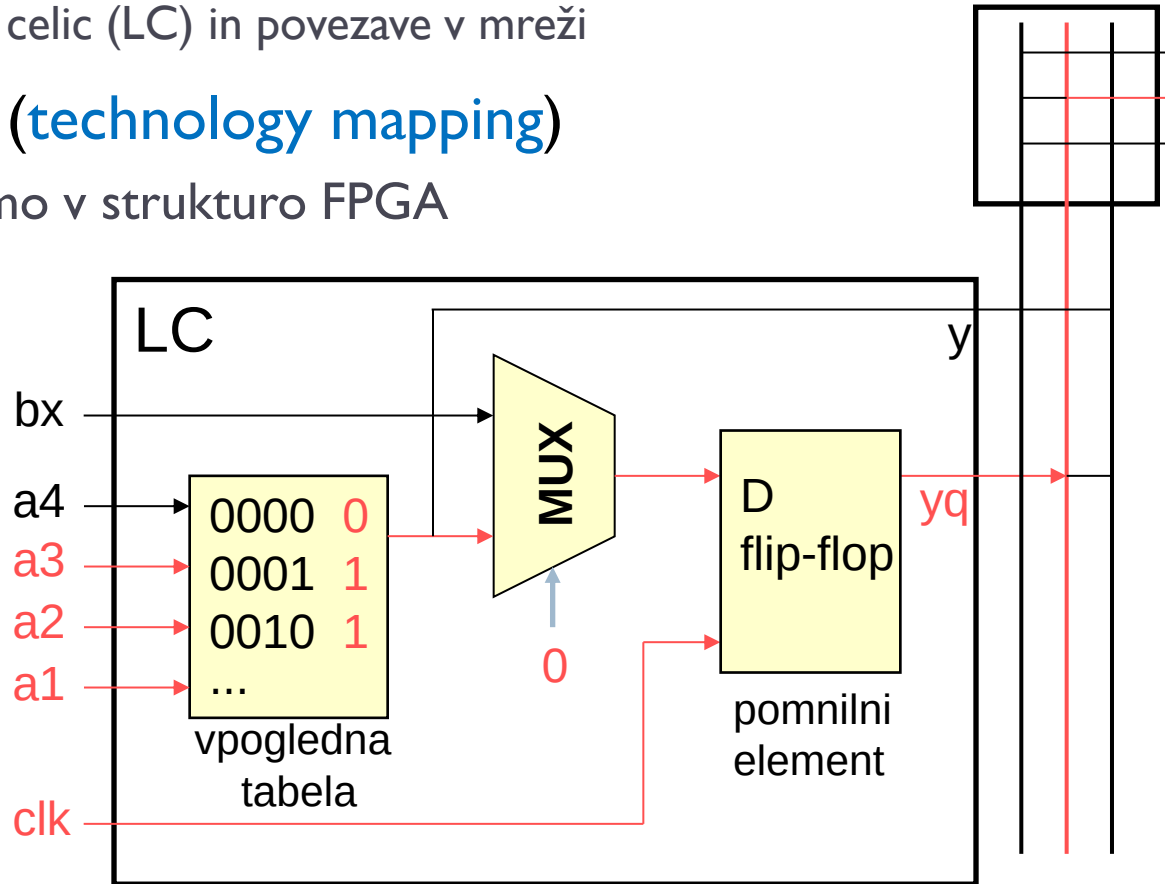
▶ mikroprocesor

- ▶ program določa ukaze, ki jih CPE izvaja



Podrobnejša zgradba FPGA

- ▶ pri programiranju (**configuration**) vpišemo vrednosti v konfiguracijski RAM
 - ▶ določa delovanje logičnih celic (LC) in povezave v mreži
- ▶ tehnološka preslikava (**technology mapping**)
 - ▶ digitalno vezje preslikamo v strukturo FPGA
- ▶ logična celica
 - ▶ LUT, MUX, DFF
 - ▶ prenosna logika
 - ▶ dodatne funkcije



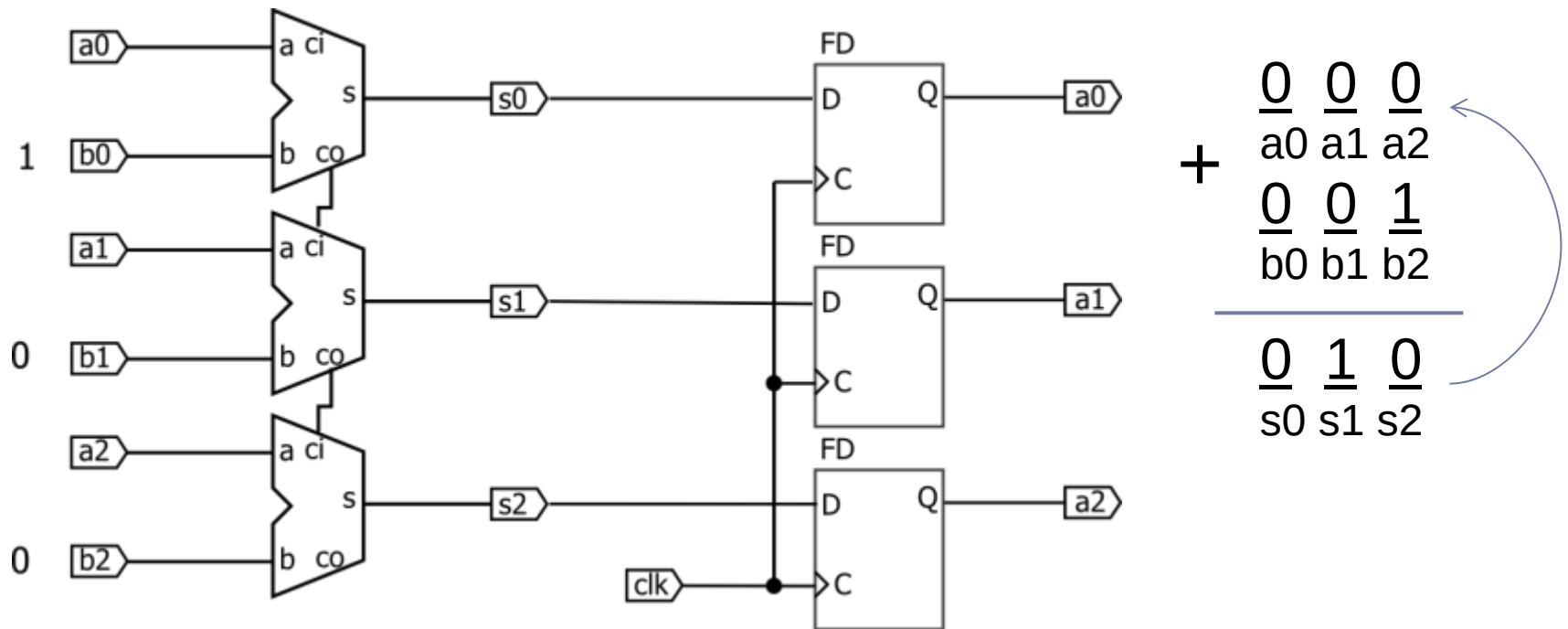
The screenshot displays the Xilinx Floorplanner interface for the VideoCard4.fnc design. The top window shows the hierarchy tree, listing components like VideoCard4 "Primitives", grab "Grabber", tw "Twice", int "int", CarryChain_5 "CarryChain", z2c "z2c_top", and RAM_Buff "Buff". The main window shows the placement grid for the xc3c50-4-tq144 device, with components placed on a grid. The bottom window shows the component list, including RAM_Buff/N145, RAM_Buff/N10, RAM_Buff/N1, and various VideoCard4 components.

- rezultat avtomatične preslikave
- načrtovanje vezja poteka na višjem nivoju

xc3s50 tq144
500 tabel (LUT)
280 D flip-flopov
8 kB DPRAM
~ 270k vrat ASIC

Sekvenčno vezje s povratno vezavo

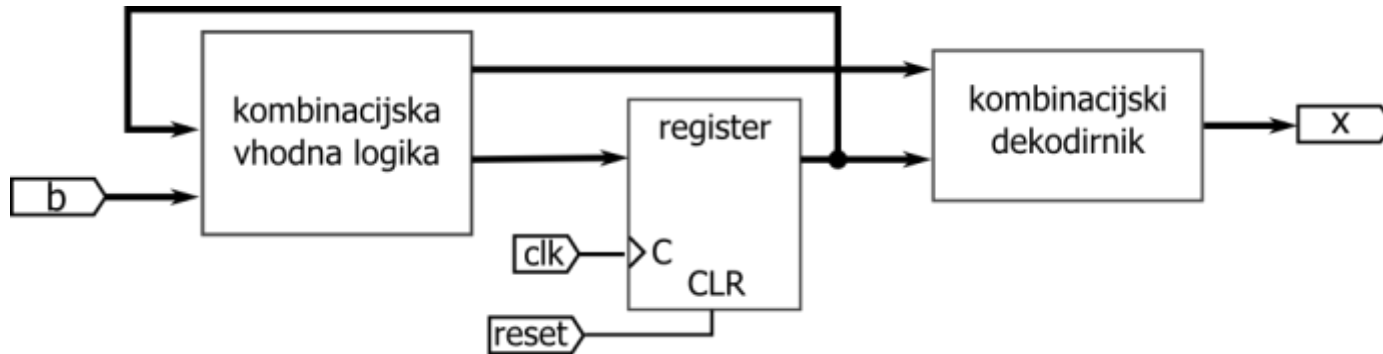
- ▶ 3-bitni seštevalnik in 3-bitni register = 3-bitni števec
 - ▶ vezje zasede 3 logične celice



- ▶ Sekvenčno vezje ima $2^3 = 8$ stanj (000 do 111)

Sekvenčno vezje: sekvenčni stroj – avtomat

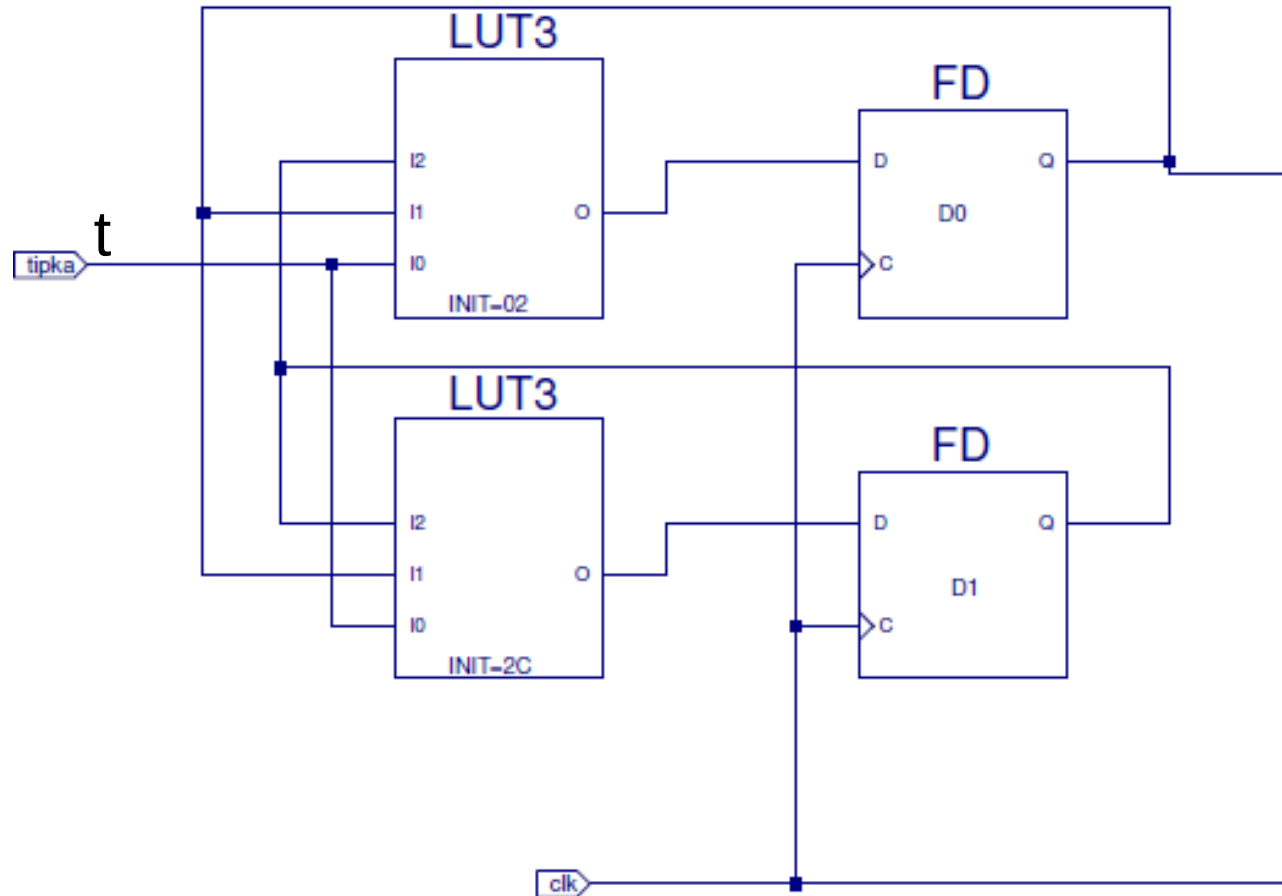
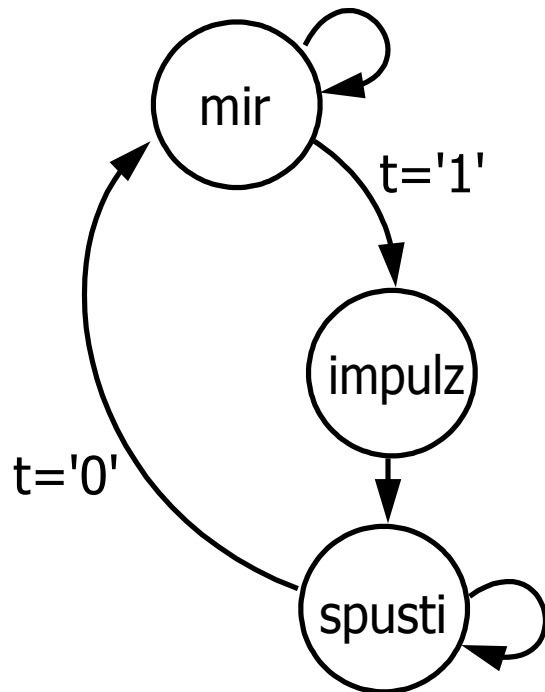
- ▶ Sekvenčni stroj vsebuje n -bitni register za 2^n stanj



- ▶ Vhodna logika je v vezju FPGA narejena s tabelami (LUT)
 - ▶ s spremembo vsebine tabel spremenimo delovanje vezja !

Sekvenčni stroj z diagramom stanj

- Vsebina vpoglednih tabel (LUT) določa prehajanje stanj

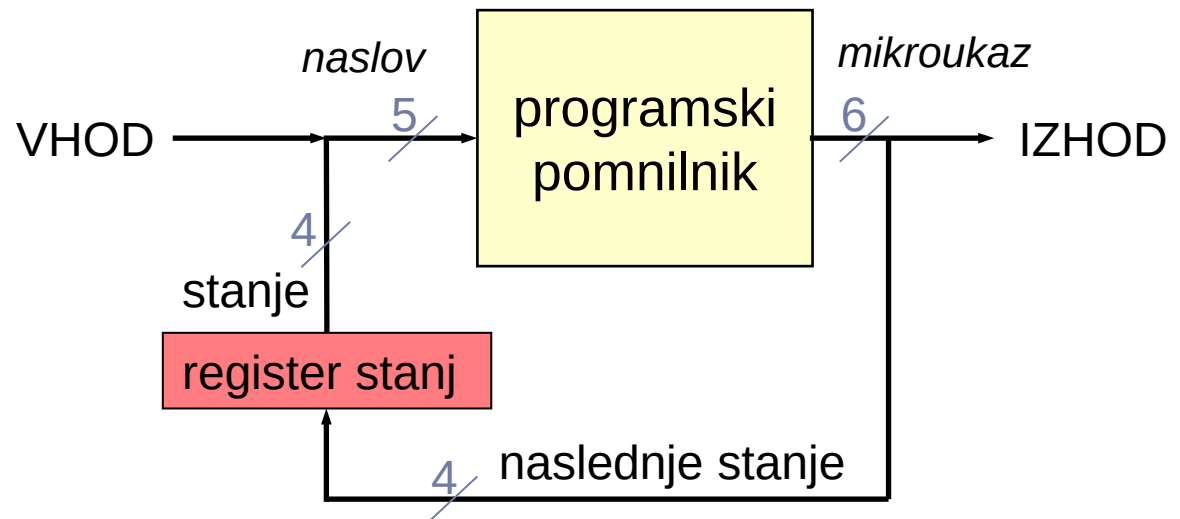


Mikro-programirano sekvenčno vezje

- ▶ Sekvenčni stroj narejen s tabelami je mikroprogramiran - **mikrosekvenčnik**

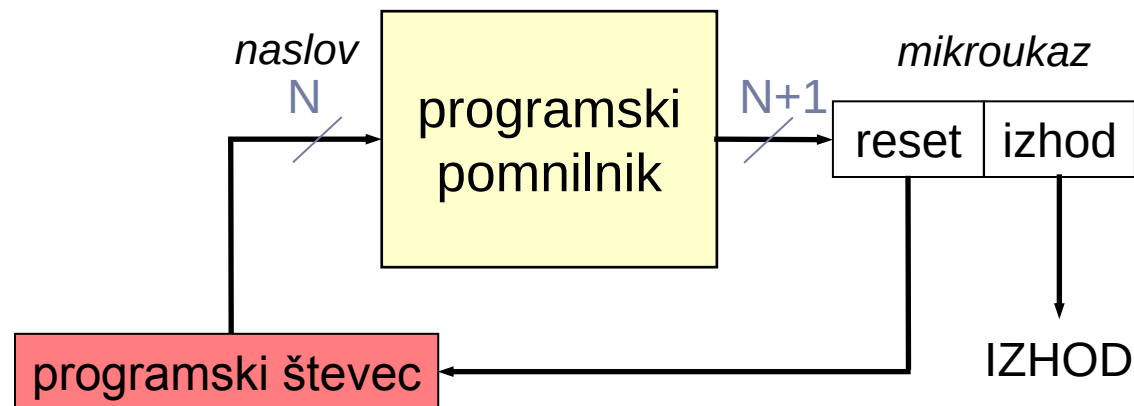
- ▶ *mikroprogramirano* vezje ima krmilne vrednosti shranjene v pomnilniku vrste ROM ali RAM

- ▶ *mikroukazi* so besede v pomnilniku
- ▶ *mikroprogram* je zaporedje *mikroukazov*
- ▶ v *programski pomnilnik* vrste RAM lahko vpisujemo *mikroukaze*



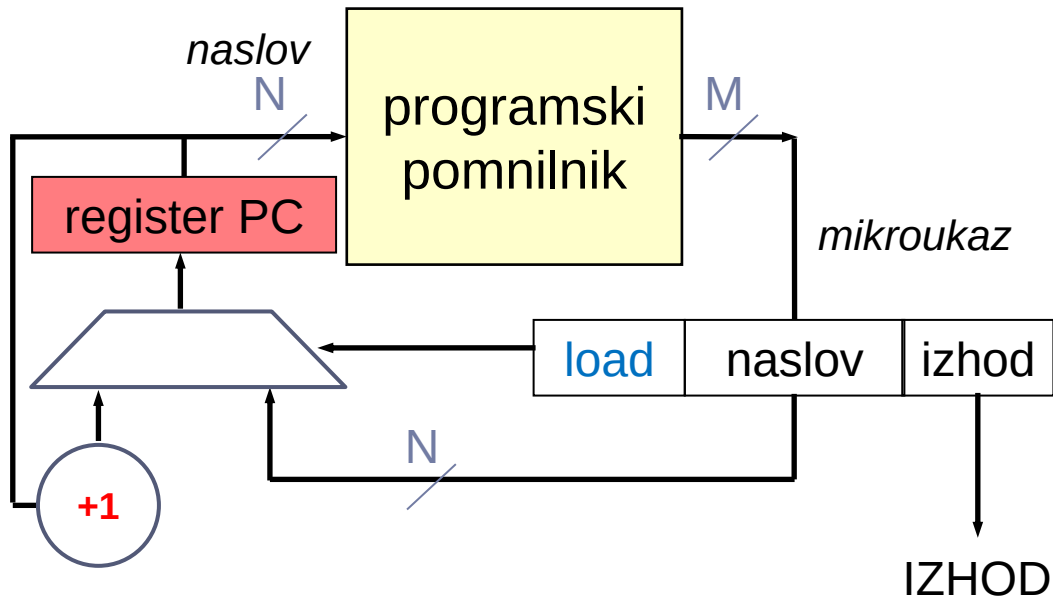
Mikrosekvenčenik s programskim števcem

- ▶ Register nadomestimo s števcem, ki avtomatsko povečuje naslove
- ▶ izvedba diagramov z enim samim zaporedjem (sekvenco) stanj
- ▶ zaporedje se izvaja ob uri (ni krmilnih vhodov)
- ▶ npr. štetje po modulu



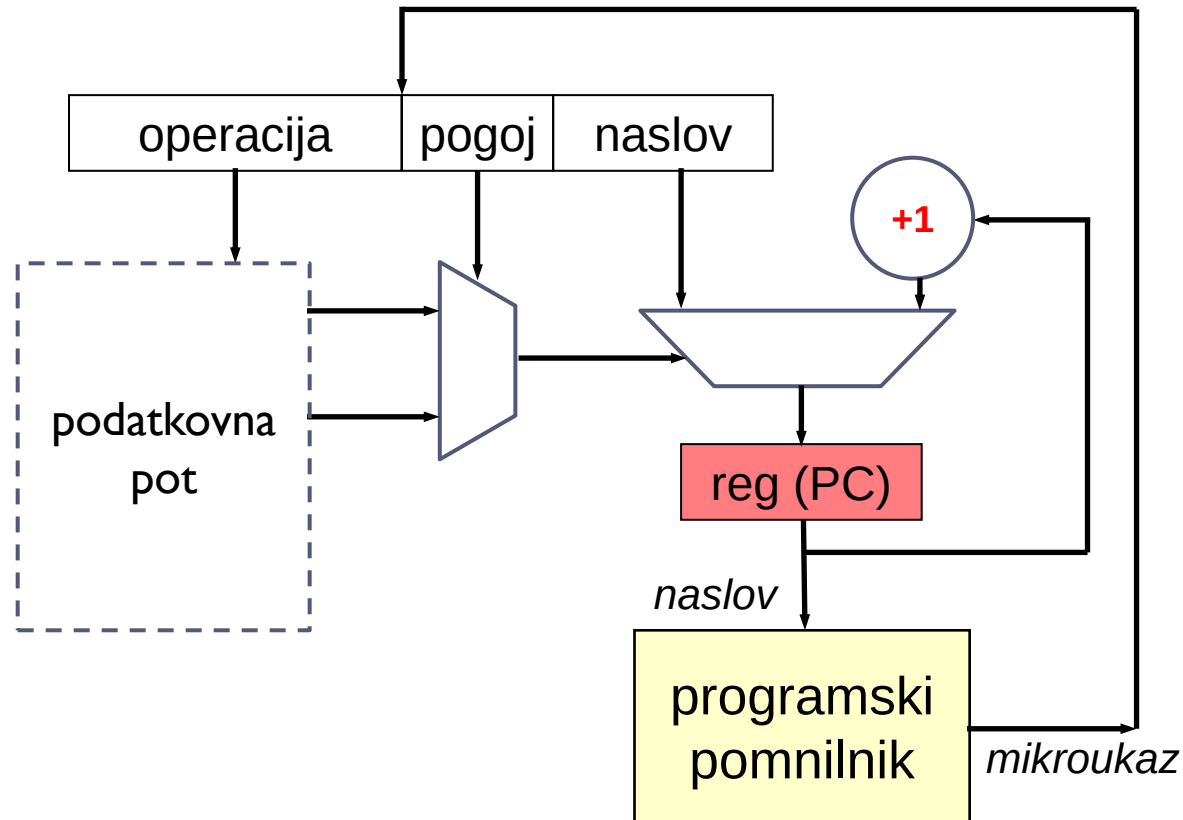
Mikrosekvenčnik s skočnimi ukazi

- ▶ Programski števec (**PC**) naloži nov naslov (**load**), kar omogoča izvedbo skokov
- ▶ algoritmi, ki nimajo vseh mikroukazov v zaporedju



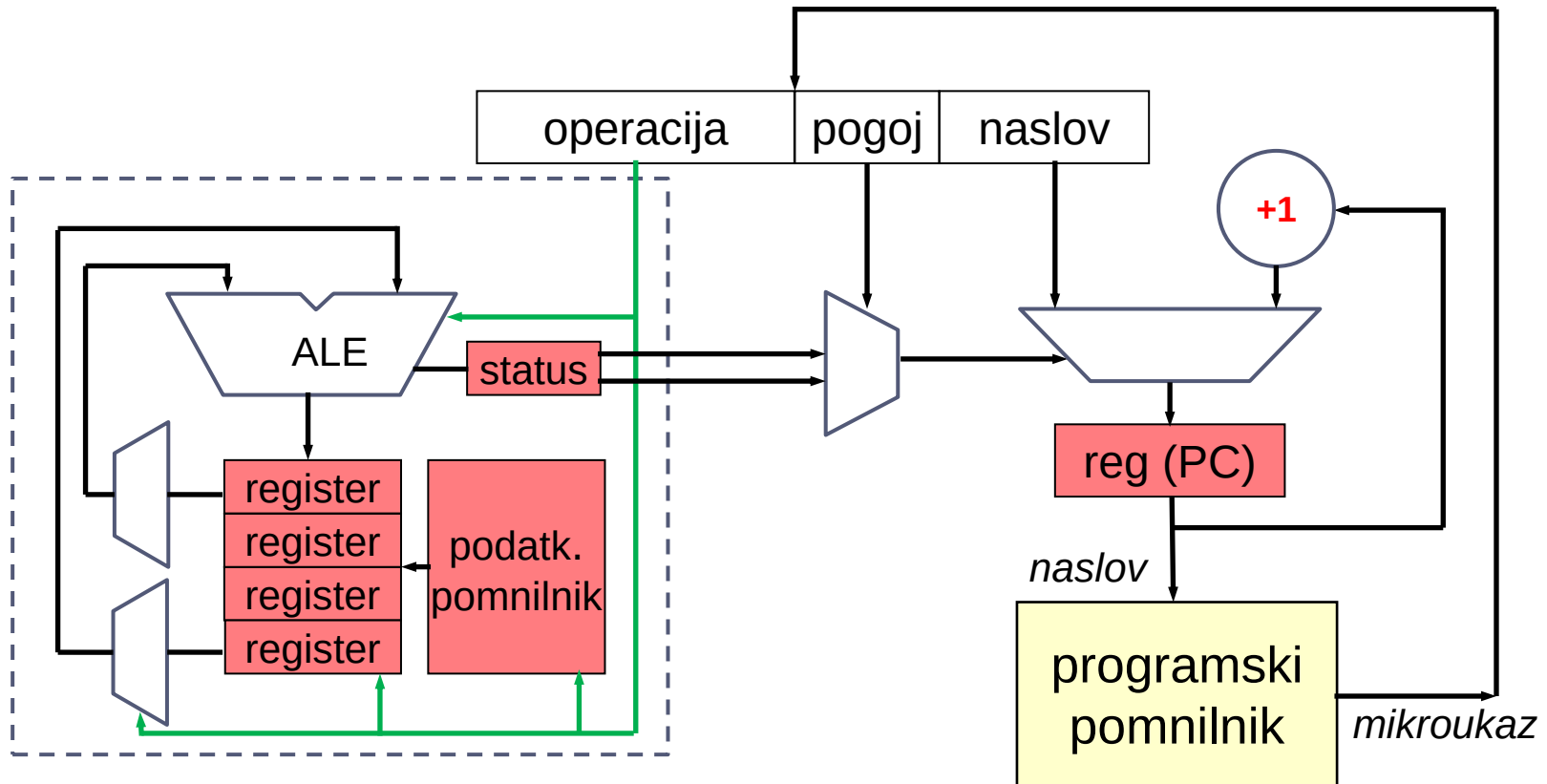
Mikrosekvenčni in podatkovna pot

► Operacije krmilijo podatkovno pot



- Podatkovna pot izvaja registrske (mikro)operacije
 - registri, ALE, podatkovni pomnilnik

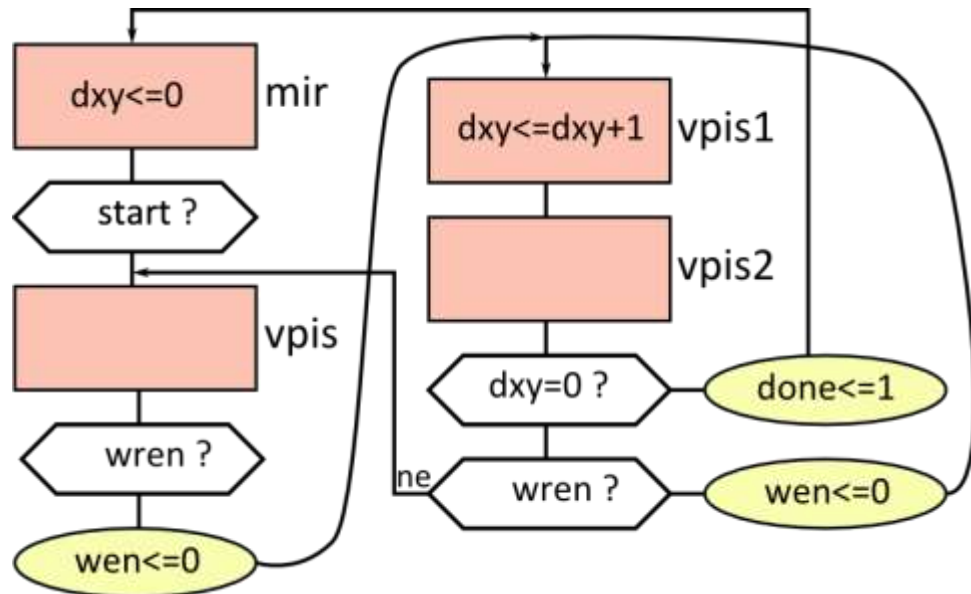
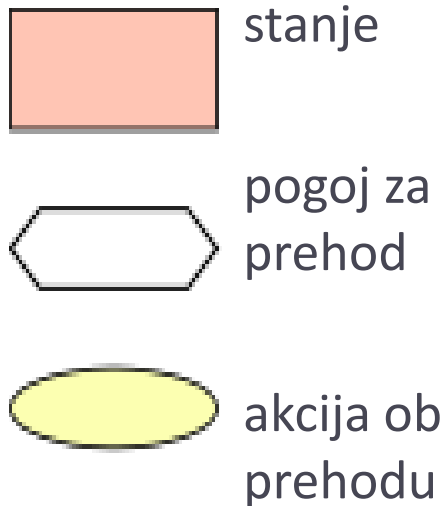
Mikroprogramirana krmilna (procesna) enota



Načrtovanje krmilne enote z ASM

- ▶ ASM = **A**lgoritmic **S**tate **M**achine
- ▶ Opišemo prehajanje stanj in podatkovne operacije
- ▶ Npr. krmilna enota za risanje kvadrata 8x8 (Pixel)

- ▶ Simboli **ASM**:



Načrtovanje mikroprogramirane enote

- ▶ Stanje določa programski števec
- ▶ Števec dxy je v registru A na podatkovni poti
- ▶ Določimo obliko mikroukazov:

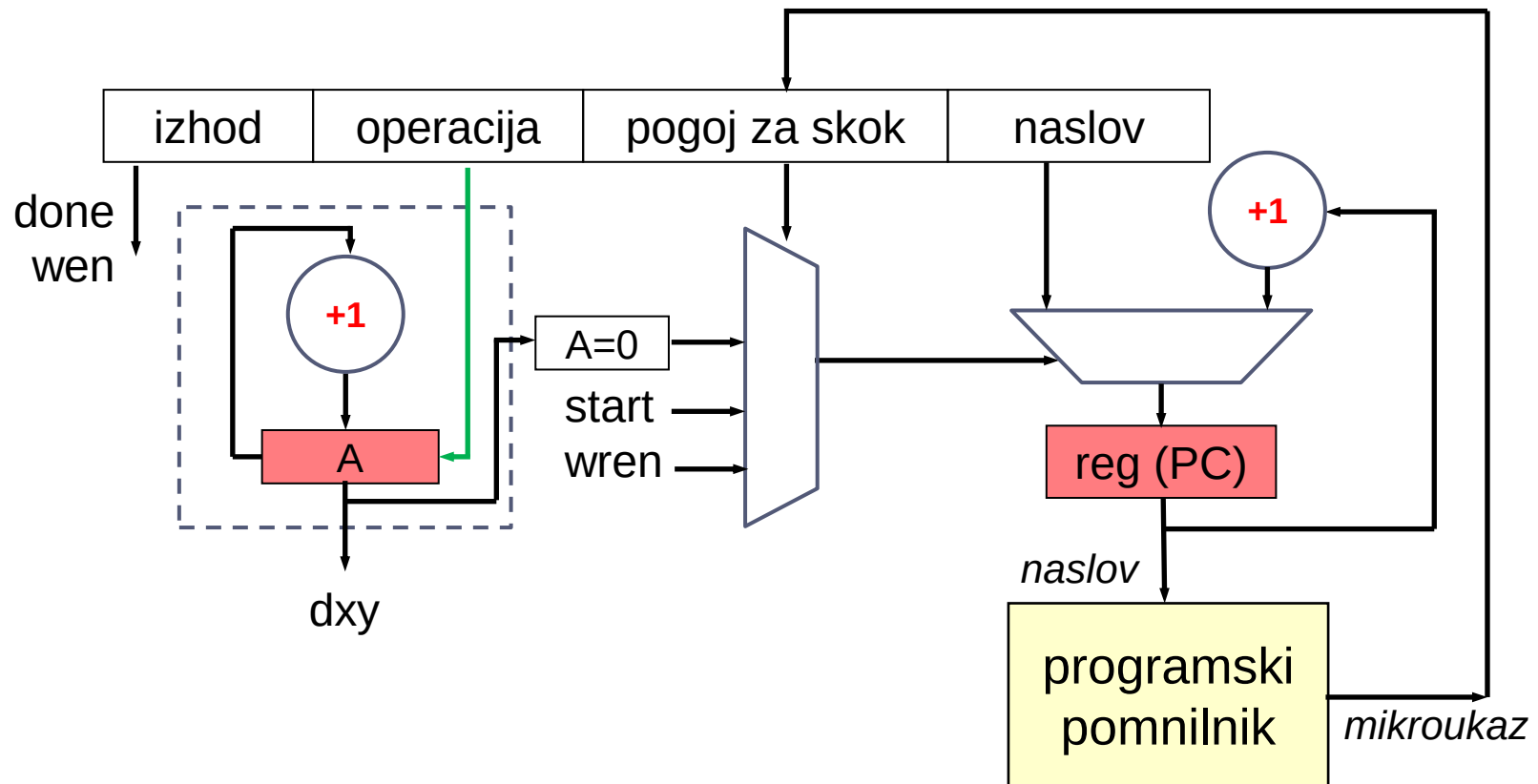
izhod	operacija	pogoj za skok	naslov
-------	-----------	---------------	--------

done, wen

- ▶ 00, ni operacije
- ▶ 01, $A \leq 0$
- ▶ 10, $A \leq A+1$
- ▶ 1000, brezpogojni skok
- ▶ 0100, skok ob $A=0$ (zero)
- ▶ 0010, skok ob $wren=0$
- ▶ 0001, skok ob $start=0$

Shema mikroprogramirane enote

- ▶ Podatkovna pot vsebuje register A (dxy)
 - ▶ operaciji: $A \leq 0$, $A \leq A+1$
- ▶ Izhod: done, wen



Kodiranje mikroprogramirane enote

► deklaracija pomnilnika

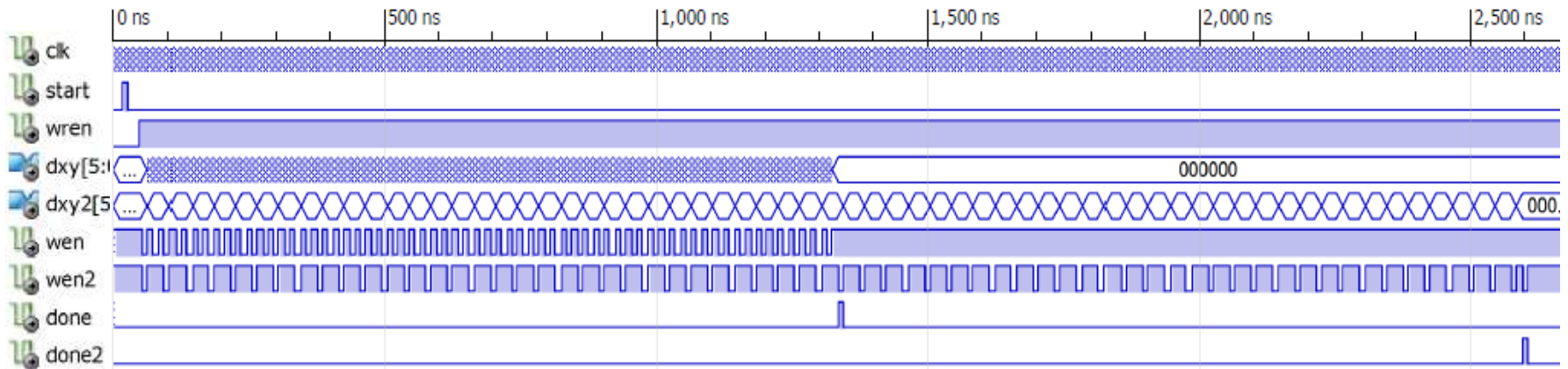
```
type mem is array(0 to 7) of std_logic_vector(10 downto 0);  
signal m: mem := (  
    B"01_01_0001_000", -- (0) a<=0, jmp 0 if start=0  
    B"01_00_0010_001", -- (1) jmp 1 if wren=0  
    B"00_10_0000_000", -- (2) a<=a+1, wen<=0  
    B"01_00_0100_110", -- (3) jmp 6 if a=0 (zero)  
    B"01_00_0010_001", -- (4) jmp 1 if wren=0  
    B"01_00_1000_010", -- (5) jmp 2  
    B"10_00_1000_000", -- (6) jmp 0, done<=1
```

► opis vezja

```
if rising_edge(clk) then  
    if inst(8 downto 7)="01" then a <= "000000";  
    elsif inst(8 downto 7)="10" then a <= a + 1;  
    end if;  
  
    if inst(6 downto 3)="1000" or  
        (inst(6 downto 3)="0100" and zero='1') or ... then  
        pc <= unsigned(inst(2 downto 0));  
    else  
        pc <= pc + 1;
```


Primerjava obeh izvedb

- ▶ Primerjava časovnih potekov na simulaciji
 - ▶ ASM cca. 2x hitrejši od procesne enote



- ▶ Kodiranje v jeziku VHDL
 - ▶ cca. 90 vrstic kode za opis obeh vezij
 - ▶ 2 notranja signala (ASM) in 6 signalov za procesor
- ▶ Velikost vezja
 - ▶ ASM zasede 25 LUT in 12 DFF
 - ▶ Procesor zasede 19 LUT in 9 FF