Laboratorij za načrtovanje integriranih vezij

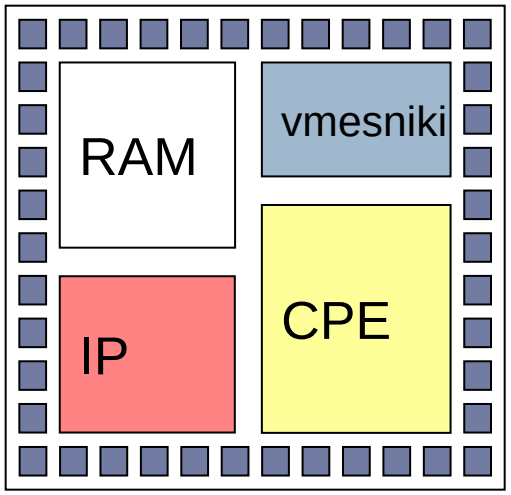
Univerza *v Ljubljani*
Fakulteta *za elektrotehniko*

Programirljivi Digitalni Sistemi

# Digitalni sistem

## Digitalni sistemi na integriranem vezju

# Digitalni sistem na integriranem vezju

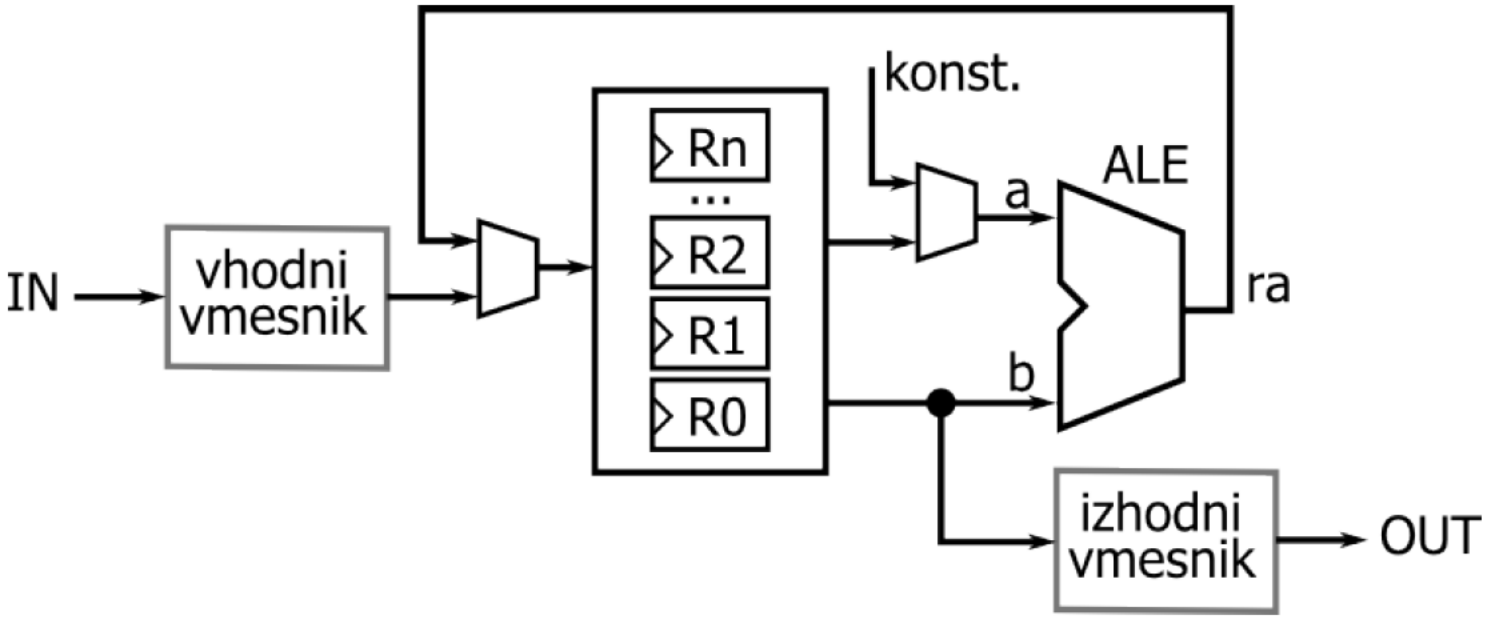



SoC: integrirano vezje, ki izvaja vse ali večino funkcij celotnega elektronskega sistema

- Vmesniki za prenos podatkov v in iz sistema
- Obdelava podatkov se izvaja v
  - Centralno Procesni Enoti (CPE)
  - namensko razvitih komponentah intelektualne lastnine (IP)
- Porazdeljeno shranjevanje in obdelava podatkov

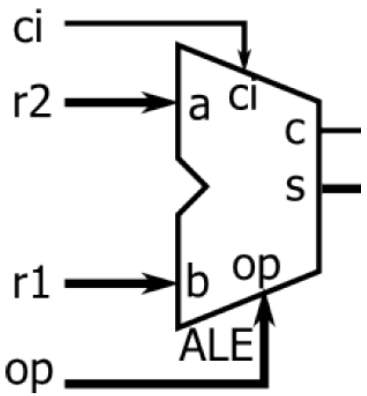
# Obdelava podatkov v CPE

- Centralno procesna enota dobi vhodne podatke iz registrov ali vhodnega vmesnika,



# Aritmetično logična enota – ALE

- ALE izvaja različne mikrooperacije, kot npr:

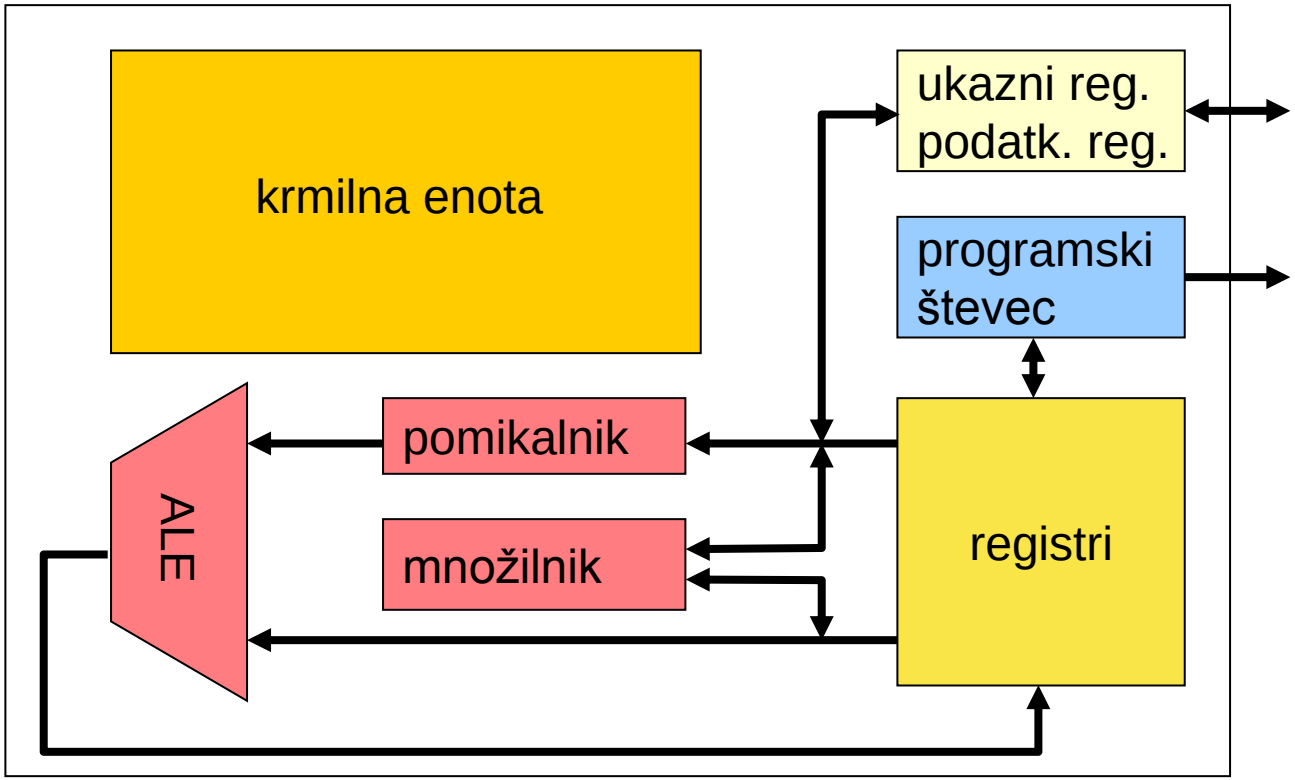



| | | |
|---|---|---|
| $s <= a$ | $c <= 0$ | (prenos podatka) |
| $s <= a + b$ | $c <= co$ | (seštevanje) |
| $s <= a + b + ci$ | $c <= co$ | (seštevanje s prenosom) |
| $s <= a - b$ | $c <= co$ | (odštevanje) |
| $s <= a - (b + ci)$ | $c <= co$ | (odštevanje z izposojo) |
| $s <= a$ AND $b$ | $c <= 0$ | (logični in) |
| $s <= a$ OR $b$ | $c <= 0$ | (logični ali) |
| $s <= 0 : a[3..1]$ | $c <= a[0]$ | (pomik v desno) |
| $s <= a[2..0] : 0$ | $c <= a[3]$ | (pomik v levo) |

- ALE je narejena s povezavo več registrskih celic
  - npr. 32-bitna ALE vsebuje 32 registrskih celic

# Centralna procesna enota ARM-7

- Zmogljivi procesorji vsebujejo dodatne računske enote
  - ciklični pomikalnik, množilnik, enote za plavajočo vejico



# Mikrooperacije v zbirniku

- Ukazi v zbirniku predstavljajo procesorske mikrooperacije

| **mikrooperacija** | **zbirnik** | **pomen** |
|---|---|---|
| $ra <= rb$ | LOAD $ra, rb$ | prenos podatka med registri |
| $ra <= k$ | LOAD $ra, K$ | prenos konstante K iz pomnilnika |
| $ra <= ra + rb$ | ADD $ra, rb$ | vsota dveh registrov |
| $ra <= ra + k$ | ADD $ra, K$ | vsota registra in konstante |
| $ra <= ra$ OR $rb$ | OR $ra, rb$ | logični ali med registri |
| $ra <= 0 \ : \ ra[3..1]$ | SR0 $ra$ | pomik registra v desno |

- Prenos podatkov iz perifernih enot

| | | |
|---|---|---|
| $ra <= IN$ | IN $ra$, ID | prenos iz vhodne enote ID |
| $OUT <= rb$ | OUT $rb$, ID | prenos iz registra na izhodno enoto ID |

# 1. primer: program za izračun povprečja

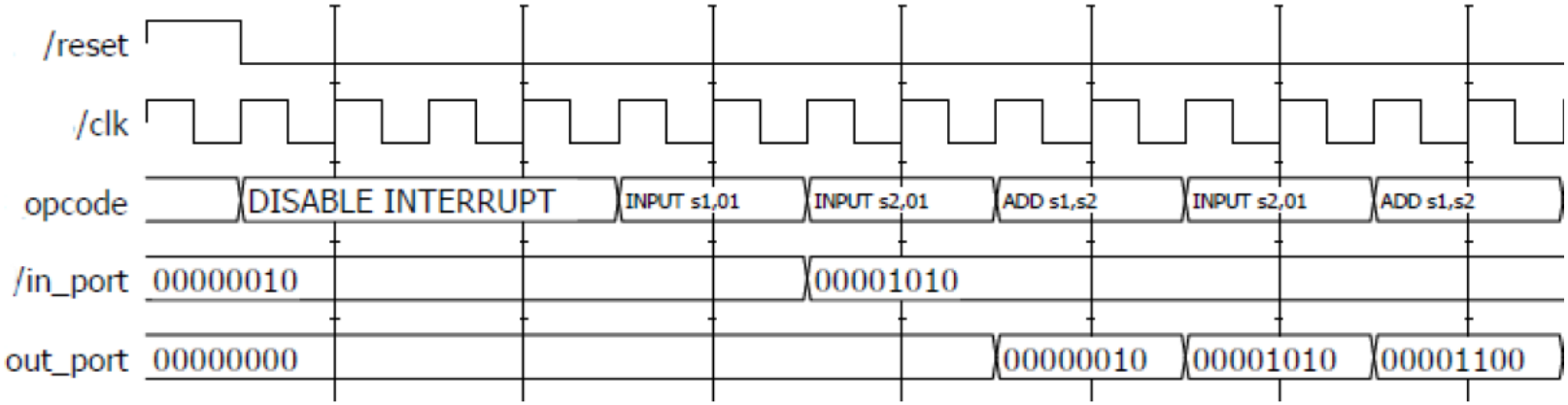
- Program izračuna povprečje štirih vrednosti

```
start:  IN    s1, VHD  ; beri 1. podatek
        IN    s2, VHD  ; 2. podatek
        ADD   s1, s2   ; vsota
        IN    s2, VHD  ; 3. podatek
        ADD   s1, s2   ; vsota
        IN    s2, VHD  ; 4. podatek
        ADD   s1, s2   ; vsota
        SR0   s1       ; deli z 2
        SR0   s1       ; deli z 2
        OUT   s1, IZH  ; na izhod
```

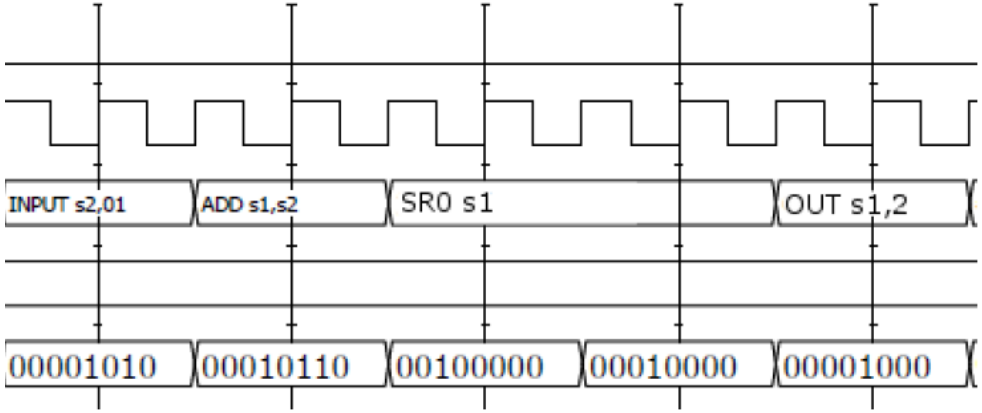
- Podatke beremo iz vhodne enote, izračunamo vsoto in jo delimo s 4 z dvema zaporednima operacijama pomika
  - pomik v desno SR0 predstavlja deljenje z 2

# Simulacija 8-bitnega procesorja

- Delovanje procesorja lahko opazujemo na simulatorju
  - primer na procesorju Picoblaze v programirljivem vezju

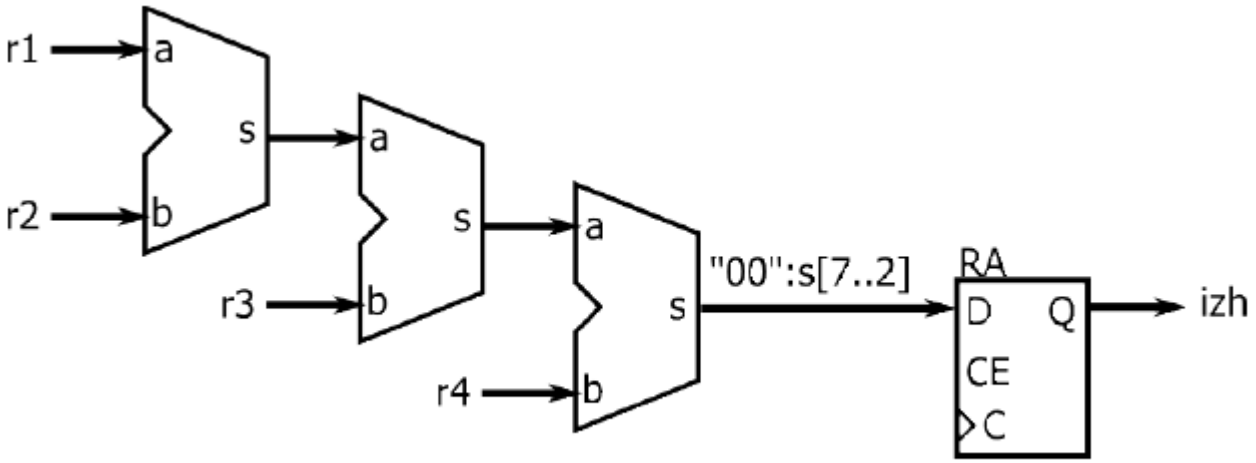



- Procesor izračuna izhod v 20 urnih ciklih



# Izračun povprečja z digitalnim vezjem

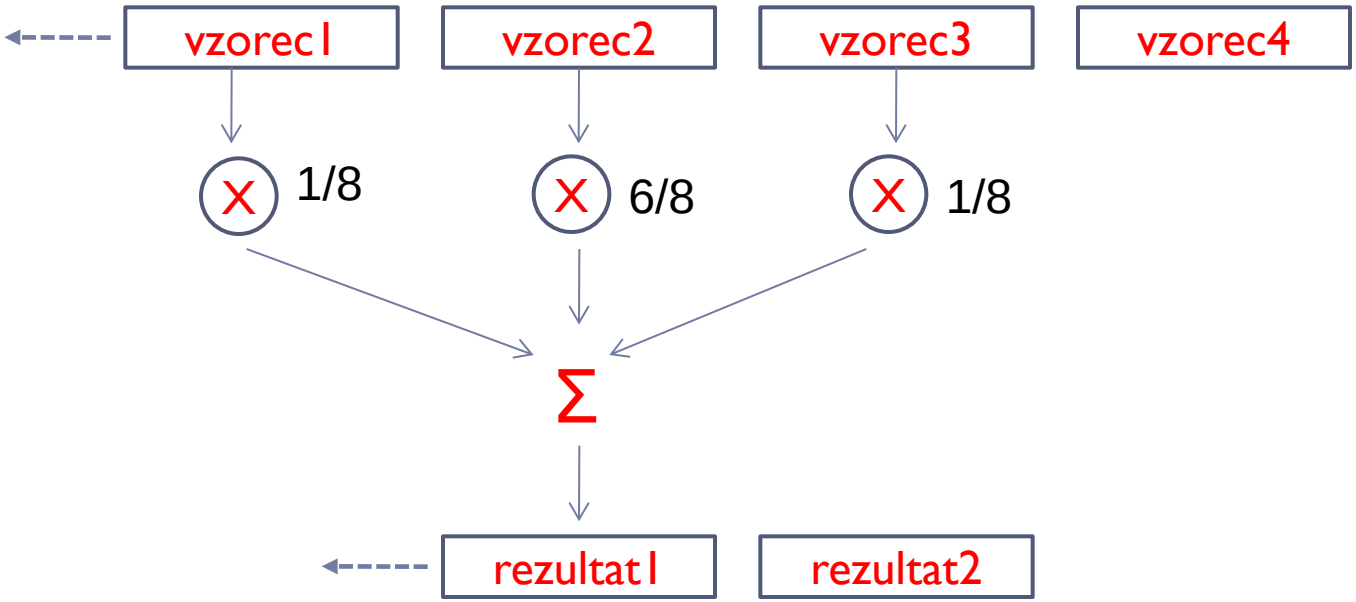
- V vezju lahko paralelno izvedemo več mikrooperacij
- Porazdeljena obdelava podatkov
- Rezultat dobimo v enem urnem ciklu !




- Prednost porazdeljene obdelave podatkov: hitrost
- Slabost: velikost vezja, pri vsaki spremembi je potrebno razviti novo vezje (razvoj programa je hitrejši)

# 2. primer: Gaussovo sito

- Zaporedje digitalnih vzorcev pošljemo skozi sito
  - obdelava zvoka, slike …
- Filtriranje izvedemo z matematično konvolucijo, kjer zaporedne vzorce množimo s konstantami in seštejemo
- Npr. Gaussovo nizko sito s koeficienti 1/8, 6/8, 1/8

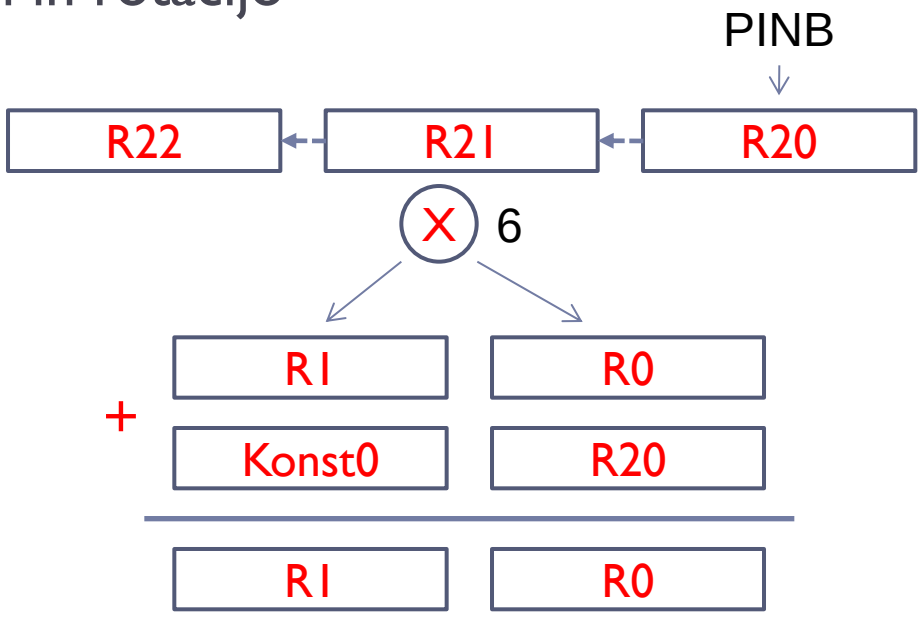


# Obdelava podatkov s procesorjem AVR

- 8-bitne vzorce preberemo iz PINB
- naredimo množenje s 6 in seštejemo 16-bitne vrednosti
- naredimo deljenje z 8 s pomikom in rotacijo

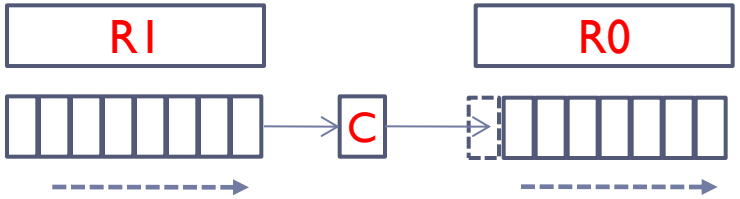
```
loop:
    MOV r22, r21  ;premakni
    MOV r21, r20  ;premakni
    IN  r20, PINB ;beri

    MUL r21, Konst6  ; x6
    ADD r0, r20      ;vsota
    ADC r1, Konst0   ;prenos
    ADD r0, r22      ;vsota
    ADC r1, Konst0   ;prenos
```



# Obdelava podatkov s procesorjem AVR

- naredimo deljenje z 8 s pomikom in rotacijo
- pošljemo rezultat na PORTD




```
LSR r1   ; 3x pomik
ROR r0   ; in rotacija
LSR r1
ROR r0
LSR r1
ROR r0

OUT PORTD, r0  ; izhod
JMP loop
```

# Izvajanje programa

- Kako vemo, kdaj je na voljo podatek (rezultat) ?

1. Naredimo usklajevalni protokol z uporabo ostalih priključkov
   - potrebnih še dodatne procesorske cikle

2. Napišemo zanko v prekinitveni rutini
   - procesor ob zunanjem prekinitvenem signalu skoči na rutino in napravi en računski cikel
   - komunikacijske enote izvedejo prekinitev, ko dobijo nov podatek

# Mikrosekvenčnik s skočnimi ukazi

- Programski števec (PC) naloži nov naslov, kar omogoča izvedbo skokov
  - algoritmi, ki nimajo vseh mikroukazov v zaporedju
  - izvedbo pogojnega skoka določa vhodna krmilna logika