

# Računala i procesi

---

# *FRISC-MC*

*Zavod za automatiku i procesno računarstvo  
grupa RASIP*

# FRISC-MC

---

- Općeprogramski primjeri
- PIO jedinice
- Rad s portovima
- Jedinica za A/D konverziju
- Primjeri

# Općeprogramski primjeri

---

- Nećemo raditi
- Podsjetite se algoritama za dijeljenje i množenje – koristit će se kao potprogrami

# PIO jedinica

---

- Spojena na port – ulazno/izlazne linije
- Može dijeliti port s nekom drugom ugrađenom jedinicom (PWM, Timer)
- Prioritet nad korištenjem porta ima VJ a ne PIO
- Određuje smjer porta kad njime upravlja
- Vidi se u memoriji kao vanjska jedinica (zauzima određeni broj adresa)
- Ne izaziva prekide

# Programski model PIO jedinice

---

- Svaka PIO jedinica (ima ih 8) ima jednu pripadajuću adresu
- Radi kao bezuvjetna jedinica (kontrolira smjer porta)
- DP<sub>x</sub> – ponaša se kao *registar* smjera porta P<sub>x</sub>

DP0 na adresi %D248 ; PIO 0

DP1 na adresi %D249 ; PIO 1

...

DP7 na adresi %D255 ; PIO 7

# Određivanje smjera porta

---

- Svakom portu pridružen je *direction* registar koji za **svaku** liniju porta određuje njen smjer
- Značenje bitova registra DP<sub>x</sub>:

Vrijednost bita	Smjer
DP <sub>x.y</sub> = 0	P <sub>x.y</sub> ulazni
DP <sub>x.y</sub> = 1	P <sub>x.y</sub> izlazni

- Promjena smjera moguća je u svakom trenutku
- Inicijalna vrijednost registara DP<sub>x</sub> je 0 (sve linije ulazne)

# Prijenos podataka

---

- Posebne naredbe za pristup portu:

**PIN R0, P0 ; citaj vrijednost s P0**

**POUT R1, P1 ; pisi vrijednost na P1**

**PCLEAR P2, 3 ; brisi P2.0 i P2.1**

**PSET P3, 4 ; postavi P3.2**

**PCOMPL P4, 0 ; ne radi nista**

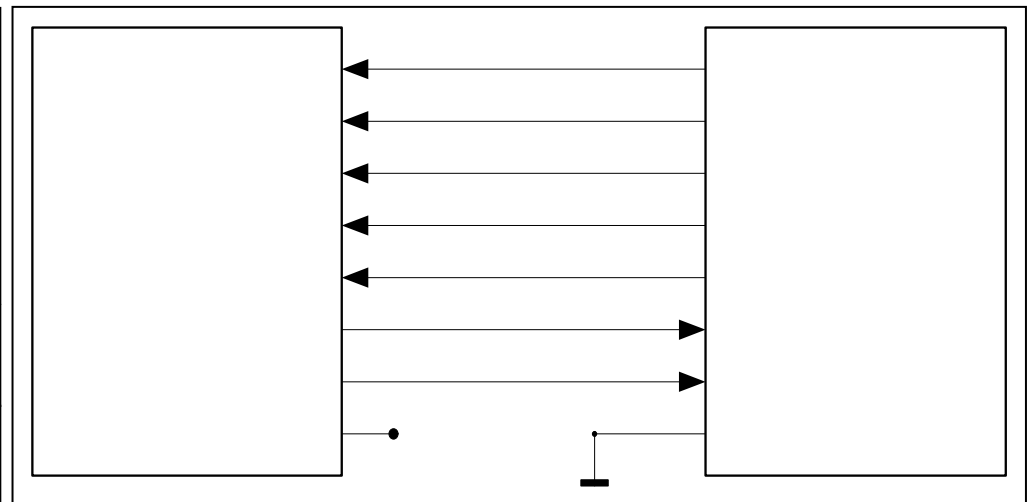
**PTEST P5, 8 ; ispituje P5.3**

# Primjer rada s PIO i portom

---

**Zadatak:** Na port P0 spojena je vanjska tipkovnica prema tablici. Tipkovnica nakon pritiska tipke postavlja liniju RDY u stanje 1, nakon čega je moguće pročitati podatak. Podatak se prenosi preko linija DT. Nakon čitanja podatka potrebno je to dojaviti jedinici impulsom na liniji ACK. Tipkovnica se uključuje linijom P0.6. Podatak spremi na lokaciju 200.

Port	Tipkovnica
P0.0 – P0.3	DT0 – DT3
P0.4	RDY
P0.5	ACK
P0.6	VCC

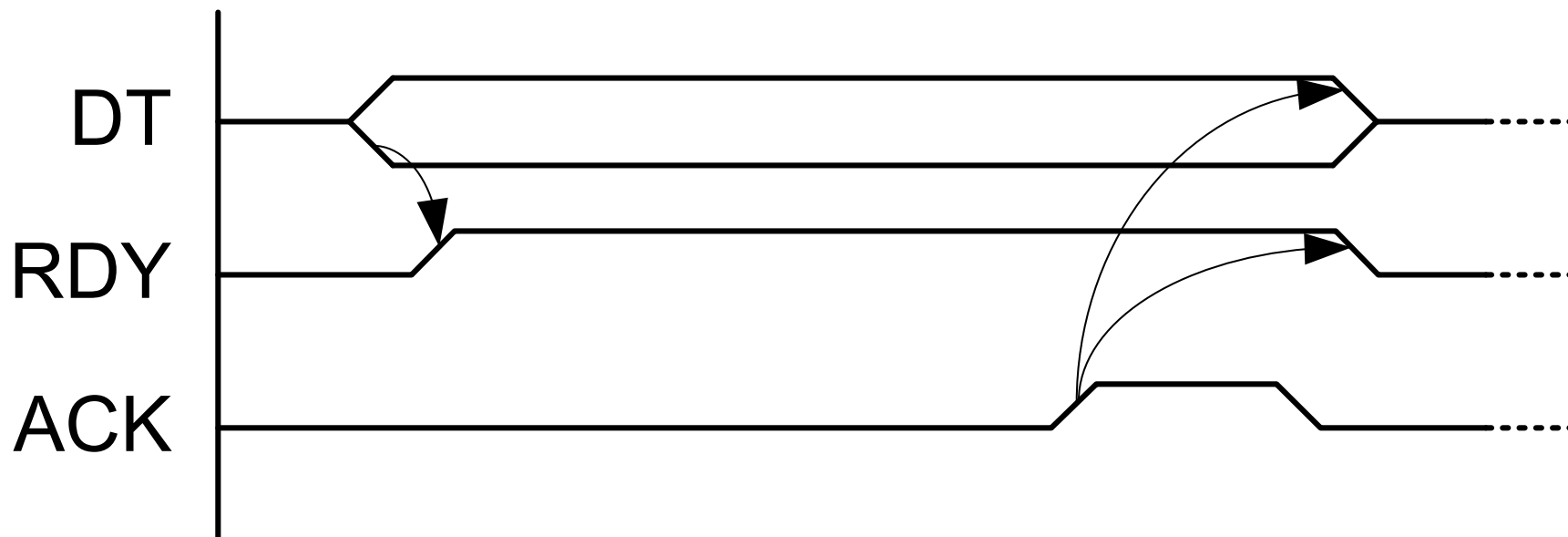




# Primjer rada s PIO i portom (2)

---

- Komunikacija između FRISC-MC i tipkovnice:



# Primjer rada s PIO i portom (3)

---

- Programiranje PIO:

```
LOADL R0, %B11100000 ; 3 bita izlazna  
STORE R0, (DP0) ; postavi smjer
```

- Linija P0.7 visi u zraku (što je greška u dizajnu), da je priključak postavljen kao ulazni došlo bi do greške, pa ga postavimo kao izlazni

# Primjer rada s PIO i portom (4)

---

- Implementacija protokola komunikacije:

```
        PSET    P0, %B01000000    ; ukljuci tipk. Vcc=P0.6

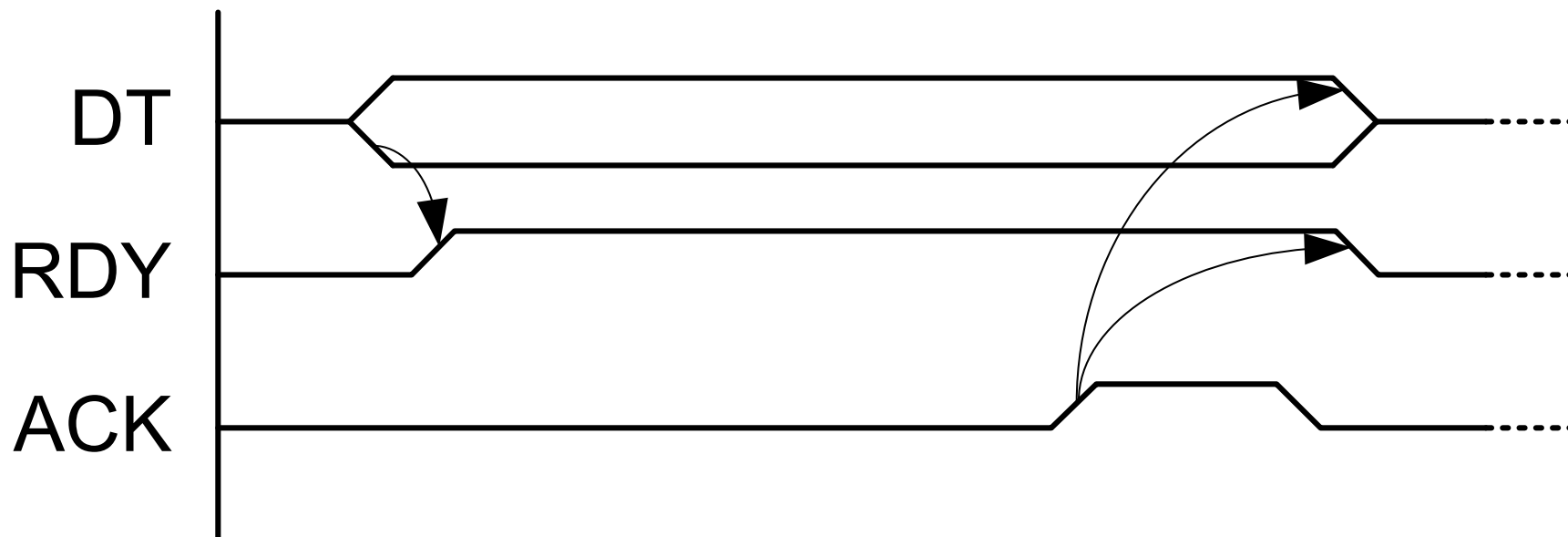
LOOP PTEST  P0, %B00010000    ; P0.4 = RDY
      JP_PZ   LOOP              ; cekaj RDY = 1

      PIN    R0, P0             ; ucitaj
      AND    %B1111, R0, R0     ; brisi visak
      STORE  R0, (200)          ; spremi procitanu tipku
      PSET   P0, %B00100000     ; P0.5 = ACK = 1
      PCLEAR P0, %B00100000     ; P0.5 = ACK = 0
      JP     LOOP              ; nastavi dalje
```

# Primjer rada s PIO i portom (5)

---

- Komunikacija između FRISC-MC i tipkovnice:



# Primjer rada s PIO i portom (6)

---

- Mogući problem u prethodnom primjeru
- Što ako je tipkovnica spora i ne spusti RDY dovoljno brzo?
- U tom slučaju pokušali bi ponovno čitati i onda?
- Potrebno je pričekati na spuštanje linije RDY!

# Primjer rada s PIO i portom (7)

---

- Umjesto:

```
...  
PSET   P0, %B00100000      ; P0.5 = ACK = 1  
PCLEAR P0, %B00100000      ; P0.5 = ACK = 0  
JP     LOOP                 ; nastavi dalje
```

- Trebalo bi:

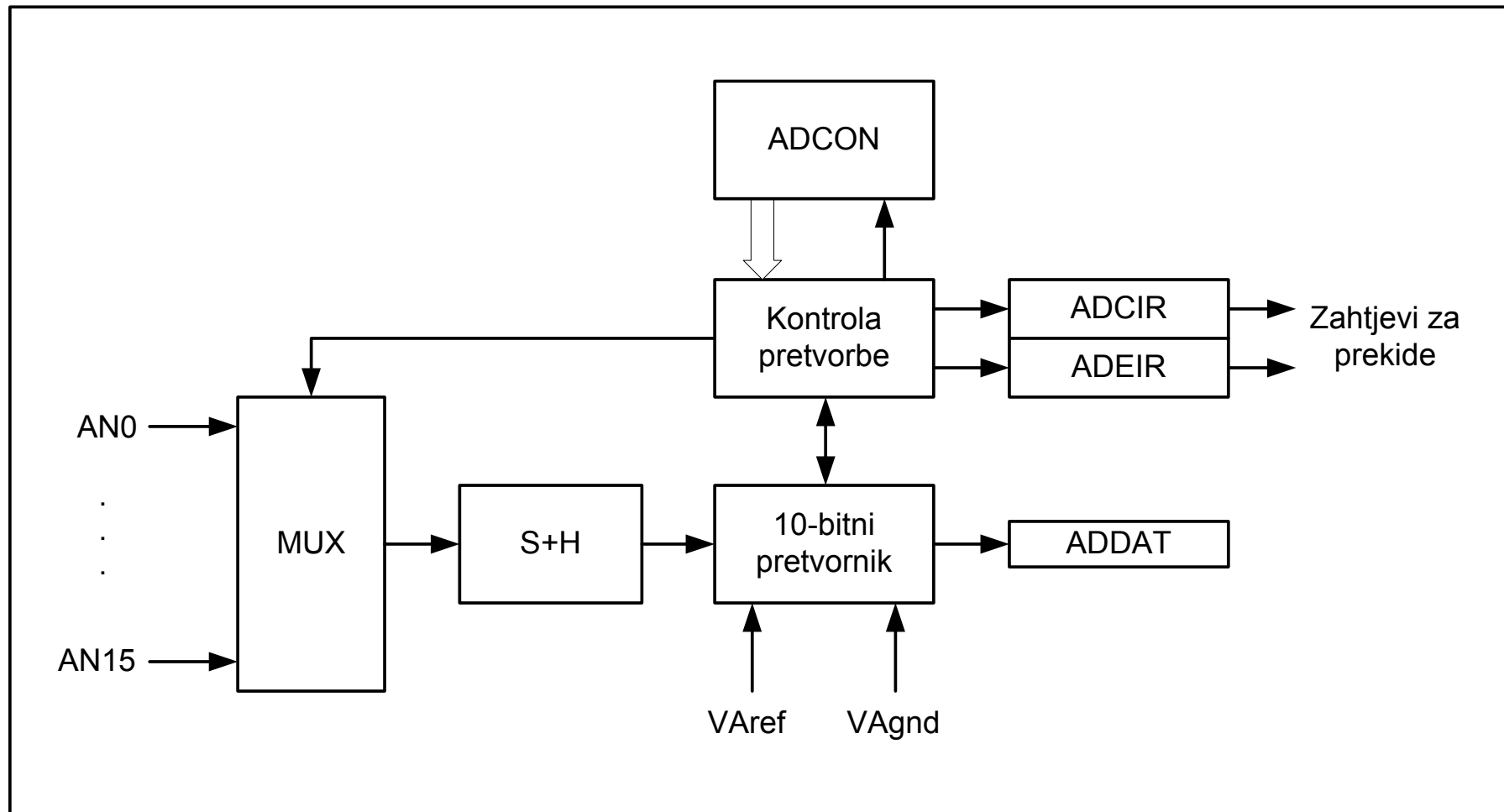
```
...  
PSET   P0, %B00100000      ; P0.5 = ACK = 1  
PCLEAR P0, %B00100000      ; P0.5 = ACK = 0  
WAIT   PTEST P0, %B00010000 ; P0.4=RDY  
JP_PNZ WAIT  
JP     LOOP
```

# Jedinica za A/D pretvorbu

---

- Nije spojena na port
- Predviđene su posebne linije za napajanje kako bi se osigurala točnost pretvorbe
- Ima posebne analogne ulaze AN0 – AN15 koji se biraju multipleksorom – programski ili automatski
- U osnovi 10-bitni A/D pretvarač sa *sample&hold* sklopom
- Zauzima 4 lokacije u memoriji kao vanjska jedinica
- Može izazvati prekide (ADCIR i ADEIR)

# A/D pretvornik – shema sklopa





# Jedinica za A/D pretvorbu

---

- Načini rada:
  - Jednokratna pretvorba odabranog kanala
  - Kontinuirana pretvorba jednog kanala
  - Jednokratna automatska pretvorba grupe kanala
  - Kontinuirana automatska pretvorba grupe kanala
  
  - Blokiranje pretvorbe do očitavanja pretvorenog kanala

# Registri A/D pretvornika

---

- ADCON – kontrola rada jedinice, načini rada, odabir kanala
- ADDAT – pohranjuje rezultat pretvorbe
- ADCIC – kontrola prekida nakon kraja pretvorbe (*conversion complete interrupt*)
- ADEIC – kontrola prekida na pojavu greške (*conversion error interrupt*)

# Registri A/D pretvornika – ADCON

---

- ADCON – kontrola rada

10	9	8	7	6	5	4	3	2	1	0
-	ADWR	ADBSY	ADST	-	ADM		ADCH			

- ADCH – odabir (prvog) kanala za pretvorbu
- ADM – odabir načina rada
- ADST – pokretanje pretvorbe
- ADBSY – zastavica koja označava da je pretvorba u tijeku
- ADWR – kontrola čekanja na čitanje rezultata pretvorbe

# Registri A/D pretvornika – ADCON

---

- ADCON – kontrola rada

10	9	8	7	6	5	4	3	2	1	0
-	ADW R	ADB SY	ADS T	-	ADM		ADCH			

- ADM – odabir načina rada
  - $00_2$  – jedna pretvorba jednog kanala
  - $01_2$  – kontinuirana pretvorba jednog kanala
  - $10_2$  – jedna pretvorba grupe kanala
  - $11_2$  – kontinuirana pretvorba grupe kanala

# Registri A/D pretvornika – ADCON

---

- ADCON – kontrola rada

10	9	8	7	6	5	4	3	2	1	0
-	ADW R	ADB SY	ADS T	-	ADM	ADCH				

- ADCH – odabir (prvog) kanala za pretvorbu
  - U rasponu od 0 – 15<sub>10</sub>, što odgovara kanalima AN0 – AN15
  - Ako se odabere pretvorba grupe kanala, onda ADC počinje sa pretvorbom od broja kanala zapisanog u ADCH pa do kanala 0 **naniže** i tako ponovno

# Registri A/D pretvornika – ADCON

---

- ADCON – kontrola rada

10	9	8	7	6	5	4	3	2	1	0
-	ADW R	ADB SY	ADS T	-	ADM	ADCH				

- ADST – pokretanje pretvorbe
  - 0 – zaustavlja aktivnu pretvorbu
  - 1 – pokreće pretvorbu

# Registri A/D pretvornika – ADCON

---

- ADCON – kontrola rada

10	9	8	7	6	5	4	3	2	1	0
-	ADW R	ADB SY	ADS T	-	ADM		ADCH			

- ADBSY – zastavica koja označava da je pretvorba u tijeku
  - 0 – A/D pretvornik miruje
  - 1 – pretvorba u tijeku

# Registri A/D pretvornika – ADCON

---

- ADCON – kontrola rada

10	9	8	7	6	5	4	3	2	1	0
-	ADWR	ADBSY	ADST	-	ADM	ADCH				

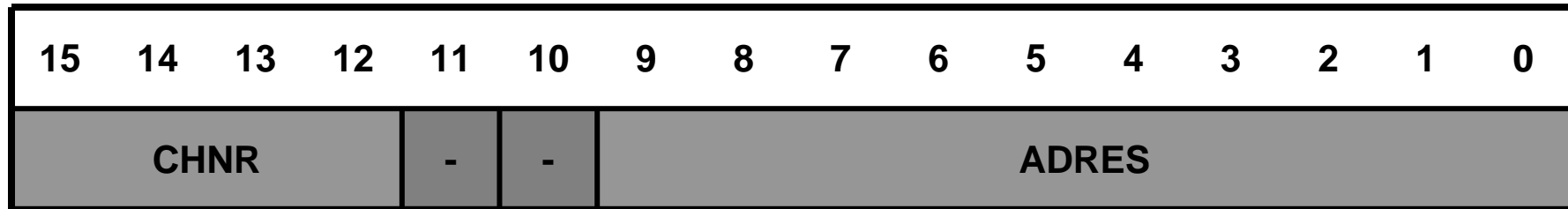
- ADWR – kontrola čekanja na čitanje rezultata pretvorbe
  - 0 – ne koristi se čekanje
  - 1 – čeka se s pretvorbom dok se rezultat ne pročita



# Registri A/D pretvornika – ADDAT

---

- ADDAT – pohranjuje rezultat pretvorbe



- CHNR – broj pretvorenog analognog kanala ( $0 - 15_{10}$ , što odgovara kanalu AN0 – AN15)
- ADRES – rezultat pretvorbe (10 bita,  $0 - 1023$ )

# Registri A/D pretvornika – ADCIC

---

- ADCIC – kontrola prekida na kraj pretvorbe (*conversion complete interrupt*)

7	6	5	4	3	2	1	0
ADCIR	ADCIE	-	-	-	ILVL		

- ILVL – određuje razinu prekida (0 – 7), razina 0 ne izaziva prekid
- ADCIE – *AD conversion complete interrupt enable* – uključuje postavljanje zahtjeva za prekid
- ADCIR – *AD conversion complete interrupt request* – označava postavljen zahtjev za prekid, automatski se briše nakon prihvaćanja prekida

# Registri A/D pretvornika – ADEIC

---

- ADEIC – kontrola prekida na grešku u pretvorbi (*conversion error interrupt*)

7	6	5	4	3	2	1	0
ADEIR	ADEIE	-	-	-	ILVL		

- ILVL – određuje razinu prekida (0 – 7), razina 0 ne izaziva prekid
- ADEIE – *A/D error interrupt enable* – uključuje postavljanje zahtjeva za prekid
- ADEIR – *A/D error interrupt request* – označava postavljen zahtjev za prekid, automatski se briše nakon prihvatanja prekida

# A/D pretvorba – načini rada

---

- **Pretvorba odabranog kanala**

- Ovaj način rada odabire se programiranjem ADM polja u ADCON registru ( $00_2$  – jednokratna pretvorba;  $01_2$  – kontinuirana pretvorba)
- Nakon postavljanja ADST bita pretvorba počinje i ADBSY bit je aktivan cijelo vrijeme pretvorbe
- Nakon svake pretvorbe sklop generira ADCIR prekid
- Kod jednokratne pretvorbe – nakon završene pretvorbe ADBSY i ADST bitovi postavljaju se u nulu i rezultat pretvorbe je spremljen u ADDAT registru
- Kod kontinuirane pretvorbe – nakon završene pretvorbe kanala rezultat se sprema u ADDAT i automatski se započinje novo uzorkovanje istog kanala. Pretvorba se nastavlja dok bit ADST ne bude pobrisan od strane programa

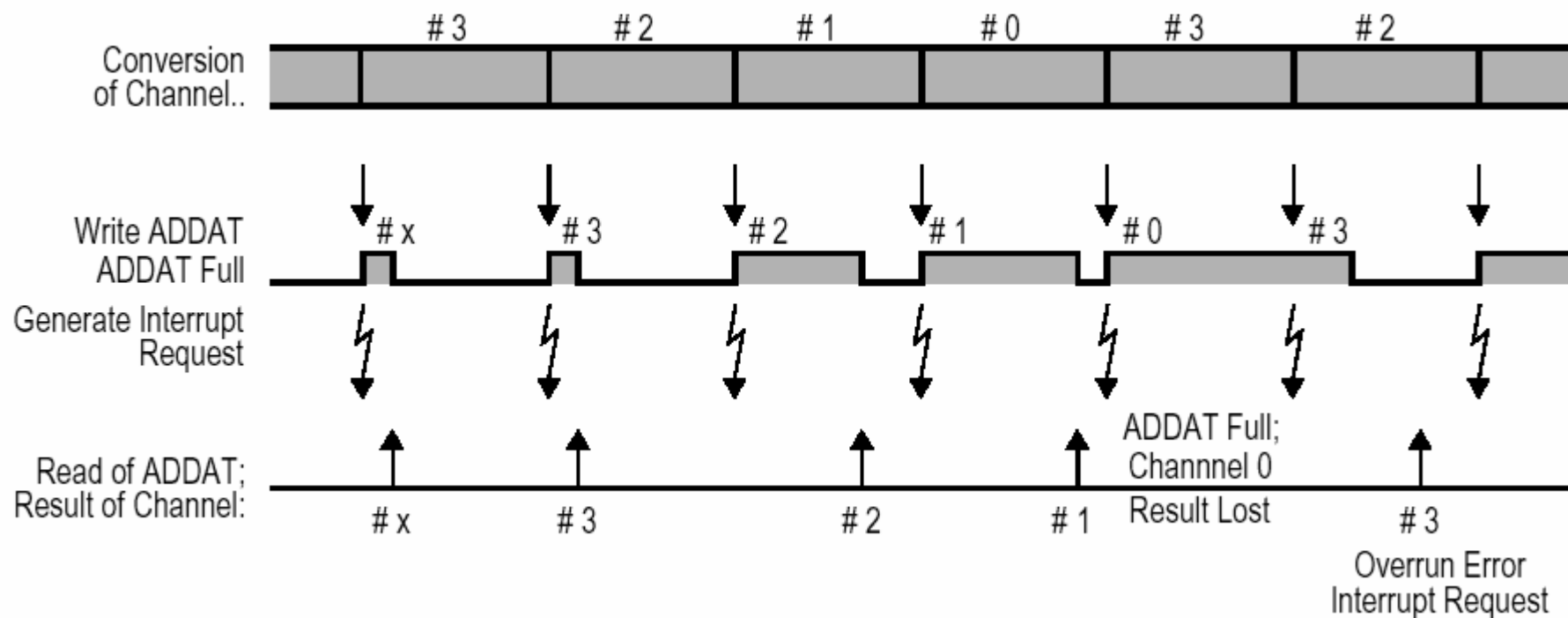
# A/D pretvorba – načini rada (2)

---

- Pretvorba grupe kanala
  - Odabire se postavljanjem ADM polja na vrijednosti  $10_2$  ili  $11_2$  (jednokratna ili kontinuirana pretvorba)
  - Pretvorba kreće od kanala specificiranog u polju ADCH i broj kanala koji se pretvara smanjuje se prema 0
  - Nakon svakog pretvorenog kanala sklop generira ADCIR prekid i automatski počinje pretvorbu sljedećeg kanala
  - Kada se završi pretvorba nultog kanala, tada ovisno o odabranom načinu rada sklopovlje ili zaustavlja pretvorbu (ako je odabrana jednokratna pretvorba) ili nastavlja ponovo od kanala određenog sa ADCH poljem (kod kontinuirane pretvorbe)

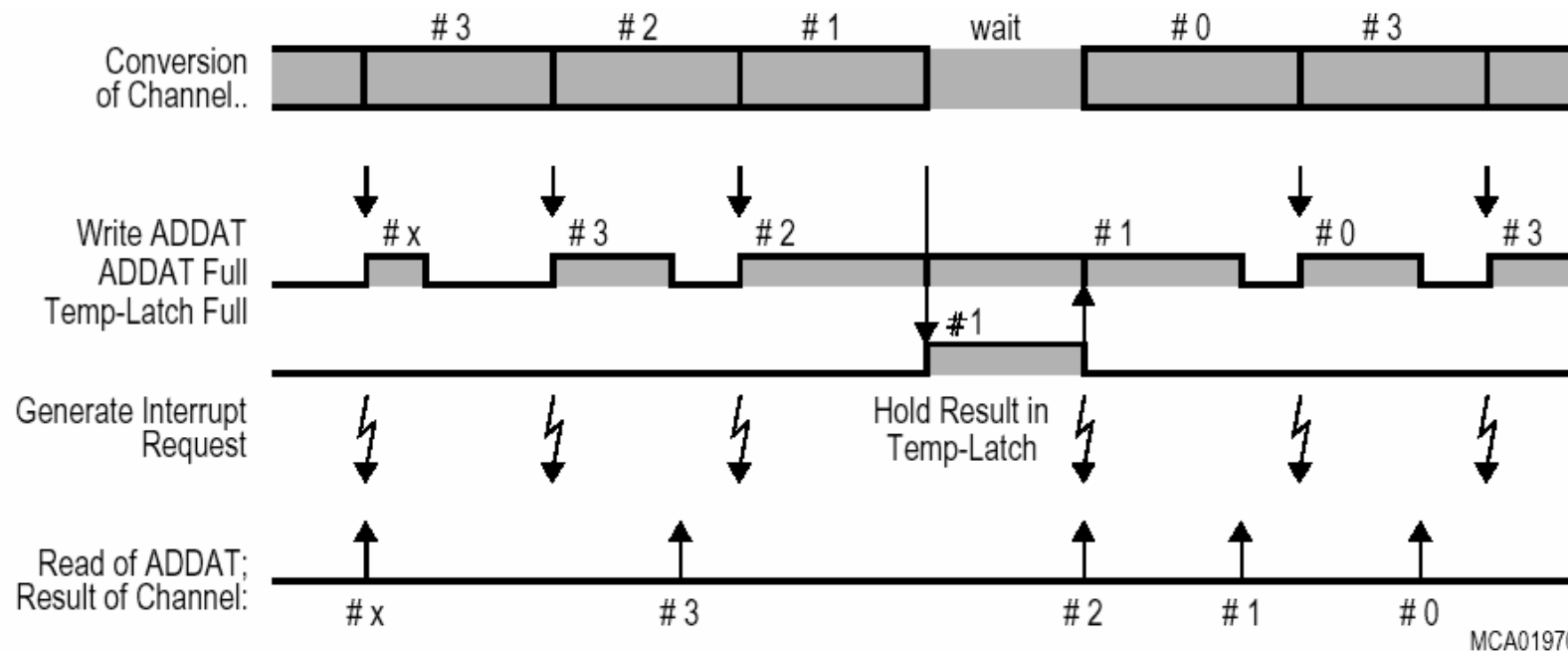
# A/D pretvorba – načini rada (3)

- Ako prilikom čitanja rezultata pretvorbe program ne reagira dovoljno brzo na obradu prekida, može doći do prepisivanja rezultata pretvorbe
- Prilikom pogreške prepisivanja postavlja se ADEIR zahtjev za prekid



# A/D pretvorba – načini rada (4)

- Čekanje na čitanje rezultata pretvorbe (*wait for read*)
  - Aktivira se postavljanjem bita ADWR u ADCON registru
  - Ako do trenutka pojavljivanja novog rezultata pretvorbe stari nije bio pročitan novi rezultat se pohranjuje u privremeni registar i daljnja pretvorba se zaustavlja do čitanja podatka iz ADDAT registra (nakon čitanja se podatak iz privremenog registra prebacuje u ADDAT registar i pretvorba se može nastaviti)



# Programski model A/D pretvornika

---

- ADC jedinica koristi 15 posebnih izvoda na mikrokontroleru
- Može raditi kao bezuvjetna jedinica (potrebno je čekati na bit ADBSY u ADCON)
- Može raditi kao prekidna jedinica

```
        `ORG 1
ADCIR   `DW  20 ; prekidni vektor ADC

        `ORG 2
ADEIR   `DW  40 ; prekidni vektor ADE
```



# Programski model A/D pretvornika

---

- Registri u memoriji

**ADCON** na adresi %D240 ; kontrola rada

**ADDAT** na adresi %D241 ; rezultat pretvorbe

**ADCIC** na adresi %D242 ; kontrola ADCIR

**ADEIC** na adresi %D243 ; kontrola ADEIR

# Mjerenje jednog kanala

---

- **Zadatak:**

Napisati program koji uzorkuje analognu vrijednost svjetlomjera spojenog na analogni ulaz AN0. Dobivenu digitalnu vrijednost potrebno je pohraniti na lokaciju  $140_{10}$  i zaustaviti mikrokontroler.

# Program

---

```
        `ORG    0
INIT    LOADL  R0, %B010000000 ; ADM=0,ADCH=0,ADWR=0,ADST=1
        STORE  R0, (ADCON)      ; pocinje pretvorba

WAIT    LOAD   R0, (ADCON)
        AND    %H100, R0, R0    ; maskiraj ADBSY
        JP_NZ  WAIT            ; cekaj dok je ADBSY=1

        LOAD   R0, (ADDAT)      ; ucitaj rezultat
        AND    %H3FF, R0, R0    ; maskiraj korisni dio
        STORE  R0, (%D140)      ; pohrani rezultat

PWRDN
```

(9 naredbi)

# Mjerenje više kanala

---

- **Zadatak:**

FRISC-MC nadzire temperaturu i tlak u procesu. Napon iz pretvarača temperature dovodi se na ulaz AN0, a napon iz pretvarača tlaka na ulaz AN1. Aktivnost procesa kontrolira se pomoću linije P2.0 (0=neaktivan, 1=aktivan).

Potrebno je konstantno mjeriti temperaturu. Ukoliko se vrijednost temperature podigne iznad 178°C, treba izmjeriti tlak. Ukoliko je tlak veći od 75 kp/cm<sup>2</sup> potrebno je zaustaviti proces i uključiti alarm postavljanjem linije P2.1.

Područje mjerenja temperature je 20 do 350 [°C], a tlaka 0 do 100 [kp/cm<sup>2</sup>].

# Mjerenje više kanala

---

- Kolika je najveća moguća greška očitavanja temperature i tlaka?
- Napisati programski kod za realizaciju zadatka. Riješiti ga uporabom prekida.
- Napisati dodatni kod koji zapisuje svaku očitanu vrijednost temperature na lokaciju  $128_{10}$ , pretvorenu u stupnjeve Celzija.

# Proračun mjerenja temperature

---

- Razlučivost mjerenja temperature odgovara 1 bitu (LSB)

$$T_0 = \frac{T_{MAX} - T_{MIN}}{2^B} = \frac{330}{1024} = 0.322[^\circ C]$$

- Digitalna vrijednost koja odgovara stvarnoj graničnoj temp.

$$T = 178[^\circ C] \Rightarrow N_T = \frac{T - T_{MIN}}{T_0} = \frac{178 - 20}{0.322} = 490.68 \Rightarrow N_T = 490$$

# Proračun mjerenja tlaka

---

- Razlučivost mjerenja tlaka (odgovara 1 bitu – LSB)

$$p_0 = \frac{p_{MAX} - p_{MIN}}{2^B} = \frac{100}{1024} = 0.097 [kp / cm^2]$$

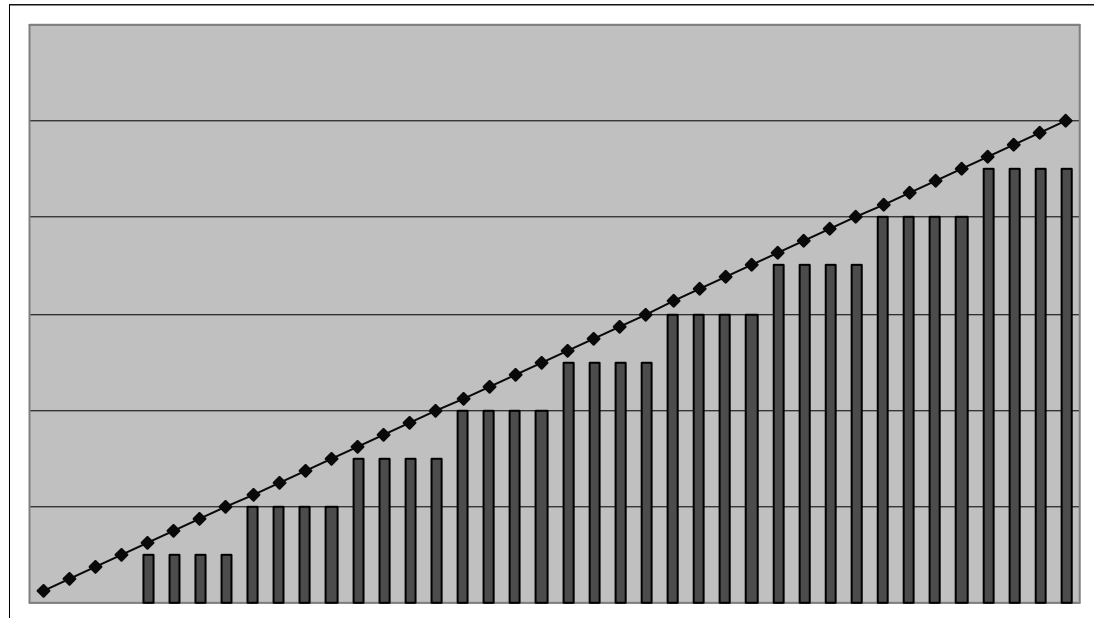
- Digitalna vrijednost koja odgovara stvarnom graničnom tlaku

$$p = 75 [kp / cm^2] \Rightarrow N_p = \frac{p - p_{MIN}}{p_0} = \frac{75 - 0}{0.097} = 773.20 \Rightarrow N_p = 773$$

# Proračun pogreške

---

- Najveća pogreška pri mjerenju temperature je  $0.322[^\circ\text{C}]$ , a pri mjerenju tlaka  $0.097[\text{kp}/\text{cm}^2]$ , ako se uzme u obzir prijenosna funkcija A/D pretvarača, koja za vrijednost analognog ulaza  $0..T1$  daje vrijednost 0, za  $T0..T1$  vrijednost 1, itd. (*stepenice*).





# Inicijalizacija prekidnih vektora

---

```
`ORG      1      ; vektor za kraj mjerenja
`DW      30     ; prekidni potprogram

`ORG      2      ; vektor za grešku u mjerenju
`DW      60     ; prekidni potprogram
```

- Lokacije prekidnih vektora definirane su prekidnim sustavom FRISC-MC

# Inicijalizacija A/D pretvarača

---

```
`ORG    0
JP  INIT                ; preskoci prekidne vektore
`DW 30
`DW 60

INIT  LOADL  R7, %HDF      ; inicijaliziraj stog (R7 ili SP)
      LOADL  R0, %B11     ; postavi P2.0 i..
      STORE  R0, (DP2)    ; ..P2.1 na izlazne
      PSET   P2, 1        ; ukljuci proces
      PCLEAR P2, 2        ; iskljuci alarm

      LOADL  R0, %H0041   ; 00000000.0.1.000.001 IE=1, ILVL=1
      STORE  R0, (ADCIC)
      INC    R0            ; 00000000.0.1.000.010 IE=1, ILVL=2
      STORE  R0, (ADEIC)

      LOADL  R0, %H80     ; GIE=1, ILVL=0
      WRST   R0

      LOADL  R0, %H290    ; 1.0.1.0.01.0000
      STORE  R0, (ADCON)  ; ADWR=1, ADST=1, MODE=01, CH=0

      IDLE
```

# Prekidni potprogram

---

```
CONVOVER    `ORG    30
            PUSH    R0                ; pohrani kontekst
            RDST    R0
            PUSH    R0

            LOAD    R0, (ADDAT)       ; učitaj podatak
            AND     %H3FF, R0, R0     ; maskiraj korisni dio
            CMP     %D490, R0        ; usporedi s 490 (178 stupnjeva)
            JP_ULT  CONV1            ; ako nije veci

            CALL    CONVP             ; pokreni mjerenje tlaka

CONV1       CALL    STORETEMP        ; spremi temperaturu

            POP     R0                ; vrati kontekst
            WRST    R0
            POP     R0
            RETI     ; povratak iz prekida
```

(13 naredbi)

# Potprogram za mjerenje tlaka

---

```

        ^ORG      40
CONVP   PUSH      R0                ; pohrani kontekst
        LOAD      R0, (ADCON)
        PUSH      R0                ; pohrani ADCON
        LOAD      R0, (ADCIC)
        PUSH      R0                ; pohrani ADCIC
        CLEAR     R0
        STORE     R0, (ADCIC)       ; iskljuci prekide
        LOADL    R0, %H281          ; 1.0.1.0.00.0001
        STORE     R0, (ADCON)       ; ADWR=1, ADST=1, MODE=00, CH=1

WAITP   LOAD      R0, (ADCON)       ; citaj ADBSY
        AND       %H100, R0, R0     ; maskiraj ADBSY
        JP_NZ     WAITP             ; cekaj dok je ADBSY=1

        LOAD R0, (ADDAT)
        AND       %H3FF, R0, R0     ; maskiraj korisni dio
        CMP      %D773, R0         ; usporedi s 773 (75 kp/cm2)
        JP_ULT    CONV2            ; ako nije veci, nista

        PCLEAR   P2, 1             ; obrisi P2.0 (iskljuci proces)
        PSET     P2, 2             ; postavi P2.1 (ukljuci alarm)

CONV2   POP       R0                ; obnovi ADCIC
        STORE     R0, (ADCIC)
        POP      R0                ; obnovi ADCON
        STORE     R0, (ADCON)
        POP      R0                ; vrati kontekst
        RET      ; kraj potprograma (24 naredbi)
```

# Prekidni potprogram za grešku

---

```
        `ORG 60  
CONVERROR  RETI
```

(1 naredba)

# Pretvorba temperature

---

- Za pretvaranje digitalnog oblika temperature u fizikalnu veličinu uz korištenje referentnih vrijednosti temperature

$$T = \frac{N_T \cdot (T_{MAX} - T_{MIN})}{2^B} + T_{MIN} = \frac{N_T \cdot 330}{1024} + 20$$

# Pretvorba temperature

---

```
STORETEMP PUSH R1           ; pohrani kontekst
                               ; R0 je zadan
LOADL  R1, %D330            ; R1=max razlika temperatura
CALL   MULU                 ; R0 * R1 -> R0

LOADL  R1, %D1024          ; max digitalna vrijednost
CALL   DIVU                 ; R0 div R1 -> R0

ADD    %D20, R0, R0        ; dodaj min. temperaturu
STORE  R0, (%D128)         ; pohrani na zadanu lokaciju

POP    R1                   ; vrati kontekst
RET
```

(9 naredbi)

# Pomoćni potprogrami

---

```
MULU    ; R0 * R1 -> R0
        PUSH R2                ; pohrani kontekst
        MOVE R2, R1
MLOOP   TEST R2
        JP_Z MEND
        ADD R0, R1, R0        ; pribroji
        DEC R2
        JP MLOOP
MEND    POP R2                ; obnovi kontekst
        RET                   ; povratak
```

( 9 naredbi )



# Pomoćni potprogrami

---

**DIVU** ; R0 / R1 -> R0

**PUSH** R2

**CLEAR** R2

**DLOOP** **SUB** R0, R1, R0 ; oduzmi

**JP\_ULT** **DEND** ; ako nije uspjelo, kraj

**INC** R2

**JP** **DLOOP**

**DEND** **MOVE** R2, R0 ; postavi rezultat

**POP** R2 ; obnovi kontekst

**RET** ; povratak

**( 9 naredbi )**

# Zaključak

---

- Inicijalizacija – 17 naredbi
- Prekidni potprogram – 13 naredbi
- Prekidni potprogram za obradu greške – 1 naredba
- Potprogram za mjerenje tlaka – 24 naredbi
- Pretvorba temperature – 9 naredbi
- Potprogram za množenje – 9 naredbi
- Potprogram za dijeljenje – 9 naredbi
  
- Ukupno:  $17+13+1+24+9+9+9 = 82$  naredbe (lokacija)
- Zauzeće memorije:
  - ROM: 64%
  - RAM: 1 lokacija + stog

# Mjerenje grupe kanala

---

- **Zadatak:**

Napisati program i pripadne prekidne potprograme koji će uporabom prekidnog mehanizma FRISC-MC kontinuirano očitavati napone na analognim ulazima AN0 do AN7 i pohranjivati ih u tablicu u memoriji. Tablica neka počinje na lokaciji  $80_{16}$  (lokacija  $80_{16}$  sadrži vrijednost kanala AN0, lokacija  $81_{16}$  vrijednost kanala AN1, itd.). Napisati potprogram za obradu eventualnih grešaka vezanih uz A/D pretvarač, koji će povećati brojač grešaka na lokaciji  $90_{16}$  svaki put kad se greška dogodi.

# Inicijalizacija prekidnih vektora

---

```
`ORG      ADCINT      ; vektor za kraj mjerenja  
`DW      10           ; prekidni potprogram
```

```
`ORG      ADEINT      ; vektor za grešku mjerenja  
`DW      20           ; prekidni potprogram
```

# Inicijalizacija A/D pretvarača

---

```
TABLE    `EQU 80
ERRCNT   `EQU 90

        `ORG 0
        JP      INIT

ADCIRQ   `DW 10
ADEIRQ   `DW 20

INIT     LOADL   R7, %HDF      ; inicijalizacija stoga
        CLEAR   R0
        STORE   R0, (ERRCNT) ; inicijalizacija brojaca gresaka
        LOADL   R0, %H41      ; 00000000.0.1.000.001 IE=1,ILVL=1
        STORE   R0, (ADCIC)
        INC     R0
        STORE   R0, (ADEIC)

        LOADL   R0, %H80
        WRST    R0            ; GIE=1, ILVL=0

        LOADL   R0, %HB7      ; 0.0.1.0.11.0111 ADCH=7, ADM=11, ADST=1, ADWR=0
        STORE   R0, (ADCON) ; pokreni pretvorbu

        IDLE
```

(14 naredbi)

# Prekidni potprogram

---

```
CONVOVER    `ORG    10
            PUSH    R0                ; pohrani kontekst
            PUSH    R1
            RDST    R0
            PUSH    R0

            LOAD    R1, (ADDAT)       ; procitaj rezultat
            MOVE    R0, R1            ; spremi mjerenje u R0
            AND     %HF000, R0, R0     ; maskiraj bitove kanala

            ROTR    %D12, R0, R0      ; izracunaj pomak u tablici

            AND     %H3FF, R1, R1     ; maskiraj korisne bitove podatka
            STORE   R1, (R0+TABLE)    ; spremi u tablicu

            POP     R0                ; vrati kontekst
            WRST    R0
            POP     R1
            POP     R0
            RETI     ; povratak iz potprograma
```

(15 naredbi)

# Prekidni potprogram za obradu greške

---

```
        `ORG      20
CONVE    PUSH     R0          ; pohrani kontekst
          RDST     R0
          PUSH     R0

          LOAD     R0, (ERRCNT) ; učitaj brojac
          INC      R0          ; povecaj
          STORE    R0, (ERRCNT) ; spremi brojac gresaka

          POP      R0          ; obnovi kontekst
          WRST     R0
          POP      R0
          RETI     ; povratak iz prekida
```

(10 naredbi)

# Zaključak

---

- Inicijalizacija – 14 naredbi
- Prekidni potprogram – 15 naredbi
- Prekidni potprogram za obradu greške – 10 naredbi
  
- Ukupno:  $14+15+10 = 39$  naredbi
- Zauzeće memorije:
  - ROM: 30%
  - RAM: 7+1 lokacija + stog