

3. Programiranje mikropcesorjev

- Predstavitev podatkov v računalniku in njihovo kodiranje
- Programski model
- Sklad v mikropcesorju
- Zbirni jezik
- Višji programski jeziki
- Orodja in pripomočki za razvoj programov

Predstavitev podatkov v računalniku

- Cela števila: nepredznačena in predznačena

0	1	0	1	0	0	1	1
---	---	---	---	---	---	---	---

= 83 nepredznačeno

= 83 predznačeno

1	1	0	1	0	0	1	1
---	---	---	---	---	---	---	---

= 211 nepredznačeno

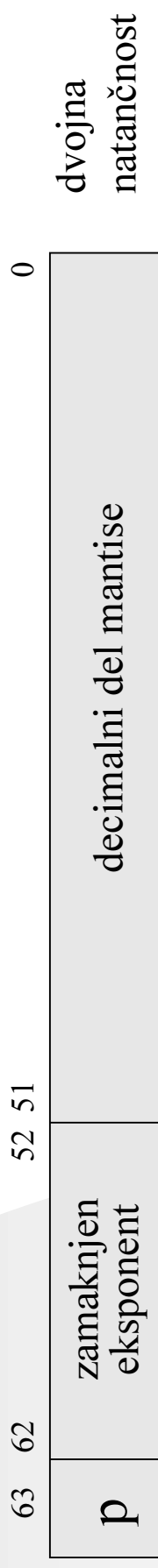
= -45 predznačeno

Enostavno seštevanje in odštevanje, kompleksnejše množenje in deljenje.

■ Realna števila (IEEE)



$$r = (-1)^p \times 2^{(e-127)} \times 1.d \quad 0 \approx 1.2 \times 10^{-38} - 3.4 \times 10^{38}$$



$$r = (-1)^p \times 2^{(e-1023)} \times 1.d \quad 0 \approx 2.2 \times 10^{-308} - 1.8 \times 10^{307}$$

$$\pm\infty : e = 111...1 \quad d = 0$$

$$\text{NaN} : e = 111...1 \quad d \neq 0$$

Potrebna posebna logična vezja za vse vrste računskih operacij.

- BCD (Binary Coded Decimal) števila

	7			4			9			2	nepakirano
7	4		9	2							pakirano

Kompleksna izvedba operacij.

- Znaki

0	1	0	1	0	0	1	1	= znak S
---	---	---	---	---	---	---	---	----------

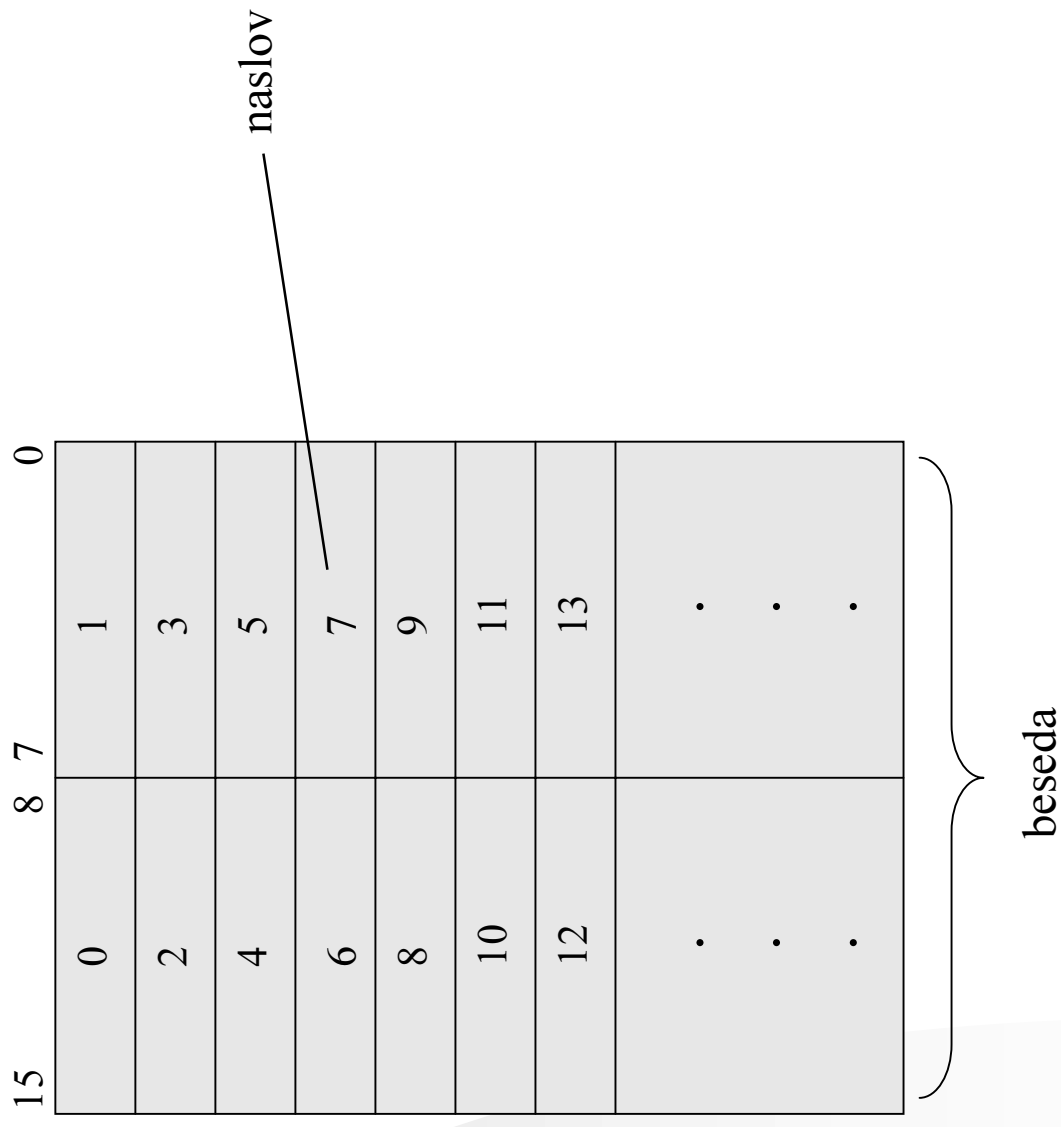
- Ostali tipi

- ◆ ura
- ◆ datum
- ◆ ...

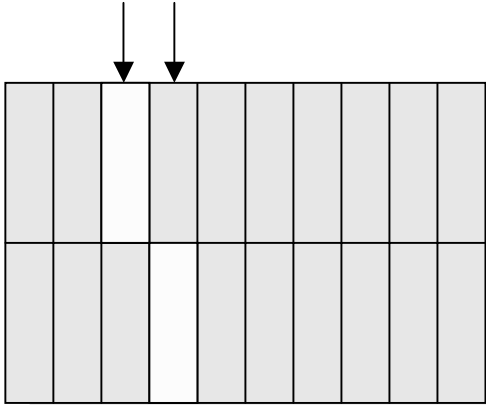
Programski model

- Registri
 - ◆ Splošno namenski registri
 - ✦ Podatkovni registri
 - ✦ Naslovni registri
 - ◆ Programski števec
 - ◆ Kazalec sklada
 - ◆ Statusni register (zastavice)
 - ◆ Bazni register, indeksni registri, segmentni registri,...

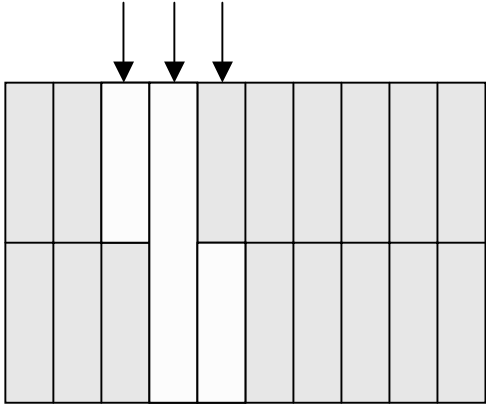
■ Organizacija pomnilnika



Problem poravnave naslovov operandov



Branje/zapis besede



Branje/zapis dvojne besede

■ Nabor ukazov

◆ Ukazi za premikanje

MOV

◆ Aritmetični ukazi

ADD, SUB, CMP, MUL, DIV, CLR, NEG

◆ Logični ukazi

AND, OR, XOR, NOT

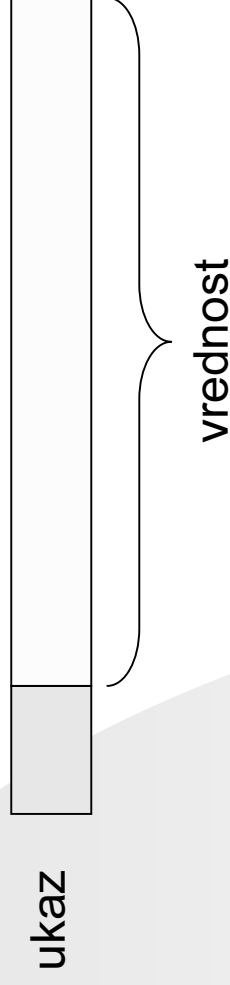
◆ Ukazi pomikanja in rotiranja

ROL, ROR, SHL, SHR, ASL, ASR

- ◆ Ukazi za manipulacijo z biti
BSET, BCLR, BCHG, BTST
- ◆ Ukazi za nadzor poteka programa
 - ◆ brezpogojni in pogojni skoki
JMP, JZ, JNZ, JMI, JPL, JC, JNC, ...
 - ◆ podprogrami
CALL, RET
- ◆ Sistemski ukazi
NOP, STOP, TRAP, ...

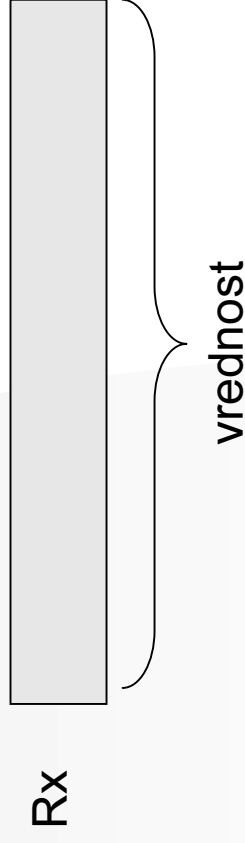
N načini naslavljanja operandov

- takojšnje



MOV #5,R0

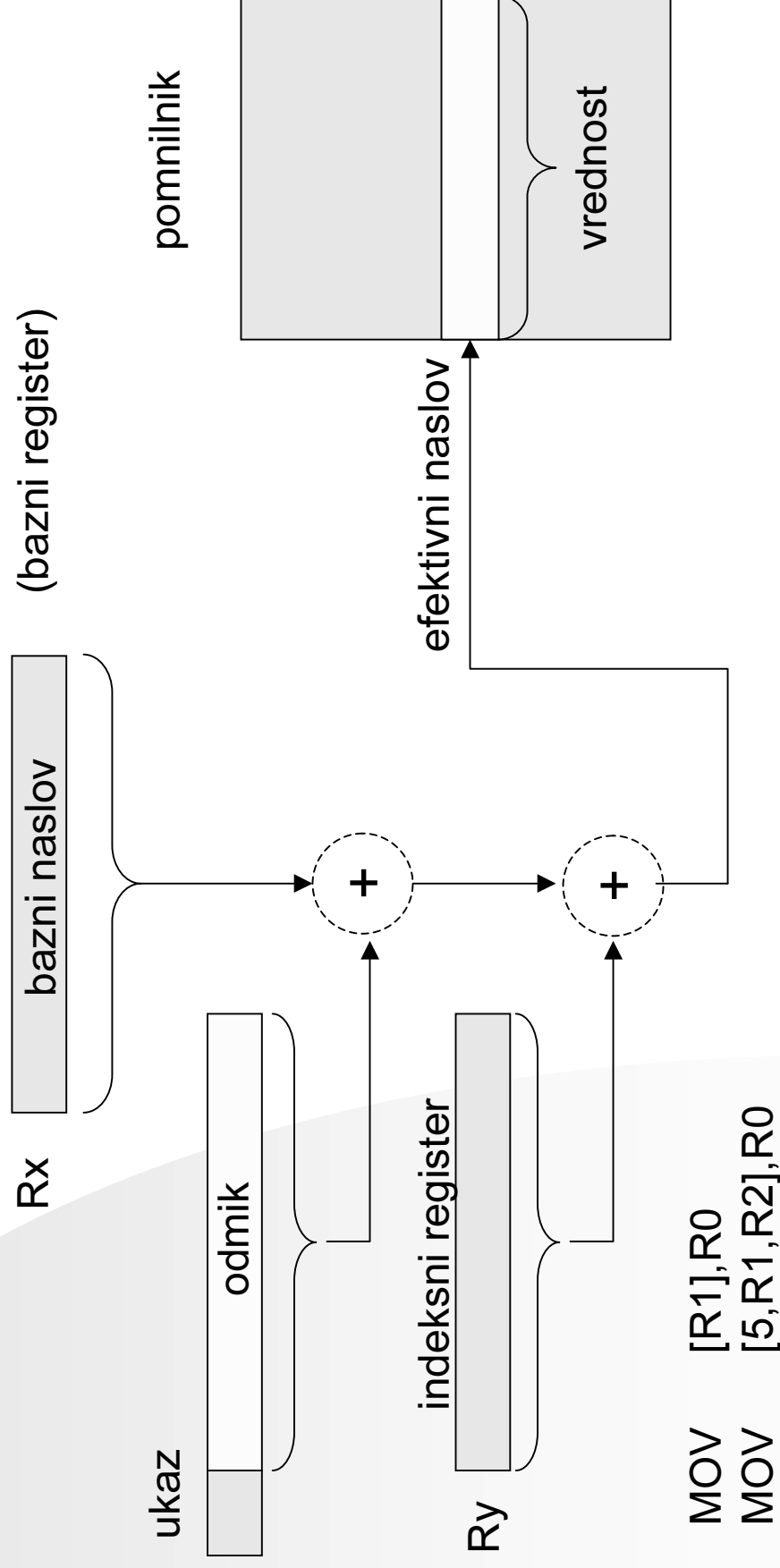
- registersko neposredno



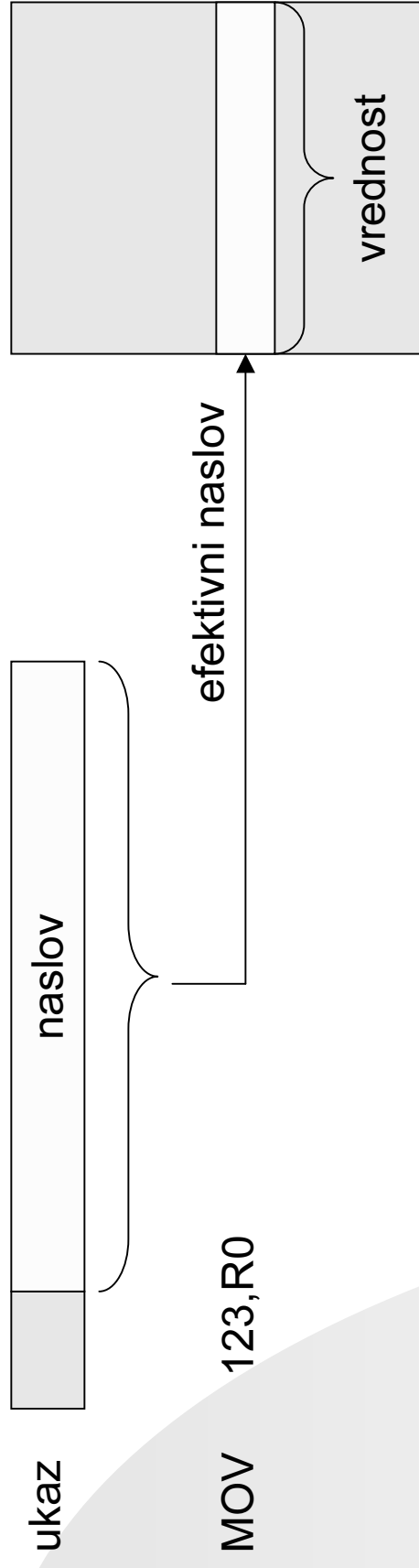
MOV R1,R0

■ registersko posredno

- ◆ brez odmika
- ◆ z odkmikom
- ◆ z odkmikom in indeksom



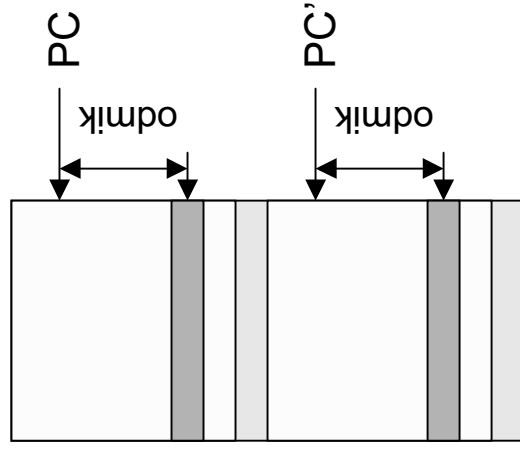
■ absolutno



■ relativno = registersko posredno glede na PŠ

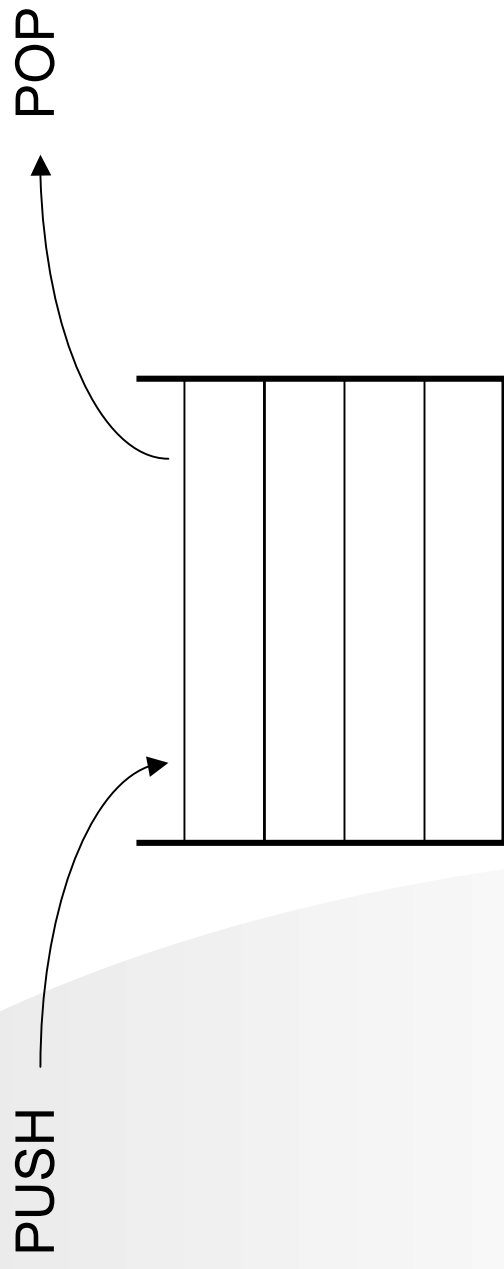
◆ relokatibilna koda

MOV [5,PC],R0

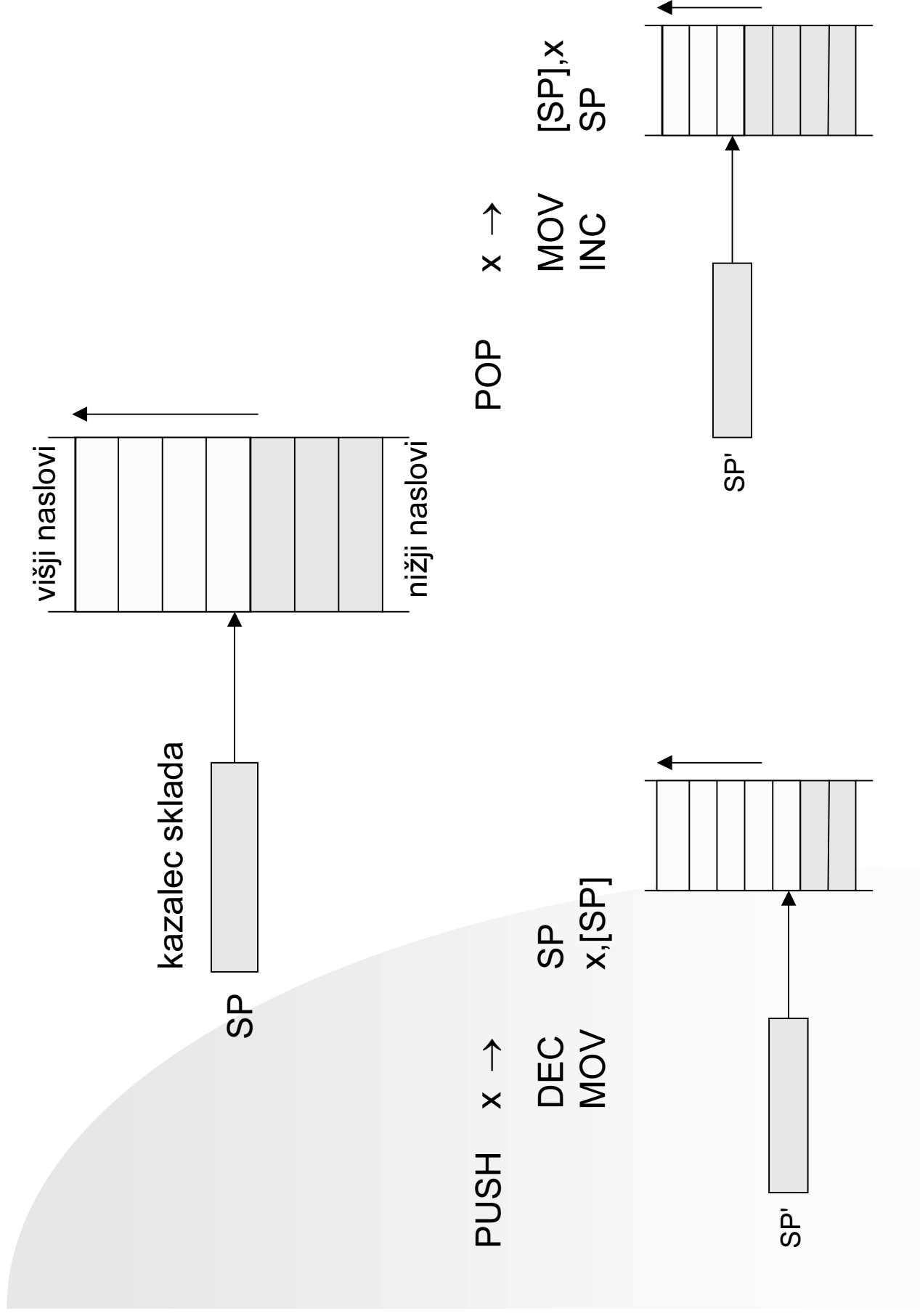


Sklad v mikrop procesorju

- Podatkovna struktura sklad (LIFO)



■ Implementacija sklada pri mikroprocesorjih



■ Uporaba sklada pri podprogramih

100 CALL A

...

200 A: CALL B

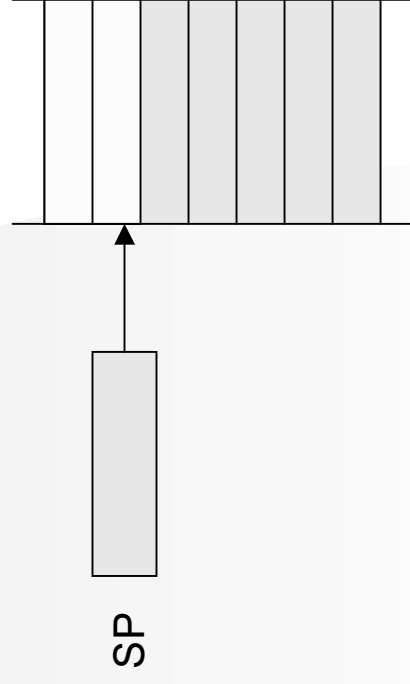
...

RET

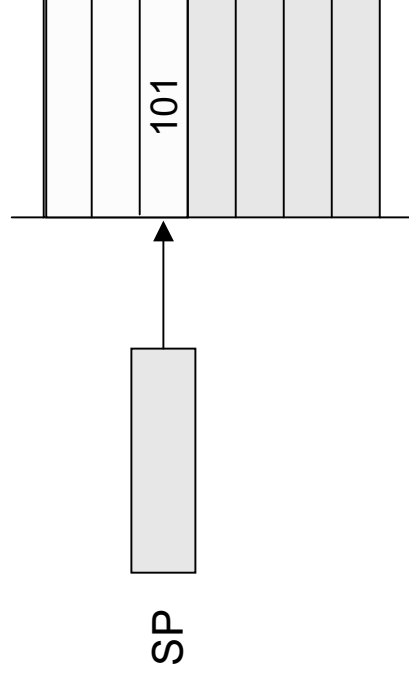
300 B:

...
RET

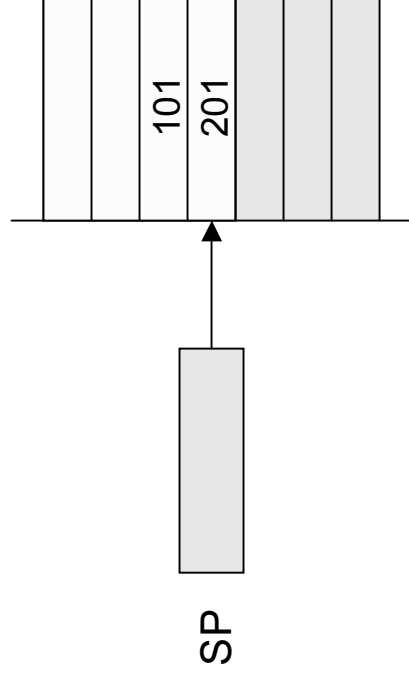
- pred izvedbo CALL A



- po izvedbi CALL A



- po izvedbi CALL B



CALL x →

PUSH PC+1

JMP x

RET →

POP PC

- Sklad kot začasna shramba

```
PUSH x
PUSH y
...
POP y
POP x
```

- Uporaba sklada za prenos parametrov

```
PUSH P2           ; prenos parametrov na sklad
PUSH P1           ; uporaba lokalnih spremenljivk
CALL PROC         ; klic
ADD #2,SP         ; sprostitvev prostora
```

```
PROC: MOV [1,SP],R0 ; dostop do prvega parametra
```

- Uporaba sklada za lokalne spremenljivke

```
SUB #s,SP          ; rezervacija prostora
MOV #0,[4,SP]      ; uporaba lokalnih spremenljivk
...
ADD #s,SP          ; sprostitvev prostora
```

Zbirni jezik

- Splošna oblika ukazov v zbirniku

labela ukaz operandi komentar

- Izvedljivi ukazi - ukazi mikroprocesorja
- Zbirniške smernice
 - ◆ določitev naslova programa - ORG
 - ◆ simbolične konstante - EQU

petka	equ	5
	org	100

◆ Pogojno prevajanje

if, ifdef, ifndef, while

◆ Makroji

labela macro arg1,arg2,...

....

endm

poslji macro znak

if debug == 1 ; preveri verzijo

...

else

...

endif

endm

poslji 'A'

poslji 'B'

poslji 'C'

- ◆ Višje nivojski ukazi

 - pagesel labela

 - banksel labela

- ◆ Smernice za tiskanje

 - list, nolist, page, space, title

- ◆ Ostalo

 - #include ime datoteke

 - PROCESSOR tip procesorja

 - #define simbol niz_znakov

 - #undefine

Višji programski jeziki

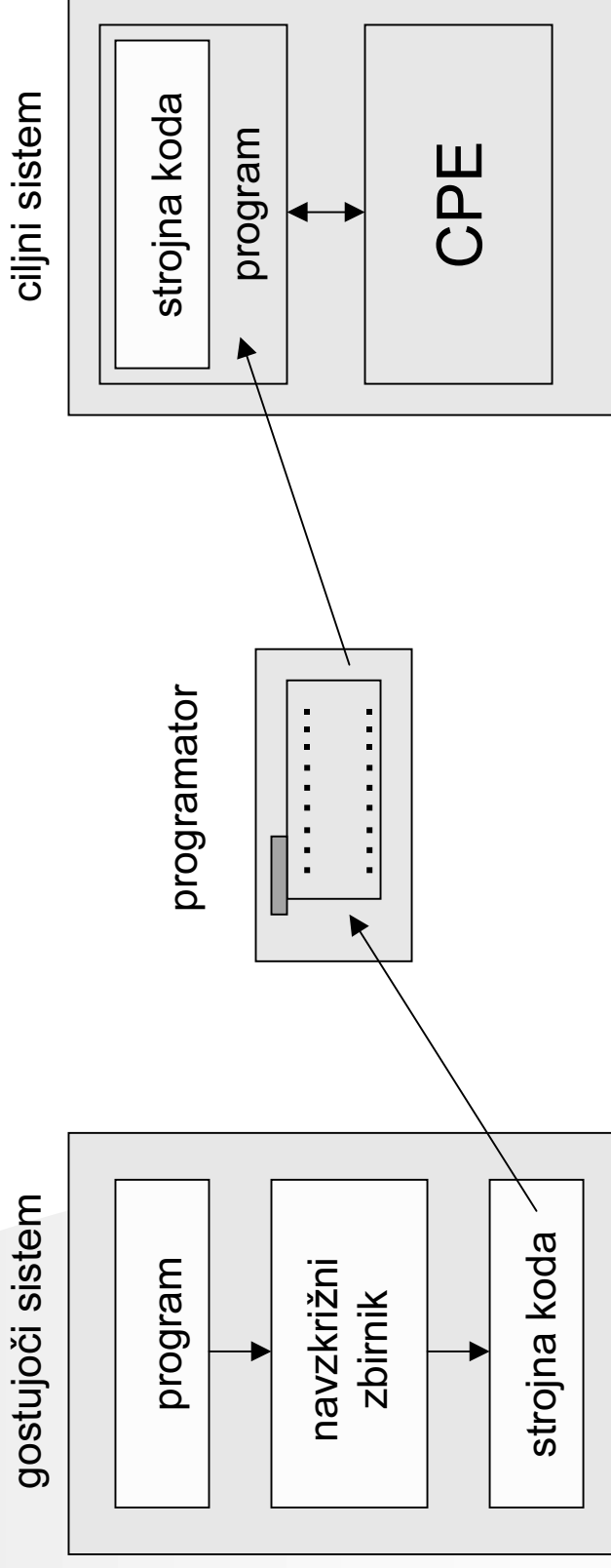
- večja produktivnost
- manj napak
- lažje vzdrževanje
- večja (in manj optimalna) koda

```
function Max (A,B:Integer):Integer;  
begin  
  if A>B then  
    Max := A  
  else  
    Max := B;  
end;
```

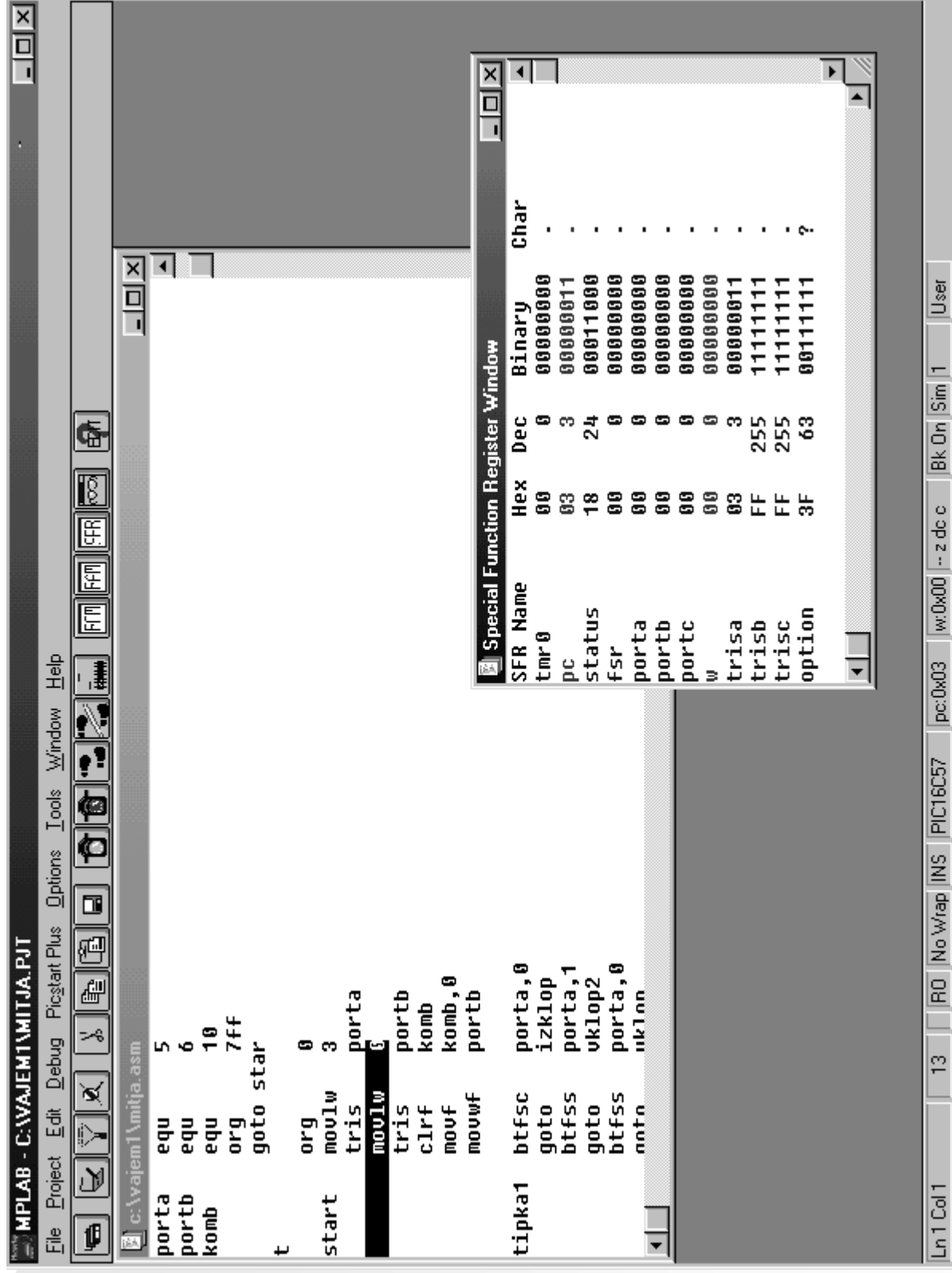
```
Unit1.pas.27: begin  
00442EBC 55      push ebp  
00442EBD 8BEC    mov ebp,esp  
00442EBF 8B550C  mov edx,[ebp+$0c]  
00442EC2 8B4508  mov eax,[ebp+$08]  
Unit1.pas.28: if A>B then  
00442EC5 3BD0    cmp edx,eax  
00442EC7 7C02    jl Max + $F  
Unit1.pas.31: Max := B;  
00442EC9 8BC2    mov eax,edx  
Unit1.pas.32: end;  
00442ECB 5D      pop ebp  
00442ECC C20800  ret $0008
```

Orodja in pripomočki za razvoj programov

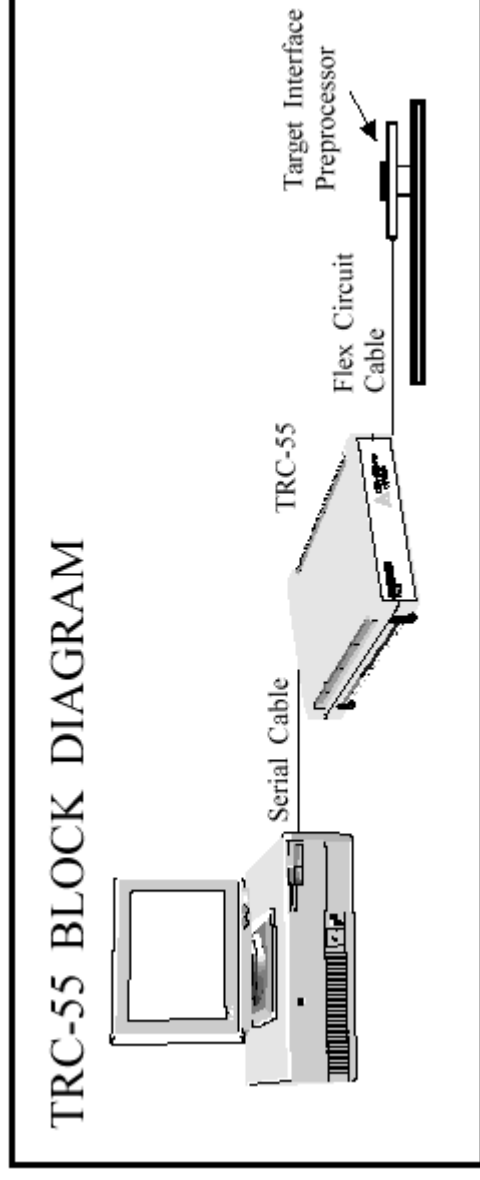
- Zbirniki in prevajalniki
 - ◆ navzkrižni (cross) prevajalniki
- Programatorji



■ Simulatorji: programska simulacija



- Emulatorji: simulacija obnašanja na strojnem nivoju



- Logični analizator, osciloskop