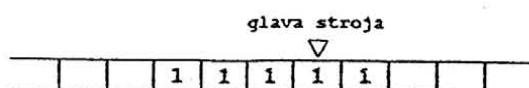


Arhitektura i organizacija računala
Prva kontrolna zadaća

- ✓ 1) (4) Nacrtati i formalno definirati Turingov stroj. Napisati program za Turingov stroj koji će na traci zapisan binarni broj proizvoljne duljine povećati za 2. Početni položaj glave stroja je negdje unutar broja. Nacrtati 5. konfiguraciju stroja uz početnu (nultu) konfiguraciju prikazanu na slici:



- ✓ 1) (4) Nacrtati stanje na sabirnici računala izgrađenog na temelju pojednostavljenog modela mikroprocesora za slijedeći programski odsječak:

| adresiranje | operacija | sadržaj | |
|-------------|-----------|--|-----|
| 100H | LDA M | ; napuni akumulator sadržajem memorijske lokacije M (apsolutna adresa), op. kod je 7FH | |
| 103H | COM A | ; komplementiraj sadržaj akumulatora, op. kod je 30H | |
| 104H | JZ NEXT | ; skoči ako je sadržaj akumulatora jednak nuli, koristi se absolutna 16-bitovna adresa, op. kod je 55H | |
| 1F0H | NEXT | NOP ; (no operation), op. kod je 01H | |
| 200H | M | FFH | ... |

Odvrediti sadržaje registara na kraju izvođenja programskega odsječka.

- ✓ 3) (3) Za jednostavni model SRISC procesora objasniti instrukciju **bit1** i pokazati kako će se ona mogla upotrijebiti za pozivanje potprograma.

- ✓ 4) (2) Objasniti i nacrtati Flynnovu klasifikaciju tipova arhitekture za slučaj kad su komponente CU (upravljačka jedinica), ME (memorijska jedinica) i PE (procesni element).

- ✓ 5) (4) Zadatak je slijedeći program s rekursivnim potprogramom:

```
BEGIN
    ...
    CALL SUB
    ...
    SUB
    ...
    CALL SUB
    ...
    RET
    ...
```

U rekursivni potprogram treba dodati kontrolni odsječak tako da dubina rekurzije bude 4. Kontrolni odsječak neka koristi globalnu varijablu N. Nadalje treba u programu označiti povratne adrese i nacrtati stanje stoga za svaki poziv i povratak iz potprograma.

TEORIJA

- 6) (3) Usporediti format instrukcija za pojednostavljeni model procesora s formatom instrukcija von Neumannovog računala (uzeti u obzir duljinu riječi od 40 bitova).

Zadatak sastavio: prof. dr. sc. S. Ribić

Samo je jedan Mali Ivica!



7.12.1999

1) ...B|B|1|1|1|1|B|...

ALGORITAM:

- glava ide do kraja broja ako je 0, stavi 1 i STOP
- dode na 2. znak (težine 2¹) ako je 1, stavi 0 i idu na sljed. itd.
- ako je i. prva znak 1, nakon što postane 0, trebamo jedan blank (B) staviti u 1 i STOP

| | 0 | 1 | B | |
|----------------|-----------------------|-----------------------|-----------------------|-------------------|
| q ₀ | q ₀ , 0, R | q ₀ , 1, R | q ₁ , B, L | → nade kraj broja |
| q ₁ | q ₂ , 0, L | q ₂ , 1, L | 1, !* | → prekociti LSB |
| q ₂ | 1, !* | q ₃ , 0, L | 1, !* | → uvedi znakenuku |
| q ₃ | 1, !* | q ₃ , 0, L | 1, !* | → ako preljev... |

* ako je broj < l
onda nade na
B u q₂

1. KONF.

...B|1|1|1|1|1|B|...

$\delta(q_0, 1) = (q_0, 1, R)$

2. KONF.

...B|1|1|1|1|1|B|...

$\delta(q_0, 1) = (q_0, 1, R)$

3. KONF.

...B|1|1|1|1|1|B|...

$\delta(q_0, B) = (q_1, B, L)$

4. KONF.

...B|1|1|1|1|1|B|...

$\delta(q_1, 1) = (q_2, 1, L)$

5. KONF.

...B|1|1|1|1|1|B|...

$\delta(q_2, 1) = (q_3, 0, L)$

6.

...B|B|1|1|1|0|1|B|...

$\delta(q_3, 1) = (q_3, 0, L)$

7.

...B|B|1|1|0|0|1|B|...

$\delta(q_3, 1) = (q_3, 0, L)$

8.

...B|B|1|0|0|0|1|B|...

$\delta(q_3, B) = (q_4, 1, !)$

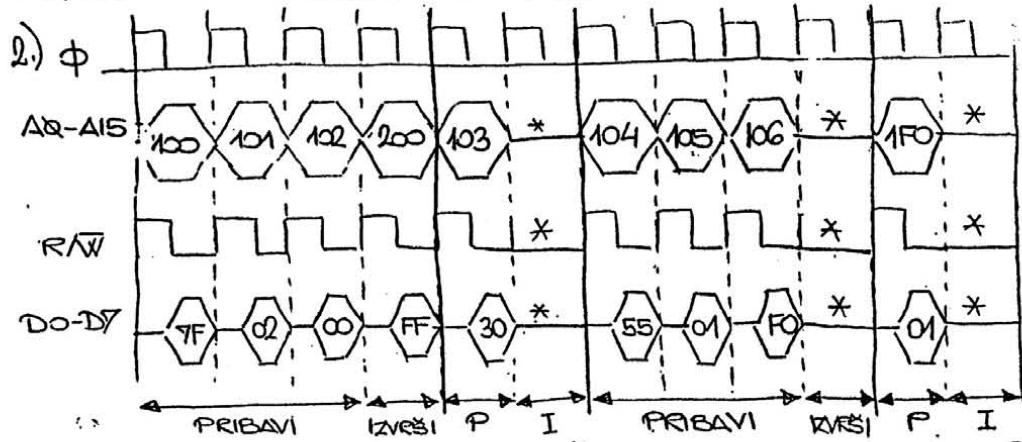
9.

...B|B|0|0|0|0|1|B|...

$\delta(q_4, 1, !)$

Konačno:

...B|1|0|0|0|0|1|B|...



100 LDA M.
(tj 100 LDA \$100)
u A=FF

103 COMA
(1 byte)
u A=00

104 JZ NEXT
(tj 104 JZ \$1FO)
u PC=1FO

| | | |
|-----|----|---------|
| 100 | 7F | LDA M |
| 101 | 02 | |
| 102 | 00 | |
| 103 | 30 | COMA |
| 104 | 55 | |
| 105 | 01 | JZ NEXT |
| 106 | F0 | |
| 1FO | 01 | NOP |

Sadržaj registara: $PC = 1F1H$
 nakon prog. isj: $A = \overline{FFH} = 00H$
 $DC = 1FOH$
 $IR = 01H$

3) btl (BRANCH & LINK): kopira PC u Linkage Reg
 (prije grananja) koji dopušta povratak iz potprogramma

U LR je sadržaj PC prije grananja, pa imamo očuvane povratne adrese.

4.) MIMD, MISD, SIMD i SISD (čista teorija...)

5.) BEGIN ...

CALL SUB

...

SUB CALL SUB

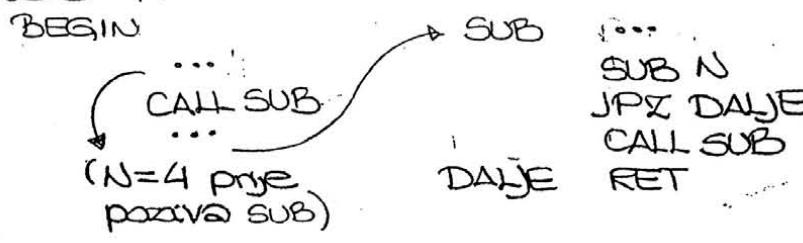
...

RET

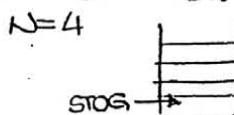
...

Budući da nije specificirano, je pretpostaviti da SP pokazuje na 1. praznu lokaciju stoga, te da povratne adrese (16 bitne) na stog idu prvo MSB, zatim LSB
 (kao zero)
 → radi jednostavnosti, stog crtam kao 16 bitnu mem., pa ovo s MSB i LSB nije važno (zaređenje)

Prepraviti rek. potprogram tako da dubina pozivke bude 4:



Kako radi:



1. poziv

$N=N-1$

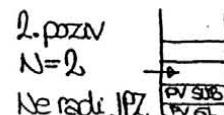
$N=3$



2. poziv

$N=2$

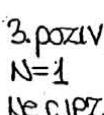
Ne radi JPZ



3. poziv

$N=1$

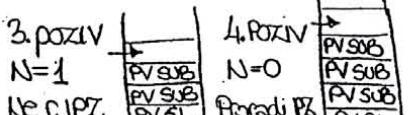
Ne r. JPZ



4. poziv

$N=0$

Proradi JPZ



Nakon što "proradi" JPZ, prestoči se daljni poziv (CALL SUB) i rekurzija je kraj. Idemo natrag:

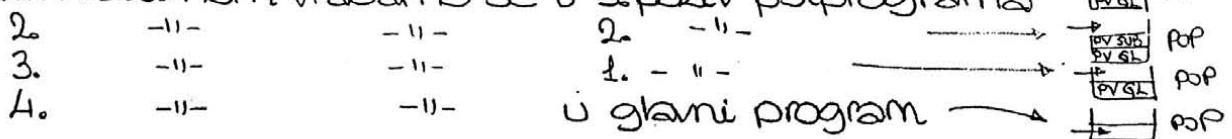
Prišim returnom vradamo se u 3. poziv potprogramma

2. -11-
 3. -11-
 4. -11-

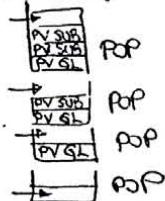
-11-

-11-

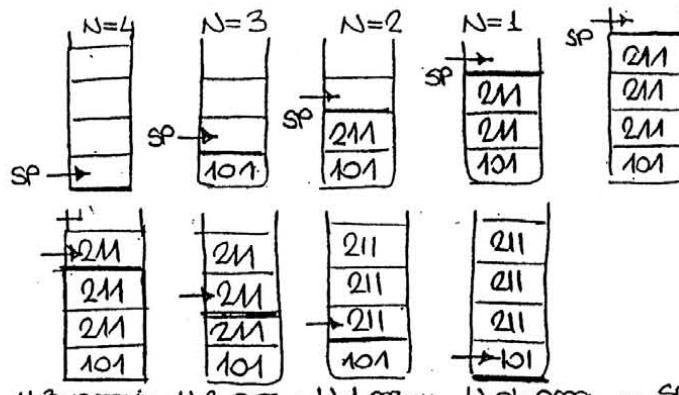
-11-



U glavni program



BEGIN
 ...
 100 CALL SUB
 101 ...
 SUB=100 ...
 210 SUB N
 JPZ DALJE
 CALL SUB
 DALJE RET



$N=0$
 Proradi JPZ DALJE

u 3. pozivu, u 2. poz... u 1. poz... u gl. prog. ... smđa?

Samo je jedan Mali Ivica!

I kontrolna zadaća iz predmeta
"ARHITEKTURA I ORGANIZACIJA RAČUNALA"

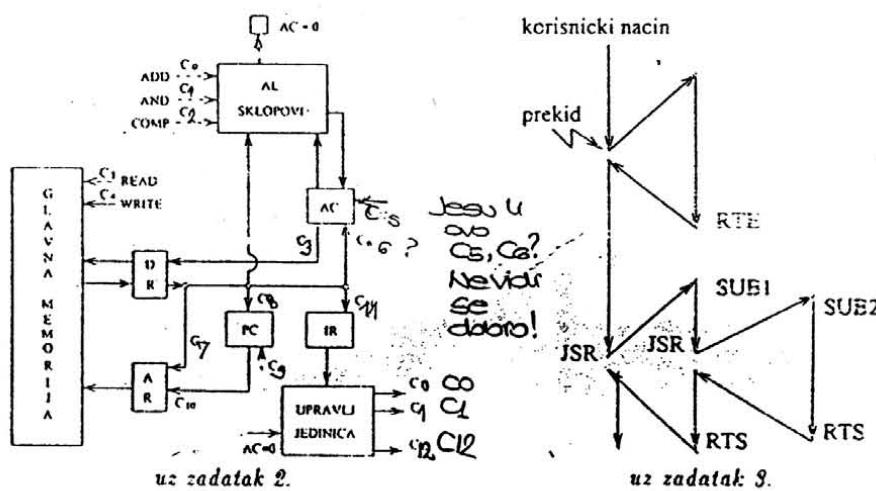
7. prosinca 1998.

- 1) (4) Definirati i objasniti Turingov stroj. Napisati program koji povećava zapisani broj na vrpci za dva. Broj je predočen u heksadekadnom brojevnom sustavu. Glava za čitanje i pisanje nalazi se inicijalno na polju koje sadrži najmanje značajnu heksadekadnu znamenku. Nacrtati, u skladu s rješenjem, treću konfiguraciju stroja (uz pretpostavku da je samo za taj slučaj zapisan heksadekadni broj $FCFF_H$).

~~TEORIJA~~ ~~2. KOLOKVIJ~~ ~~RJEŠENO NA STR.~~

- 2) (4) Nacrtati sklopovsku izvedbu upravljačke jedinice za 8-instrukcijski procesor. Odrediti strukturu kombinacijskog dijela upravljačke jedinice za fazu PRIBAVI (koja je za sve instrukcije jednaka), i za fazu IZVRŠI za instrukcije uvjetnog i bezuvjetnog grananja.

- 3) (4) Nacrtati stanja stogova za sljed događaja prikazan na slici (scenarij se odnosi na računalo temeljeno na procesoru MC 68000). Objasniti što se događa ako se, umjesto povlaštene instrukcije RTE, zabunom upotrijebi instrukcija RTS.



- 4) (4) Nacrtati i objasniti kategorije arhitekture prema Flynnnu. Skice neka se temelje na komponentama: upravljačka jedinica (CU), memorija (M) i obradbeni jedinici (P).

- kol. 5) (4) Za 8-instrukcijski model procesora realizirati dio upravljačke jedinice koji se odnosi na fazu PRIBAVI. U realizaciji upotrijebiti postupak koji se temelji na elementima za kašnjenje. Nacrtati i izvedbu elemenata za kašnjenje koji zadovoljava sljedeće zahtjeve:

- njegov izlaz mora biti točne amplitude i točnog vremena trajanja;
- mora biti sinkroniziran glavnim signalom vremenskog vođenja;
- sva kašnjenja neka su trajanja jedne periode taktnog signala.

PITATI ??

- 6) (3) Za pojednostavljeni model mikroprocesora nacrtati stanja na sabirnicama za sljedeći programski odsječak, počevši od adresu LOOP = 0100_H . Početno akumulator A sadrži podatak B_H .

| | operacijski kodovi |
|--------------|-----------------------|
| LOOP INC A | op kod INC A = $C5_H$ |
| STA A \$ 204 | op kod STA = 04_H |
| JMP LOOP | op kod JMP = 11_H |

Zadatke sastavio: prof.dr.sc. S. Ribarić

Samo je jedan Mali Ivica!

7.12.2028

Nacrtati 3. konf.

1.) hexa broj za 2

| | q ₀ | q ₁ |
|----|----------------|----------------|
| 0 | 2, 8 | 1, 8 |
| 1 | 3, ? | 2, 8 |
| 2 | 4, ? | 3, ? |
| 3 | 5, ? | 4, ? |
| 4 | 6, ? | 5, ? |
| 5 | 7, ? | 6, ? |
| 6 | 8, ? | 7, ? |
| 7 | 9, ? | 8, ? |
| 8 | A, ? | 9, ? |
| 9 | B, ? | A, ? |
| A, | C, ? | B, ? |
| B, | D, ? | C, ? |
| C, | E, ? | D, ? |
| D, | F, ? | E, ? |
| E, | 21 0, L | F, ? |
| F, | 21 1, L | 1, ? |

... A | F | C | F | F | A ...

ALGORITAM

- ako broj < E, uvedaj za 2 i stop:
- za E, F, uvedaj za 2 i idi levo:

- ⊗ → ako broj < F uvedaj za 1 i stop
- ako broj = F uvedaj za 1 i idi levo
- ako blanc (A) stavi 1 i stop

1 KONF.

2 KONF.

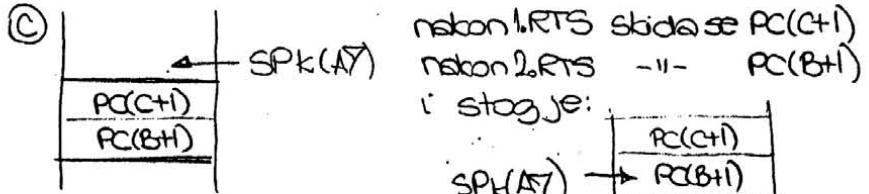
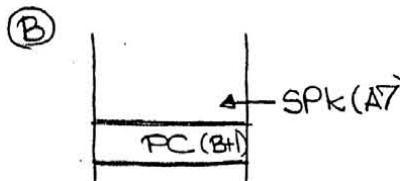
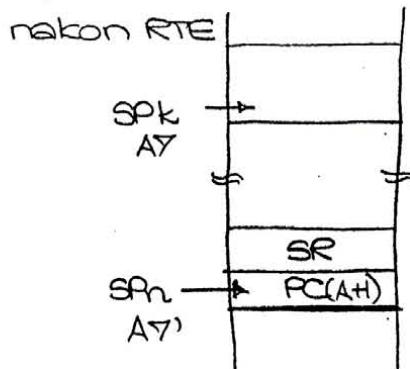
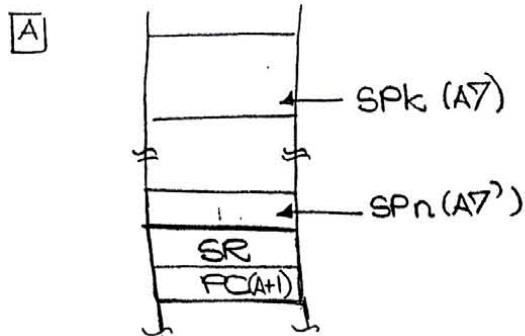
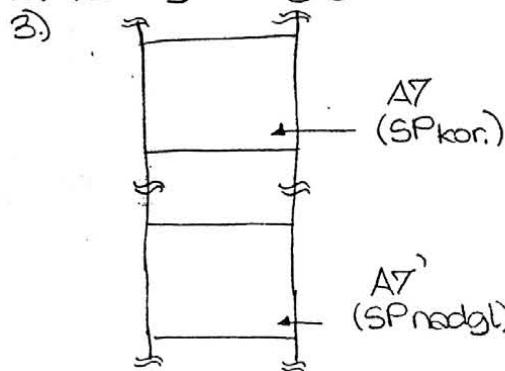
3 KONF.

Konačno:

A | F | C | F | F | A $\delta(q_0, F) = (q_1, 1, L)$ A | F | C | F | A | A $\delta(q_1, F) = (q_1, 0, L)$ A | F | C | O | 1 | A $\delta(q_1, C) = (q_2, D, ?)$ A | F | D | O | 1 | A

$$\begin{aligned} 15 + 2 = 17 \quad \lambda F + 2 = \\ = 11 \end{aligned}$$

2.) TEORIJA (knjiga)



Samo je jedan Mali Ivica!

5) Nismo radili

6) LOOP = 0100H

$$A = B_H$$

100 LOOP

...
INC A

101

STA A \$204
JMP LOOP

104

...

107

OP. KODOVI

INC A = 05H

STA = 04H

JMP = 11H

→ jednobitna instr.

→ 3 bitna instr.

→ 3 bitna instr.

A0 - A15

100 * 101 102 103 204 104 105 106 *

R/W

R R R W R R *

D0 - D7

05 * 04 02 04 C M 01 00 *

$$u \ A = C_H$$

P I

PRIBAVI I

PRIBAVI I

OPET SE ISTO IZVŠAVA,

SAMO SE SAD U A nalazi

C, a ne B; i D se spremi

na \$204 PREKO vec

wpisanoj C (C se gubi)

INC A \Rightarrow C+1=D

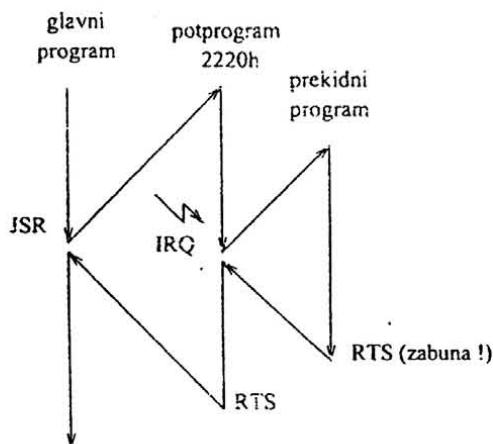
I kontrolna zadaća iz predmeta
"ARHITEKTURA I ORGANIZACIJA RAČUNALA I"

3. prosinca 1997.

- ✓ 1. (4) Definirati i objasniti Turingov stroj. Napisati program za Turingov stroj koji rješava sljedeći problem. Na vrpcu, kao početno stanje, upisan je binarni broj, čiji položaj u odnosu na glavu za čitanje i pisanje nije poznat. Stroj treba pronaći broj i uvećati ga za 2.
- ✓ 2. (4) Uslijed sklopovske pogreške adresna linija A0 pojednostavljenog modela procesora kratko je spojena s masom. Nacrtati stanja na sabirnici i odrediti sadržaje relevantnih registara tijekom izvođenja dviju instrukcija od adrese 0100_H . U A se početno nalazi 01_H . Sadržaj memorije prikazan je na slici 1.
- ✓ 3. (4) Zadano je stanje mikroprocesora MC68000: $PC=010004_H$, $SSP=A00008_H$, $USP=B00010_H$, $SR=0006_H$, i slijed događaja prikazan na slici 2. Procesor se nalazi u glavnom programu u korisničkom načinu rada. Nacrtati stanja stogova za slijed događaja prikazan na slici 2. Podatke koji nedostaju označiti simbolički. Obratiti pozornost na to da je programer zabunom završio prekidni potprogram naredbom RTS.

| | ... | operacijski kodovi: |
|------|-----|----------------------------|
| 0100 | 03 | 02 = op kod DEC A |
| 0101 | 02 | 03 = op kod LDA < adr_16 > |
| 0102 | 04 | 04 = op kod STA < adr_16 > |
| 0103 | 7C | 05 = op kod INC A |
| 0104 | 04 | 7C = op kod CCM A |
| 0105 | 05 | |
| | ... | |
| 0204 | 27 | |
| | ... | |
| 0304 | 02 | |
| | ... | |
| 0404 | 03 | |
| 0405 | 04 | |
| | ... | |

Slika 1. Uz zadatak 2.



Slika 2. Slijed događaja uz 3. zadatak.

- 4.(4) Nacrtati realizaciju jednostavnog memorijskog modula veličine 4×4 bita, korištenjem binarnih ćelija u skladu s von Neumannovim modelom. Nacrtati unutrašnju strukturu binarne ćelije. Označiti slijed upravljačkih signala koji odgovara operaciji pisanja.

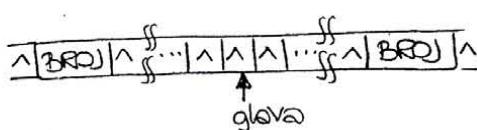
TEORIJA

5. (4) Ukratko objasniti Flynnovu klasifikaciju arhitekture, te nacrtati konfiguracije CU-MU-PU za svaku od kategorija. Definirati endo- i exo- arhitekture prema Dasgupti.

Zadatke sastavio: prof.dr.sc. S. Ribarić

3.12.1989.

1.) Bir. broj → njegov položaj nije poznat? Treba ga i. pronaći!

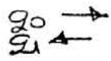


3 mogućnosti:

- (1) glava je negdje na broju
- (2) glava je levo od broja
- (3) - " - desno - " -

(1) → idu na kraj broja i počni uvećavati

(2) } ? → ALGORITAM TRAŽENJA BROJA
U (20) si → ako učitaoš ^, upiši X, predi u (21), pomak L⁴



(promj. smjer)

U (21) si → ako učitaoš X, pomak R

↓ ako učitaoš X, pomak R

↓ ako učitaoš 0, 1 NAŠAO SI BROJ → korak (1)

U (21) si → ako učitaoš ^, upiši X, predi u (20), pomak R (promj. smjer)

↓ ako učitaoš X, pomak L

↓ ako čitaoš 0, 1 NAŠAO SI BROJ → korak (1)

ALGORITAM UVEĆANJA (POZNAT) → ROK 7.12.1989. 8

| | 0 | 1 | ^ | X | | |
|-----|-----------|--------|--------|--------|-----------------|-------------|
| (2) | 20 22,0,R | 22,1,R | 21,X,L | 20,X,R | | ^ ^ ^ ^ 10 |
| (3) | 21 22,0,R | 22,1,R | 20,X,R | 21,X,L | | ^ ^ X X 110 |
| (1) | 22 22,0,R | 22,1,R | 23,^,L | 23,X,L | + našao kraj br | ^ X X X 110 |
| | 23 24,0,L | 24,1,L | - | - | → prekosi LSB | X X X X 110 |
| | 24 .,1,! | 25,0,L | .,1,! | .,1,! | → br < 2 | 1 0 1 X X |
| | 25 1,! | 25,0,L | .,1,! | .,1,! | | 1 1 1 |

2.) A0 je uvijek 0! adrese su uvijek PARNE

$$PC = 100_4$$

$$A = 01_4$$

100 03 → LDA \$204

$$PC = 100$$

D₀-D₇ → 03 → IR

101 02 → STA \$7004

PC = PC + 1 = 101, ali na A₆-A₁₅ se "pošalje" 100 jer A₀ = 0

102 04 →

$$PC = 101$$

103 0C →

D₀-D₇ → 03 → DC = 03 --

104 04 →

$$PC = PC + 1 = 102$$

105 05 →

PC = 102 na A₆-A₁₅ = 102

...

D₀-D₇ → 03 → DC = 0304

204 07 →

Nešto što je na adresi \$0304 spremi se u A (A = 02)

...

PC = 103 na A₆-A₁₅ = 102

304 02 →

Si 102 se dekodira 04 → STA

...

PC = 104 → A₆-A₁₅ = 104

404 03 →

DC = 04 --

405 04 →

PC = 105 → A₆-A₁₅ = 104

...

DC = 0404

...

A₆-A₁₅ = 404 na \$404 se

...

sprema A = 02

...

Samo je jedan Mali Ivica!

3.) MC 68000

$$PC = 010004H$$

$SSP = A00008H$ → sistem stackpointer

$USP = B00010H$ → user stack pointer

$$SR = \underline{\underline{0006H}}$$

$$PC = 00010004H \quad (32b)$$

$$SSP = 00A00008H \quad (32b)$$

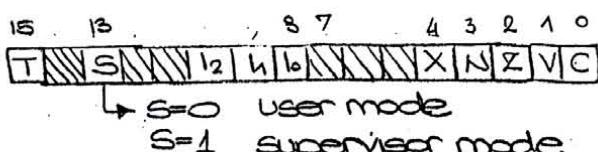
$$USP = 00B00010H \quad (32b)$$

$$SR = 0006 \quad (16b)$$

status register u MC 68000 je 16-bitovni register

KORISNIČKI DIO : SR₇ do SR₀

NADGLEDNI DIO : SR₁₅ do SR₈



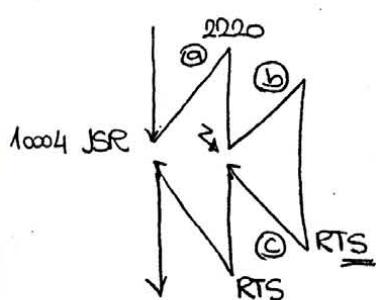
$$SR = 0006H \rightarrow S=0 \text{ (USER MODE)}$$

- X (extend)
- N (negative)
- Z (zero)
- V (overflow)
- C (carry)

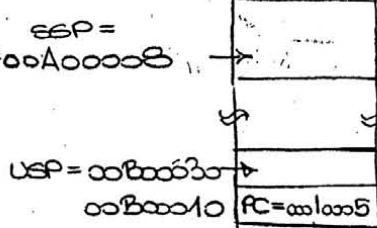
12 bits prek.maska
T trace mode

$$32 = 2^5 = 40H$$

$$\underline{\underline{00100000}}$$



① Nakon JSR \$2000,



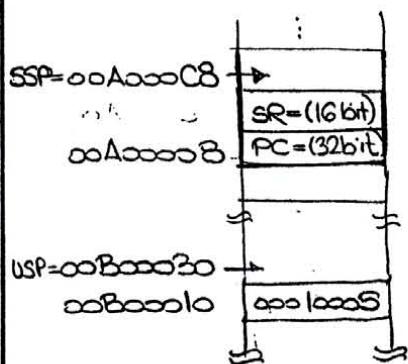
$$USP = USP + 4 \text{ bytes}$$

$$USP = 00B00030$$

SSP = SSP (ne mijenja se)

USP pokazuje na povratnu adresu u gl. program
(na adresi 10004 je JSR
pa je PC nakon uvedanja 10005)

② Nakon zahtjeva za prekid, SSP se uvedava za 6 byteva
(4 bytea PC i 2 bytea SR koji ima S=0, ali ostale bitove ne
znamo, možda su promijenjeni tokom izvođenja instrukcija)



USP = USP (ne mijenja se)

$$SSP = SSP + 6 \text{ bytes}$$

$$SSP = 00A000C8H$$

$$6 \cdot 32 = 3 \cdot 2 \cdot 2^5 = 12 \cdot 2^4 = C \cdot 2^4 = COH$$

Inače,
memorija je bajtno raspoređena,
pa npr. PC učinimo 4 memo. lokacije, (1.150)
a SR - " 2 - " - " - "

(ja sam malo pojednostavljeno pokazala)
Inače se to bavi i ne smije tako... hm :)

③ Budući da smo u sistem (supervisor) modu, dozvoljeno je koristiti i povlaštene i nepovlaštene instr., pa inst. RTS nade inicirati iznimku, ved ve se sa SSP greshkom "skinuti" 4 byte (umjesto 6) i to staviti u PC register.

REZULTAT: nepoznat nastavak izvođenja instr.

(nikad nedemo dodati natrag (osim slučajno)

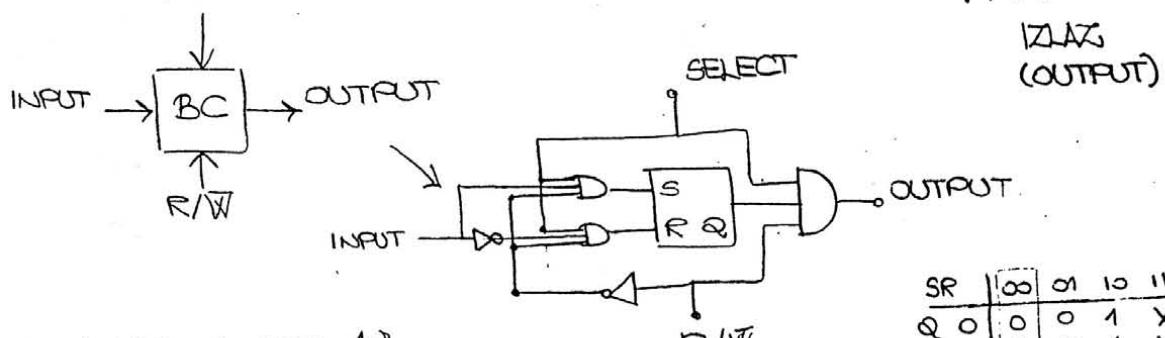
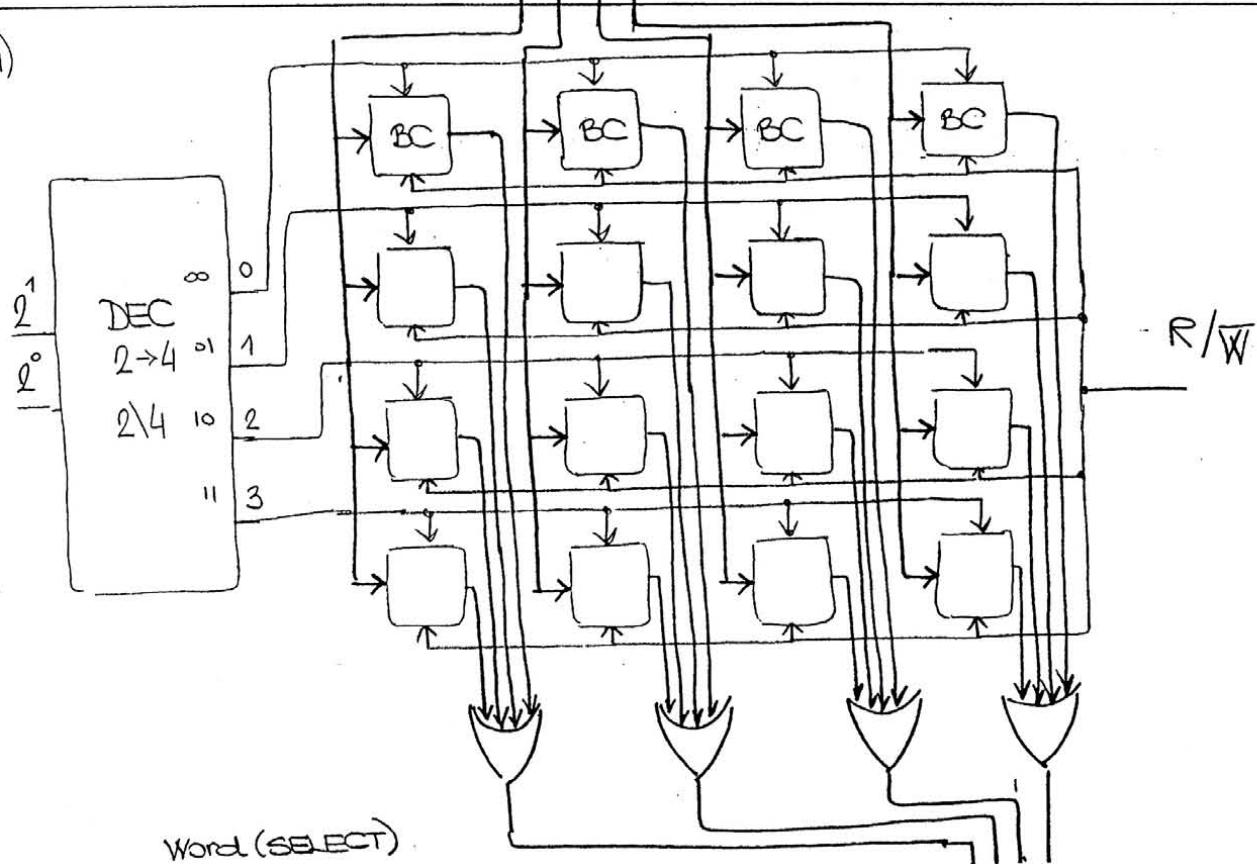
do potprogramma kojem smo napustili pri
prekidu, a tako ni do glavnog programa)

$$SSP = SSP - 4 \text{ bytes} = \underline{\underline{00A000A8}}$$

$$USP = USP$$

A)

ULAZ (INPUT)



PISANJE : $\text{SELECT} = 1$
 $\text{R}/\bar{\text{W}} = 0$

Prestpost. $Q = 0$ (odn 1)
 $\text{INPUT} = 1$ (zavim upisati 1)

I sklop (S) $S = (\text{SEL}) \wedge (\overline{\text{R}}/\bar{\text{W}}) \wedge (\text{INPUT}) = 1 \wedge 1 \wedge 1 = 1$ set postavljaju
 . I - " " (R) $S = (\text{SEL}) \wedge (\overline{\text{R}}/\bar{\text{W}}) \wedge (\overline{\text{INPUT}}) = 1 \wedge 1 \wedge 0 = 0$ Q u 1

| SR | 100 | 01 | 10 | 11 |
|----|-----|----|----|----|
| Q | 0 | 0 | 0 | X |
| R | 1 | 1 | 0 | X |

Samo je jedan Mali Ivica!

2. KOLOKVIJ

7)

MEMORIJA S PREKLAPANJEM; PRIRUČNA MEMORIJA

Brzina CPU : memory = 10:1

① LATENTNOST MEMORIJE (Memory latency) T

$$\rightarrow \text{zahvat memorije} \quad (t_1) \quad T = t_2 - t_1$$

$$\rightarrow \text{prihvatanje podataka iz memorije} \quad (t_2)$$

Utjecaj:

ARHITEKTURNI FAKTORI: istodobni pristup memorij modulu. (veće konenice)

TEHNOLOŠKI FAKTORI: skupoda brze gl. memorije

② KAPACITET MEMORIJE (\uparrow =skupina) primarna (reda M byte) sekundarna (reda G byte)

• MEMORIJSKA HIJERARHIJA:

- 1) Lokalna memorija ($t=t_{CPU}$) $8 \div 64$ registra (Risc > 100) 512 byte
- 2) Priručna -||- (-||-) $4 \div 64$ kbyta (Icache, Dcache)
- 3) Glavna, radna, primarna memorija (reda M byte); sponja
- A) Sekundarna memorija (reda G byte); još sponja [disc]

• TEHNIKA PREKLAPANJA

• GL. MEMO \div CACHE MEMO

• GL. -||- \div SEKUND. -||- [koncept VIRTUALNE MEMORIJE]

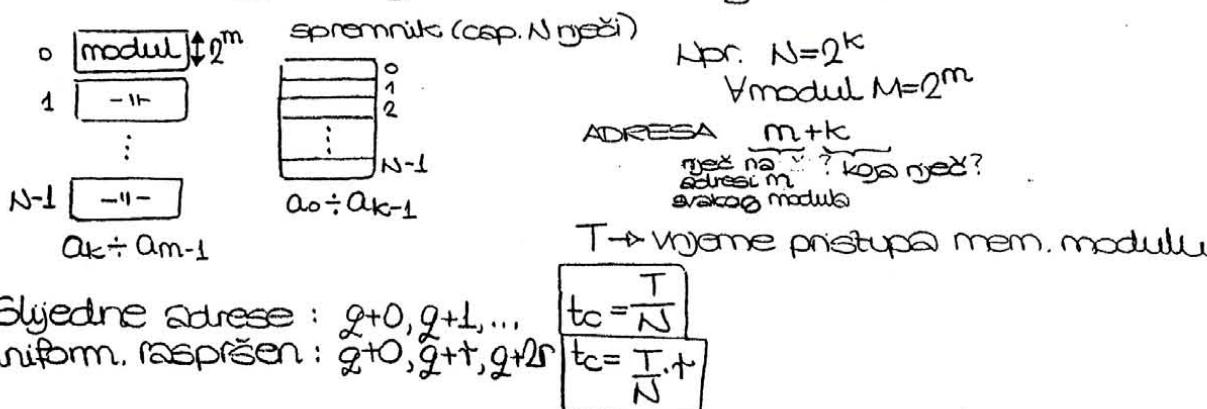
Cilj: što brža memo; što veći kapacitet

a) Memo s preklapanjem: (INTERLEAVING)

\rightarrow s N-terostrukim preklapanjem:

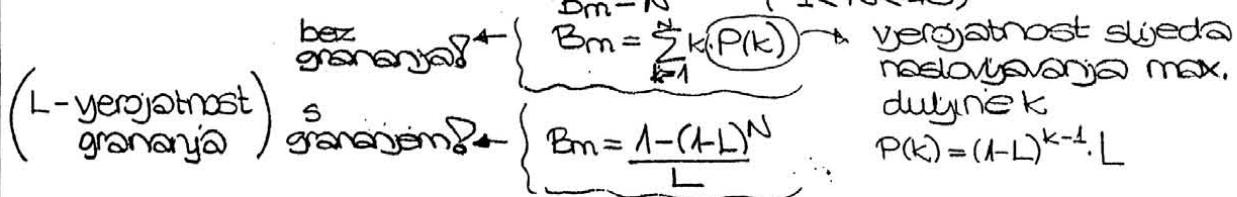
N memo. modula:

N sljедnih adresa se jednolikom razdijeljuju po modulima
podatak i je u M_j modulu ako $j = i \bmod N$



Memorijска propusnost: sred.vrijed. broja modula kojima se (MEMORY BANDWIDTH) B_m pristupi u A memo. oktisu

$$B_m = N^{0.56} \quad (1 \leq N \leq 45)$$



b) Prikupljena memorija $4 \div 64$ kByte $V_c : V_{RAM} = 5 : 1$

LOKALNA SVOJSTVA PROGRAMA: $W(t, h)$

- ✓ VREMENSKA LOKALNOST: program u bliskoj budućnosti naslovjava one objekte koje je naslovio u blist. prošlosti
- ✓ PROSTORNA LOKALNOST: program će nasloviti one koje imaju adrese bliske onim ved naslovljavanjem.

$W(t, h) \rightarrow$ radni set (Working SET)

skup mem. lok. ili blokova koji su u vrem. t naslovljani u posljednjih h pozivanja.

ORGANIZACIJA GL. MEMORIJE: 2^M sljедnih nječi u mem.
 2^W - - - - - u A bloku memorije

$$B_M = 2^M \cdot 2^W = 2^{M+W}$$
 blokova

PRIKVĀNA MEMORIJA: 2^W sljednih nječi u A bloku.
Bp blokova (Bp značak 8)

$$2^P = Bp \cdot 2^W$$
 sljednih nječi u pri. memoriji

BLOČNI PRIKVĀUČAK: blok + adresna značka

µP generira LOGIČKU ADRESU (ako je virtualni adr. mehanizam)
Ona se \Rightarrow u FIZIČKU - - - (pomoći adresnog translacijskog - - -)
↳ dio za blok (m) } modul za adresno
↳ dio za nječ u bloku (k) } preslikavanje
uspostavlja značke (F.A.)

Pogodak (HIT) \rightarrow uspostavljeno uspoređivanje

Promasaj (MISS) \rightarrow ne - - - - -

FUNJENJE - KROZ: tražena nječ se proslijedjuje odmah, ne čeka se prebacaj cijelog bloka u CACHE

k - broj naslovljavanja:

$$\frac{1}{k} \rightarrow \text{gl. mem. } \left\{ \begin{array}{l} (\text{hit}) \\ (\text{ratio}) \end{array} \right\} \Rightarrow h = \frac{\text{br. nasl. s pogotkom}}{\text{br. ukupn. nasl.}} = \frac{k-1}{k}$$

$$\frac{k-1}{k} \rightarrow \text{CACHE } \left\{ \begin{array}{l} (miss) \\ (\text{ratio}) \end{array} \right\} \Rightarrow 1-h \quad \begin{array}{c} \text{pristup CACHE-u} \\ \uparrow \end{array} \quad \begin{array}{c} \text{pristup gl. mem.} \\ \uparrow \end{array}$$

$$\text{Srednje vrijeme pristupa: } T_{ACC} = T_C \times h + (1-h) \times T_M$$

$$\text{Sr.vrij. izvršavanja instr.: } t = k_1 + k_2 \left(k_3 \cdot h + k_4 (1-h) \right)$$

\downarrow br. instr. \downarrow naslovljavanje
+ otvara se memorije + stankom

$k_3 \rightarrow$ čitanje sa stankom (hit)
 $k_4 \rightarrow$ - - - - - (miss)

OBNAVЉАЊЕ САДРЖАЈА GL. MEMORIJE

1) POHRANJIVANJE - KROZ

(uvjet: se obnavlja i gl. memory).

2) KOPIRANJE - NAZAD

(prijam bit \gg)

⊕ jednostavna rjevedba
konzistentna memorija

⊕ povezan promet
sporost (μ sek.)

⊕ dodatno stvaranje
koherencija

3) UPISIVANJE U MEĐUSPREMNIK

[inadica (1)] \Rightarrow µP predaje taj zadatci spremniku i ne čeka s obradom (izvršenje upisa?)

ORGANIZACIJA CACHE-a:

1) ROTPUNO ASOCIJATIVNA PM

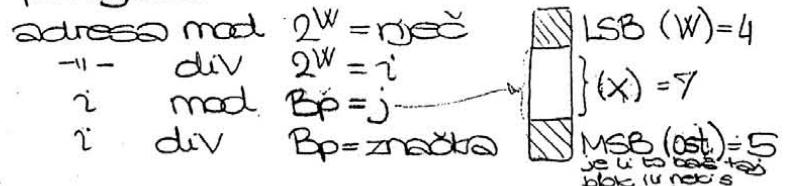
Θ skupa i složena
asocijativna pretraga

$$2^M \rightarrow \text{gl. } 2^W \rightarrow \text{kap. blok}$$

$2^{M-W} \rightarrow$ širina značke je li pod.
+ bit valjanosti → u cache =
je li, onaj u gl. mem?

2) IZRAVNA PM \rightarrow svaki blok gl. Mem. samo na određenu bločnu približak

$$\begin{aligned} 2^M &= 2^{16} \Rightarrow \text{cap} \\ 2^W &= 2^4 \\ 2^W &= 2^4 \\ 2^X &= B_p = 2^Y \end{aligned} \quad \left. \begin{array}{l} \text{cap} \\ B_M = 2^{12} \\ 2^Y \end{array} \right\} \text{CACHE}$$



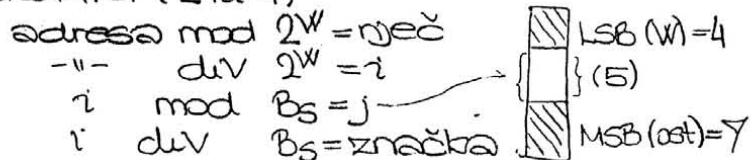
⊕ jednost. izvedba, zamjena

⊖ Vtren \Rightarrow 1 bl. pknly = 1 blok
(ako ima slobodnih \Rightarrow zamjena)

3) SKUPNA ASOCIJATIVNA PM

→ BROJ BL. PTKLJ. U SKUPINI (2 ili 4)

$$\Theta \cdot B_S = B_P$$



ALGORITMI ZAMJENE: \rightarrow OPT (buduća naslovljavanja najdalja)
FIFO (First In First Out)
LRU (Least Recently Used)

VEĆINA CACHE-a: Vd. bloka Cache = Vd. bl. gl. memory (vedina)
-||- ≠ -||- (SUBBLOČNA VEĆINA)

→ ne valja prevelika

(zbog vrem. lokalnosti ne povedava puno h-ratko)

NACINI PRIBAVLJANJA BLOKOVA:

1) Na zahjev

2) Pretpribavljanje: a) uvjet
(ONE LOOKAHEAD)

↪ DLE ciklusa

→ za vrijeme bespoštenog ciklusa

b) u slučaju promašaja

c) prijevno (započinje l. naslovljavanjem)

→ u već pretpribavlji. bloku

KOHERENTNOST MEMORIJE: \rightarrow vrijednost dobivena s LOAD =
↓
= -||- posjeduje STORE na tu adresu
(konzistentnost)

- problem u višekorisničkim rač. sustavima

Rješenja: STATIČKA (sprječeći nekonz. tijekom prevodenja prog.)
(FLAG) DINAMIČKA (-||- -||- -||- izvodenja -||-)

CENTRALIZIRANA ↳ + dodatna sklopovljiva za detekciju i riješenje
DECENTRALIZIRANA (posebna sabirnica...)

PRIMJER: CISC: { MC 68030 izravno presl, subbločni prenos, bitovi (4) valjanosti
Intel i486 skupna asoc. (4), nema subbl., pohr. kroz.

RISC: MIPS (na 1 chipu 2 procesora)

CRISC i860 (na chipu 1 datke i 1 cache)

8.

VIRTUALNI MEMORIJSKI SUSTAVI

- veliki zahtjevi glede kapaciteta glavne memorije
danasa: < 8 ili 16 MByte (umjereni)
- sekundarna memorija: do 8 GByte

Uporabom virtualne memorije stvara se konstrukciju povišene veličine glavne memorije jednake kapacitetu sekundarne memorije, te brzine gotovo jednake glavnoj memoriji.

| CPU | LOCAL MEMORY | CACHE MEMORY | PRIMARY MEMORY | SECONDARY MEMORY |
|----------------|--------------|--------------|-----------------------|----------------------|
| BRZINA, 1-10ns | 1ns-10ns | 5ns-50ns | 10ns-150ns | 1-20 ms |
| CAPACITET | 512 byte | 4-64 Kbyte | 128 Kbyte 64 Mbyte | 800 MByte 4 GByte |

Jednostavni, jednokorisnički rač. sustavi:

- ⇒ METODA PREKRIVANJA (OVERLAYS)
- program ⇒ nekoliko NEOVISNIH BLOKOVA (programskih i podatkovnih)
 - u gl. memoriji je samo prigr. modul koji se trenutno izvodi, i njemu potreban blok podataka (privremeno pohranjen)
 - trajno pohranjen ⇒ dio programa za premeštajne module

Višekorisnički računalni sustavi; složeni sustavi:

PREKRIVANJE NEDJELOTVORNO:

- konfliktnе situacije (razni korisnici traže iste mem. lokacije)
- zahtjev za dodjeljivanjem zajedničkih pravnih modula (uvodenje mehanizma zaštite pristupa prigr. - " -)
- veličina blokova je varijabilna

Zato se upotrebljavaju:

- ⇒ DINAMIČKO RUKOVANJE MEMORIJOM
- (⇒ RAZLIKOVANJE ADRESNIH PROSTORA (fizičkih i logičkih))
- KONCEPTI VIRTUALNE MEMORIJE

VIRTUALNA MEMORIJA: jednorazinska memorija (one-level storage)
ATLAS, 1959. (Manchester)

↳ organizacija sustava u stranice

FIZIČKI I LOGIČKI ADR. PROSTOR: međusobno neovisni?

FIZIČKA MEMORIJA: skup stvarnih, fizičkih mem. lokacija
gl. memorije za pohranu programa i podataka
(priklučena na sabirnicu računala)

FIZ. ADR. PROSTOR: skup adresa jednoznačno dodatajene f. mem.

LOG. ADR. PROSTOR: skup adresa koje upotrebljava programer, generira program ili process.

ODNOSI: L.A.P = F.A.P (uredunjni sustavi, 8bitni up) A po 64 Kbyte
L.A.P < F.A.P (ograničenja)
→ L.A.P > F.A.P

16, 32, 64-bitni μ P : šini se salarnica (krovno adresirivo 2^{24} , 2^{32} byte)
 (stvarna, fiz. memorija puno manjeg kapaciteta)

mehanizmi L.A. \Rightarrow F.A.

KONCEPT VIRTUALNE MEMORIJE: log. adr. prostor se ostvaruje
 upotreboom glavne i sekundarne memorije

ORGANIZACIJA LOG. ADR. PROSTORA

1) LINEARNI MEMORIJSKI MODEL (FLAT MODEL)

L.A.P čine slijedne adrese od 0 do $2^m - 1$ (homogeni prostor)
 (ne razlikuje se DATA od INSTRUCT.)

2.) MODEL MEMORIJE U ODSJEĆIMA (segmented model)

L.A.P \Rightarrow odsječci

Apsogječak je linearni log. prostor (2-dimenzionalni) model

PRIMJER

① 32-bitni CISC i486 (16 bitni segmenti, selektori)

* SEGMENTNI REGISTRI (pristup kôdu CS
 sadrže kazaljku na SS) 16 bitni
 * SEGMENTNI OPISNIK (smješten u tablici) 64 bitni (PRISTUP, VEL.ČINA, BAZNA ADRESA)
 (vidljivi dijelovi: nevidljivi: DS, FS, GS)

Prvi poziv: iz opisnika čitamo i skrivamo u nevidljiv dio

Vsyededi: vredna ih naslovujući segmente čiji su selektori ved u S.R.,
 pa se adresa formira iz nevidljivog dijela.

SELECTOR: LSB(2) Requested Privilege Level

(1) Table Indicator (koja tabl.: Global ili Local operaci)

(1B) 1 od 2^{13} opisnika u tablici

ADR. OPISNIKA: $\times 8$ (SHLR 3x) + osn. adresa tablice organizacija:

1) LINEARNA: svaki opisnik ima Baznu Adr. = 0 i vel. segmenta 4G

2 segm. -1- \rightarrow 1 za kôd, 2. za podatak

2) LIN. ZAŠTIĆENA: vel. segm. = stvarna vrijednost adrese

(ako se pokuša pristupiti nepost. mem. lok. \Rightarrow IZNIMKA)

3.) ODSJEĆCI: V program \rightarrow svoja tabl. segmentata, i mem. odsječke (do 6 memo.segm. jer 6 segm. reg za SELECTRE.)

ADRESNO PRESLIKAVANJE: log. adr. ph \rightarrow fiz. adr. pros. (tijekom izvođenja)

$f: L \rightarrow F \cup \phi$ $f(a) = \begin{cases} a' & (\text{adr.fiz.mem.}) \\ \phi & \text{u glavnoj (fiz.) mem.} \\ \text{virtual.} \\ \text{adress} & \begin{cases} a' & (\text{podatak s v.adr. a se ne nalazi u gl.m.}) \\ \text{MISSING-ITEM FAULT} & \end{cases} \end{cases}$

TABLICA PRESLIKAVANJA: fiz. adrese pridružene log. adresama DENNINGS: (skup registara)

a se pohranjuje u Reg. Virt. Adrese i adresira elem. tablice

Ako se podatci određeni virt. adr. a nalazi u gl.m. \Rightarrow elem. tabl. = a' (fiz.adr.)
 a ako ne \Rightarrow elem. tabl. = a' (adresa podatka u sec. mem.) u gl.m. \Rightarrow

Ako PROMAŠAJ: suspenzija procesa,

dohvat pod. iz sec. mem. u glavnu, upisati f.adr u tabl.

NELOGIČNOST:

ako a predstavlja adresu elem. tablice, a a-va ima koliko je i l. adr. pr., onda je vel. tablice = veličinu log. adr. prost.)?

RJESENJE:

dijelyenje fiz.adr. prost. i log. adr. prost. u blokove

ako su blokovi:

- stalne veličine: STRANICE } broj reg. tablica = broju blokova log. str.
 - promjenjive - " : SEGMENTI } adr. prostora

Ako se f. i l. str. prostor podijeli u blokove:

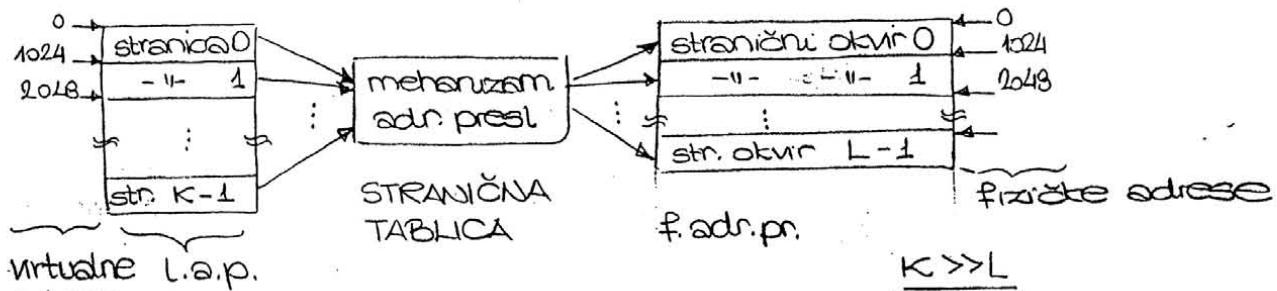
- jednake, stalne veličine \Rightarrow VIRTUALNI MEM. SUSTAV SA STRANIČENJEM
- promjenjive veličine \Rightarrow -||- MEMORIJSKI -||- -||- SEGMENTACIJOM
- kombinacija \Rightarrow V. M. S. SA STRANIČNIM SEGMENTIMA

① STRANIČENJE :

blokovi l.o.p \Rightarrow STRANICE

-11- F.a.p ⇒ STRANIČNI OKVRI (OKVRI)

] A blok 2⁹ il. 2¹⁰
slijednih lokacija



VIRTUALNA ADRESA : STRANIČNI BROJ + POMAK
 adresa elementa + određuje rječ u stranici
 stranične tablice ← reg. baze STR.TABL. (BR)

ELEM. STR. TABL. SADRŽI : POČ. FIZ. ADR. STR. OKV. + POLJE UPRAVlj. BITOVA
ili POČ. ADR. STR. U 2. MEM. (V, AR, P)

AR → READ ONLY (R) P - max. prioritet (0) } razina
 (W) : prioriteta
 (X) muri. - " - (3) }

PT(B) ADE \Rightarrow poč. zadn. str. okvira ??

Prestijskak : Ako je rečenje i razina pristupa $\leq P$

(AKO) V=1 f.Q = PT(p), ADR + pomak.

NÁD V = I + Q - P (P), ALE

NAČE (ZAKLJUČAK PRETHODNE)
IZNIMKA (POVREDA PRAVA PRISTUPA)

1. PROBLEM:

Potreblja 2. pristupa gl. mem. za vlast. stranicu tablice za podatak iz gl. mem. spajnji

РЕШЕЊЕ:

④ Tablica preslikavara izvedena u CACHE MEMORY koja sadrži samo nedavno koristene elemente str. tablice. (POTPUNO ASOCIJAT.)

b) PRETRANSLACIJA [već postojeci str. okvir + POMAK]

② PROBLEM : Stranična fragmentacija \Rightarrow zuzeđe mem. neraspoloživo za DATA

RJEŠENJE:

Asocijativno, umjesto izravno preslikavanje straničnog broja u poč. adresu straničnog okvira:

AKO IZRAVNO:

stranična tabl. ima br. elem. = br. stranica u l.a.p.? (K)

$K \gg L \Rightarrow$

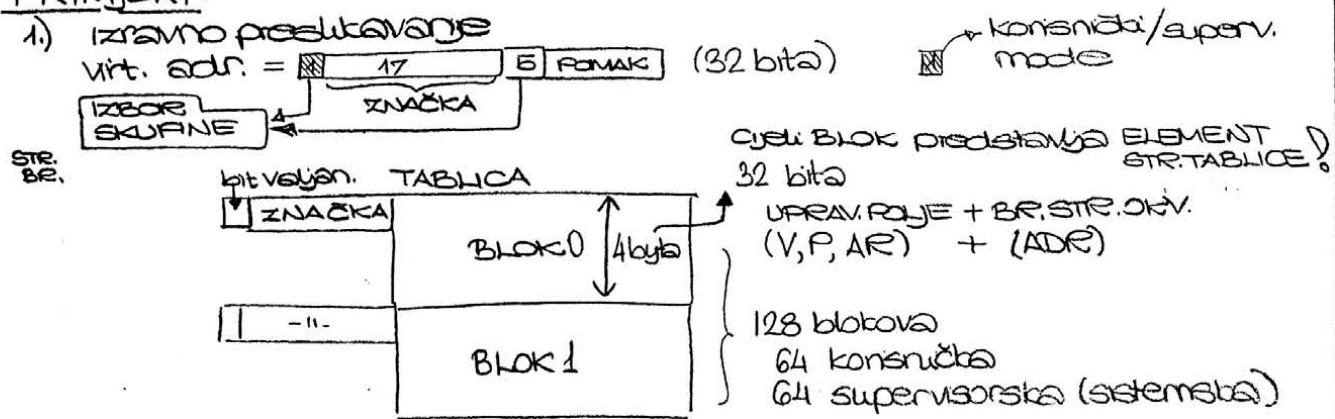
AKO ASOCIJATIVNO:

stranična tabl. ima br. elem. = br. straničnih okvira? (L)

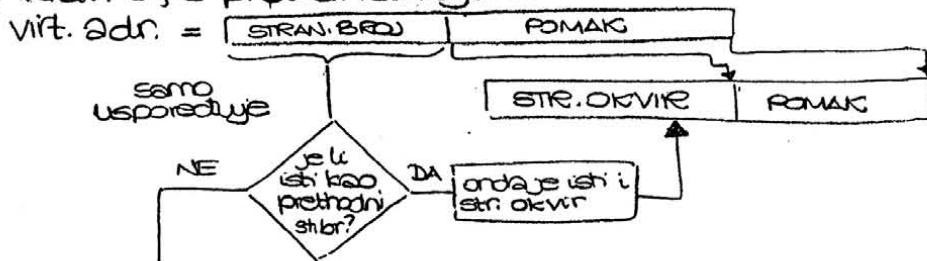
Vel. tablice je MANJA

PRIMJERI:

1) IZRAVNO PRESLIKAVANJE

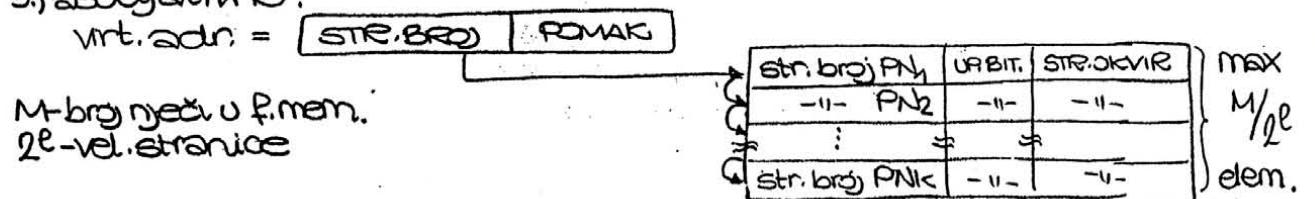


2) IZRAVNO, S PRETRANSLACIJOM:



IZRAVNO PRESLIKAVANJE: iz transl. spremn. (CACHE) ili str. tablice (memorij)
prihvati broj stran. okvira ...

3) ASOCIJATIVNO:



NAMJENSKI PRIMJERI:

⇒ DVORAZNIČKO STRANIČENJE (i860) (Intel 80386, i486)

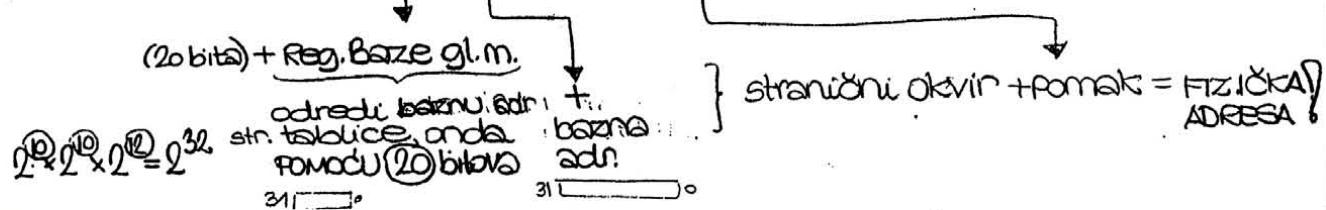
1. razina ⇒ glavna tablica (vel. 1 stranica) 4K sadrži 32 bitne elem.

sadrži adrese za 2^{10} straničnih tablica (2. razina)

2. razina ⇒ stranična tablica (vel. 1 str. / 4K) sadrži 32 bitne elem.

sadrži adrese za 2^{10} stranica (str. okvir)

VIRT. ADR : 32 bita $10 + 10 + 12 \rightarrow$ pomak

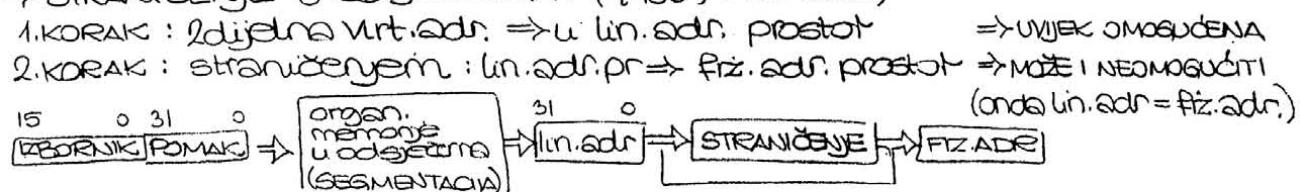


Stranične tablice mogu biti smještene:

- u memoriji (gl.) \Rightarrow problem $2 \times$ pristupa ($2 \times$ spojje)
- u translacijski spremnik, na proc. čipu i sadrži posljednje 64 korишtenе stranice, skupna ASOC. CACHE MEMORY $4 \times 16 = 64$

256 m.adr.
BS $B_p \times 4K$

\Rightarrow STRANIČENJE U ODSJEĆIMA: (i486, Intel 80386)



\Rightarrow CISC MC 68040 \Rightarrow 2 nezavisne mem. uprav. jedinice (I/O) [32bit-logadr]
- trorazinsko straničenje; ATC (Transl. spremnik Cache) [-" - fizadr]

Zašto višerazinska shema?

32bita adresa $\Rightarrow 2^{32}$ (4G) prostora (byte)

1 stranica 4 Kbyte $\Rightarrow 2^{32} / 2^{12} = 2^{20} \Rightarrow$ MILIJUN STRANICA

(1/2 fiz.)

Svaki elem. stranice 4 byte \Rightarrow STABLA 4 $\cdot 2^{20} = 4M$ byte (memoriјe)

Višeraz. memorija dopušta da elementi, potreboni za translaciju, nedostaju u tablicama bilo koje razine.

Tako se smanjuje veličina tablica (straničnih)

U gl. mem. se nalaze samo trenutno naslovjavane stranice.

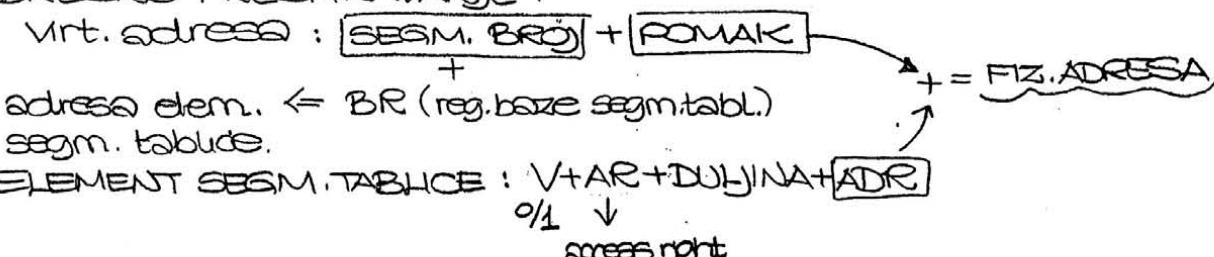
Umjesto 1 velike str. tablice \Rightarrow aktivne tablice iz pojedinih razina

② SEGMENTACIJA

\rightarrow bločno strukt. viš. programski jezici \Rightarrow prog. log. cjeline razli. veličina

\rightarrow blokovi razli. vel. \Rightarrow SEGMENTI (odnose se ravno na objekte u program. razini; vel. promjenjiva i tijekom programa)

ADRESNO PRESLIKAVANJE:



AKO $V=0$ TADA iznimka (promakšaj segmenta \Rightarrow gl. memorij.)
INACE AKO AR valjano

TADA AKO dulj < POMAK

TADA iznimka - pretek.

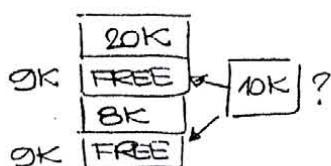
INACE F.A = ST(P).ADR + POMAK

INACE iznimka. (povreda prava pristupa)

PROBLEM:

FRAGMENTACIJA MEMORIJE \rightarrow Vagoska frag. mem.

RJEŠENJE: operacija izbijanja (sk \uparrow)
(vrem. i memor. zahjeva)



3) SEGMENTACIJA SA STRANIČENJEM

prilagođena strukturama
viših programskih jezika

↳ jednostavna sklopovska rečenica

vrline objekta

(segm. tabl.)

↳ aktivni proces P, njemu odgovarajuća ST(P) mora biti u gl. memoriji, no ne i sve PT(P) (stran. tabl.), već samo one koje pripadaju radnom skupu & ↳ indikaciju PT ⇒ bit valjanosti IS ('u elem. ST)

$$\text{VRT. ADR} = \boxed{\text{SEGMENTNI BROJ}} + \boxed{\text{STRANIČNI BROJ}} + \boxed{\text{POMAK}}$$

$$+ \quad \text{elem. ST add.} \Leftarrow \text{BR (reg. broj. ST)}$$

$$\hookrightarrow \boxed{VSTAR + DULY + \boxed{\text{ADREPRET}}} \rightarrow \text{elem. stranične (PT) VP} + \boxed{\text{ADR}}$$

(baza PT)

2 prometnja:

→ miss segmenta ($VS=0$) u gl. m. nema niže ↓ stranice traž. segm.
→ -||- stranice ($VP=0$) -||- -||- tražene str.

Vezu s višeraziničkim straničenjem?

Ako AR iz elem. ST u elem. PT }
i ako elem. ST nema podatka o DULY } \Rightarrow izračunska shema
 \Rightarrow Bitno smanjenje stranične fragmentacije: u gl. m. su samo tablice trenutno aktivnih processa, umjesto za cijelu l. adr. pr.

ZAMJENA, SMJEŠTANJE I NAČIN PRIBAVLJANJA BLOKOVA

Koja str. se vrada u sec. mem. da bi u njen okvir došla tražena str?
(Algontam zamjene) FIFO, LRU ili slučajni izbor, PFF (Page Fault Frequency)

FFF ⇒ izrada stranice, prometnja ispod def. max. vrijednosti
 $P \Rightarrow$ freq. -||- -||-

$Df \leq 1/P$ str. se ne mijenja
 $Df > 1/P$ str. se mijenja

Algontam zamjene kod kojih vel. m. bloka (br. stranica) koji sudjeluje u zamjeni se mijenja. \Rightarrow k-to naslovljavanje

Working-set Algoritam WS(k-h) skup stranica koje se pojavljuju u oknu naslovljavanja veličine h (unatrag gled.)
 $R =$ sljed. naslovly. stranica = $(4, 14, 54, 5, 56, 20, 20, 21, 4, 5, 5, 6, 20)$

$$WS(6,4) = \{4, 5\}$$

od k-h ← do k-1

DA → postavljanje

(neostvariv)

↑ (ostvareno)

VMIN ⇒ Naslovly.: indeksi str. biti naslovljene ponovo u dolaz edem oknu

SMJEŠTANJE ⇒ izbor područja

ALG. NAIJBOLJEG PRISTAJANJA (po rastućoj vrijed. \uparrow)

-||- NAJGORIĆESE -||- (po \nwarrow vrijedn.)

-||- BINARNIH DRUŠAVA

(VEĆINA)

PRIBAVLJANJE ⇒ na zahtjev; pretpribavljanje

ZAŠTITA MEMO. PROSTORA $\begin{matrix} \nearrow \text{sigurnost} \\ \searrow \text{privatnost} \end{matrix}$

→ na razini viših programskih jezika (kao što su uvez. jer mogu ugasiti sigurnost)

→ na arhitekturnoj razini (konstrukcijski i sistemski mode)

→ na razini memo. prostora: izolacija procesa (potpuna \times dyadicom), selektivni pristup

7) PREKIDNI SUSTAV PROCESORA

PREKID: neodređivani događaj koji za posjednicu ima privremeni prijenos upravljanja s tekuceg programa na program koji poslužuje prekid.

Prekid mogu izazvati:

- U/I uređaji
- vremenjski nadgledni sklopovi
- upravljelinice memorije (Memory Manipulation Unit)

IZNIMKE IZAZIVA:

- prekid
- pokusaj izvođenja privilegirane instr. u konzničkom modu
- instrukcija (TRAP # N), ili dijelyenje s nulom.
- greška na sabirnici.

Obrada iznimke: reakcija μP na zahtjeve za prekid i zamke.
(opdenitja od obrade prekida)

PREKIDNI SUSTAV:

- posebna uprav. linija (ili više njih) : PREKIDNA LINIJA
- bistabil ili registar za pohranjivanje zahtjeva za prekid
- signali INT, INTACK ...

POSLOVUJANJE PREKIDA:

- 1) μP potvrđuje prekid (INTerrupt Acknowledge)
- 2) pohranjivanje se, PC (i radnih registara)
- 3) grananje programa na prekidni program (PP)
- 4) nakon izvođenja PP obnavlja se PC, SR i nastavlja s "norm." radom

INTEL 8080 (8bitni)

- 1 prekidna linija (asink. INT)
- RST n

PROCEDURA:

① sinkronizacija asinkronog INT signala

② pohranjivanje statusa (SINC)

MEM postaje 0, što onemogućava pribavljanje ap. koda iz memorije. PC minje.

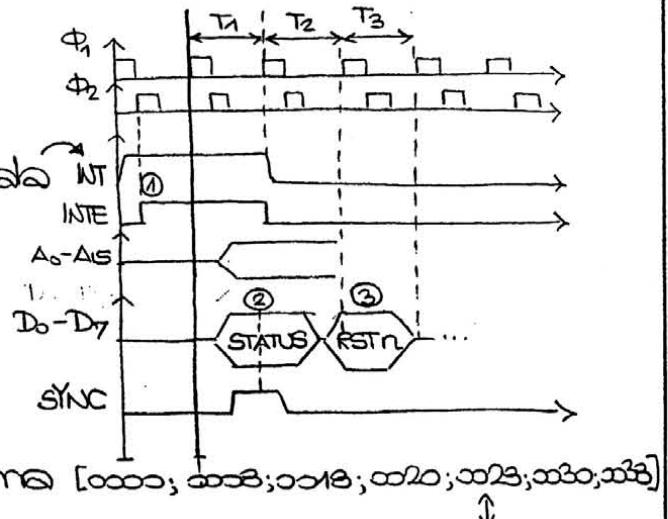
③ izvodi se RST n (koja se poziva užinum međusklopom), koja pohrani PC na stog i grana se na 1 od 8 8-bytnih potprograma [0000; 0008; 0010; 0018; 0020; 0028; 0030; 0038]

RST n: 11 xxx 111 \Rightarrow adresa potpr. (u PC ide) 0000000000000000 xxx 000

2 instrukcije upravljanja prekidom:

EI (INTE=1; omogućen prekid)

DI (INTE=0; onemogućen -||-)



RESET \rightarrow BRAŠE PC, STO i reg.
ispodnje rad od
0000H (3 ILT)

MC6800 (8bitni)

- 2 prekidne linije (asink. IRQ, NMI)

PROCEDURA:

① sinkronizacija zahtjeva i isativanje prekidne maske I u SR reg.

(ako I = 1 \rightarrow zahtjev onemogućen; inače pohrani PC, IX, A, B, CCR na stog) radni registri.

- stog se umanjuje za \overline{Y} ($PC(2), IX(2), A(1), B(1) \cup CCR(1)$)
 - ne očekuje od uzročnika prekida vektor prekida [$RST \underline{\underline{Q}}$], već ima svoju vlastitu adresu prest. vektora [$FFFF8 \cup FFFF9$] PC_H PC_L
 - RTI → obnavlja radne registre
- Maskirajući prekid se omogućava s CLI i onemogućava s SEI
Nemaskirajući \overline{Y} je uvijek omogućen (autovektor $FFFFC \cup FFFF9$)
Programski prekid (SWI) (autovektor $FFFFA \cup FFFF9$)
RESET (autovektor $FFFFE \cup FFFF9$)

Z-80 (8-bitni)

- 2 prekidiune linije (NMI [nemaskirajući] i $TINT$ [maskirajući])

PROCEDURA:

maskira se
pomoću $IFFF1$
 $\downarrow 3$ NAČINA

- onemogućava se novi prekid ($IFFF1 = Q$)

- smješta se podatak iz posljed. stolpa (ne memory!) na DATA BUS
NAČINI MASKIRAJUĆEG PREKIDA

- 1) MODE \overline{Q} : $RST n$ ($n \rightarrow 1$ od 8 prest. program)
- 2) MODE $\overline{1}$: $RST 38H$ (adresa potpuna je $0038H$)
- 3) MODE 2 : PRUE, NA poč. RADA, definiraj sadrž. IVR :

$\left\{ \begin{array}{l} LD A, \square \square H \\ LD I, A \end{array} \right.$

- onemogućava novi prekid
- pohranjuje PC ($\cup SR$) na stog

- uzročnik prekida postavlja 8 bitni VEK. BROJ [$ISB = Q$] u \square
- IVR [inicijalno se postavlja] + VEK. BROJ = adresa na kojoj je PC_L , (vjededa sadrži: $PC_H \Rightarrow$ zato VEK. BROJ mora biti paran)

Oblukuje se (IVR-om) tablica od 128 prekidiunih vektora, koja se smješta bilo gdje u memoriji

INTEL 8086 (16 bitni)

- sklopovski (vanjskom logikom \overline{Y} maskirajući) \overline{Y} nemaskirajući
- 2 VRSNE PREKIDA \overline{Y} programski (instr. INT ili, nezavino npr. DIV $\cup Q$)
- TABLICE VEKTOARA: $0000 \div 03FF$ \overline{Y} CS (kodni segment) \overline{Y} IP (instrukt. krozalo)
- ima 256 elem.: svaki sadrži 2 16-bitne adrese \overline{Y}

PROCEDURA:

- ako se prekid prihvada, nakon tekude instr. \Rightarrow pohraniti CS stog IP
- novi CS i IP se adresiraju iz tablice.

Adresa prek. krozala: oznaka prekida $\times 4$

NPR $OP = 9 \Rightarrow$ Adr. 32_{10} tj. $[0020 \text{ do } 0023]$
 $OP = 255 \Rightarrow 102_{10}$ tj. $[03FC \text{ do } 03FF]$

MASKIRAJUĆI PREKID:

- 1 linija zahtjev za pr. INT
- ako IF u se = Q , prekid dissable

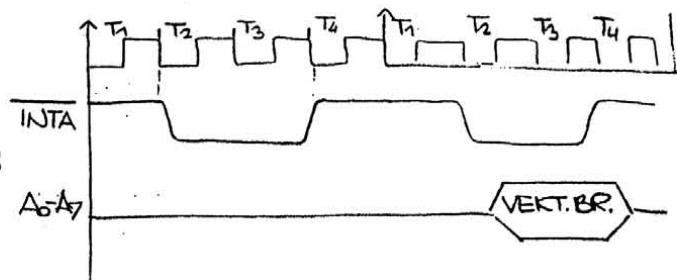
Da bi prekid bio prihvaden, treba proći 2 strjna ciklusa:

U 1: aktivirati INTA [T_2-T_4]

U 2: ponovo INTA; varijacija prekidiuna logika stavlja Izvor PREKIDA (vek. broj) na DATA BUS:

Taj broj $\times 4$ = adresa u tablici

Izvor IF i T zastavice (spojeci gnezđenje mask. pn i trade)
Pohraniti CS i IP na stog.



MC68000 (16-bitni)

OBRADA IZNIMKE: prijenos upravljanja s izvodenog u nadgledni program

→ Iznimke izazvane VANJSKIM DOGADAJIMA:

- prekid
- pogreška na sabirnicu
- reset

→ Iznimke izazvane UNUTR. STANJEM μP:

- ilegalne instrukcije; neugradene instr.
- pokušaj izvođenja privileg. instr. u USER MODEU
- pradjenje [T=1]
- djelovanje s Q; neparna adresa ... itd.

OBRADA IZNIMKE U KORACIMA:

① SR pohrani u povremeni registar

U SR postavi S \Rightarrow 1, T \Rightarrow Q

② Utvrdi vek. broj ; tj. VEKTOR IZNIMKE (ili "vanjski" u "internu")

③ Pohrani PC i SR (iz PRVKE. REG) na stog.

④ Na temelju VEK. BROJA iz TABLICE odredi novi PC //

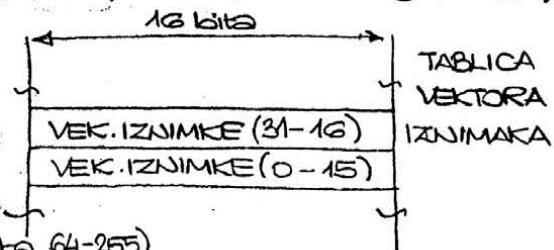
** Vektorski broj \Rightarrow "unut. iznimke": μP ga generira
"vanjske" - " " : gen. ga uzročnik prekida
(8bitni \downarrow)

Određivanje adrese elementa tablice: 1b \times 4 (xxxxxx \times xxxx)

El. tablica je adresa VETORA IZNIMKE (1. instr. prekidačeg progr.)

tj.
VEK. BROJ = $\overbrace{XXXX XXXX}^{8 \text{ bits}}$

$\times 4 \downarrow$
0000 0000 0000 00XX XXXX XX 00
0000 0000 0000 00XX XXXX XX 10



Adresa je 24 bitovna //

TABL. VETORA IZNIMAKA: (za pr. korisnika 64-255)

◦ sadrži 256 vektora [od Q do 255]

prvi 1Kbyte, tj

◦ nalazi se od adrese 000000 do 0003FF (- - - 512 ječi mem.)
(prekidi konzola od 00100)

VETOR IZNIMKE :

◦ sadrži 2 (16-bitne) ječi (osim V. IZNIMKE Q)

◦ nalazi se u nadglednoj mem. podataka, određenoj stanjem linija F00, F01 : F02.

Vek. IZNIMKE Q \Rightarrow RESET

◦ osim nove vrijednosti PC, ima i 2 dodatne ječi za SSP //

OBRADA PREKIDA: zahtjev vanjskog uređaja na 1 od 7

IPLO IP11 IP10

razina prioriteta (određene s IPLO, IP11, IP10)

| IPLO | IP11 | IP10 | MAX |
|------|------|------|-----|
| 0 | 0 | 0 | MAX |
| 1 | 1 | 0 | MIN |

\Rightarrow dopušteno grijevanje prekida!

IPLO IP11 IP10

ječi pr. Zastavice l₂, h₁, l₀ pokazuju razinu (min).

Zahtjev za prekid:

a) Aktivan: obrađuje se nakon zavr. tekuke instr.

b) Neješen: μP 1. obrađuje iznimku više razine, pa onda njega

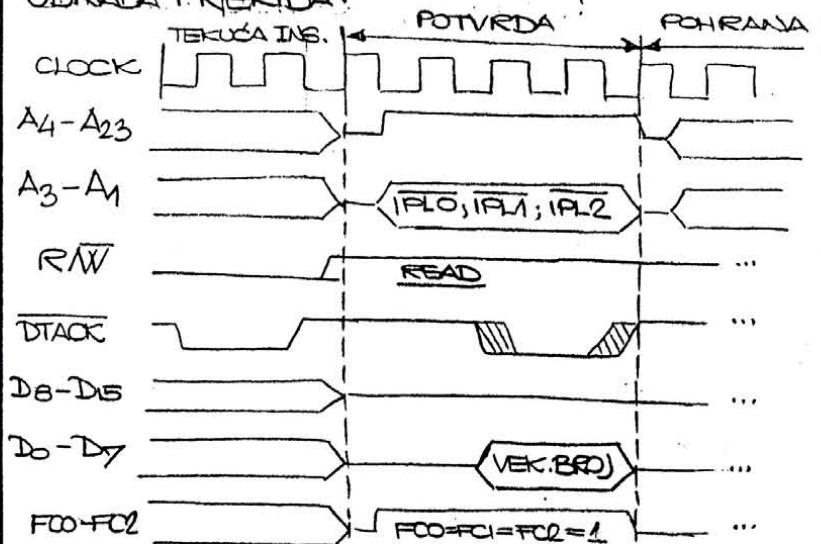
c) Onemogućen: zahtjev razine niže od one definirane s l₂ h₁ l₀

U KORACIMA:

① Isto kao kod IZNIMKE, + u SR stavi nove l₂ h₁ l₀ (odgovara razinama na prek. linijama IPLO-IP12)

Potvrda prekida: obavještava prihvadaju V.B ②

OBRADA PREKIDA:



$A_4 - A_{23}$ u potvrdi $\Rightarrow 1$
 $A_1 - A_3 \Rightarrow IPLO - IPL2$

$FC0, FC1, FC2 \Rightarrow "INTACK"$
 $D0 - D7 \Rightarrow Vekt. BROJ$

Nakon potvrde:
 $PC \leftarrow SR$, na SSP (SSP - 6)
 $PC \leftarrow Vekt. IZNIMKE$

RTE → Return from Exception : $SR \leftarrow (SSP + 2)$
 $PC \leftarrow (SSP + 4)$

TRACE → nakon V instr. → rutina za pradjenje (ISPIS RADN. RESISTARA)
 Zadatakom T (SR(15)) se upravlja samo u SISTEM MODU

- KORACI :
- ① vek. kao iznimka
 - ② μP generira Vek. BROJ, nadaje se Vek. IZNIMKA
 - ③ PC i SR na stog (SSP)
 - ④ PC ← V.I.

Pradjenje se nade dogoditi ako :

- 1) je instr. ilegalna, ili privilegirana (a u USER MODU smo)
 - 2) je instr. prekinuta RESETOM, sabir. pogr. ili pogr. adrese
- vede se desiti iznimka (vede prioriteta)

Zanimljivo:

- 1. se obradi iznimka TRACE, pa tek iznimka PREKIDA
- 1. se obradi iznimka izazvana instrukcijom koja se izvodi, pa tek onda TRACE.

VJESENTRIKE IZNIMKE:

- ① IZNIMKE NAMŠES PRIORITETA : (Razina 0) je hierarhija ...

- RESET, sabirnička pogreška, pogreška adrese
- izvršavaju se trenutno, ne "čekaju" završ. tekude instrukcije?
- μP pohraniti na stog detaljniju info. od *PC i SR (min. kont.) (jer je prekinuta tekude instrukcija)
- $\oplus PC \leftarrow PC + C$ C je vrijednost od 2 do 10 byteva

STOG : PC uveden za C

Kopja & menjati instr. koja se izvodila (za vrijeme npr. sabirničke pogr.) info. o vrsti pristupa (R,W ili ZVRŠI) zbitni funk. kod definiran u tronu iznimke

- ② (Razina 1)

TRACE, PREKID, poveća privileg. i ilegalna instr.

→ dopuštaju završetak tekude instr.

→ pohranjuju na stog min. kontext.

- ③ (Razina 2) : TRAP, DIV S & ... \Rightarrow nema hierarhije (SLJEDNO SE ZVODE INSTR.)

ZNAČAJKE RISC-a:

- 1) Izvršavanje instr. u 4 periodi taktal signalu
- 2) Sve instrukcije jednake duljine; mali broj instrukcija (≤ 15)
- 3) Pritup memoriji ostvaren samo instrukcijama LOAD, STORE
- 4) Podrška viših program. jezika (C)
- 5) mali broj načina adresiranja (≤ 4)
- 6) - - - - formata instrukcija (≤ 4)
- 7) Relativno velik skup registara opće namjene (> 32)
- 8) Upravljačka jedinica realizirana stlopovalski

PRO I CONTRA POVEĆANJA SLOŽENOSTI ARHITEKTURE

PRO

- 1) Uprogramiranje (LSI tehnol.)
 - 2) Jezgrovitost programa
 - 3) Strategija tržista
 - 4) Srodnost s prethodnicima
 - 5) Podrška viših program. jezika
 - 6) Multiprogramiranje
 - 7) Boje što manje s mem. raditi jer je skupla i spora
- Šta je 1 instr. → štoviše posla?

CONTRA

- 1) Memorija je sve brža i jeftinija
- 2) Uprogram. CPU je velika ($40-65\%$ čip-a)
- 3) Jezgrovitost se postiže pročišćavanjem skupa instrukcija
- 4) Podrška viših program. jezika bolje je što su CALL/RETURN brže, jer su one najčešće.
- 5) Statistike: 10 instr. $\rightarrow 80\%$ svih mogućih
 $21 - \text{--} \rightarrow 95\% - \text{--} - \text{--}$
 $30 - \text{--} \rightarrow 99\% - \text{--} - \text{--} - \text{--}$

RISC
10 %

ZAKJUČCI:

- 1) Složenost instrukcija ne znači i brže izvođenje programa.
Različitost duljina instrukcija i broja perioda taktal signala za njihovo izvođenje usporava protočnu strukturu RISC-a
- 2) Prevodicići viši program. jezika koristi mali podskup instrukcija
- 3) Složenje instr. \Rightarrow više vremena za oblikovanje instr. i µP
- 4) - - - arhitekture \Rightarrow podložnija pogreškama pri oblikovanju µP

MJERA PERFORMANSE RAČUNALA

- vrijeme odgovora (obavljanje kompletнog zadatka) $t_{CPU} = BT$
- - - - - CPU: konkretno vrijeme (obrada programa)
sistemsko - - - (troši ga OS-ka obavlj. zadatak prog.)
- $CPI \rightarrow$ srednji broj perioda taktal sign. po instr. $= B / IC \cdot br. instr.$
- $t_{CPU} = IC \cdot CPI \cdot T$
- propusnost ($1/\text{vrijeme odgovora}$) Million Op. Per Second MIPS, MFLOPS
- SPEC : System Performance Evaluation Cooperative
(10 osnovnih ispitnih programa (Fibonacci, Hanoi Towers...))
- ↳ srednja geometrijska vrijednost performansi za 10 isp. prog.

CISC:

- količina prometa (Mbyte) CPU-memorija

Protočni segmenti

- IF - instr. fetch
- ID - - - decoding & operand fetch
- EX - - - execution
- ME - memory access
- WB - Result Writeback

VRSTE ADRESIRANJA

- 1) UPUTNO ADR. → vrednost operanda je izravno uključena u samoj instrukciji (nema $\langle ea \rangle$) (operand neposredno zadat)
- 2) IZRAVNO ADR. → adr. operanda je u adresnom MEMORIJE polju instrukcije. $\langle ea \rangle =$ adr. redi.
- 3) IZRAVNO ADR. → adr. operanda je "ime" registra REGISTARA
- 4) POSREDNO ADRESIRANJE → adr. polje u instrukcije (INDIREKTNO) adresena je adresa lokacije na kojoj se nalazi adr. operanda
- 5) INDEKSNO ADR. → 2komponentni način
 - 1 komponenta = baza address
 - 2 - " - = adresno polje s reg. brojem, koji je u fink. indeksnog registra (sadži pomak)
- 6) BAZNO ADR. S POMAKOM
 - 1 komp = reg. broj. u fink. baza registra
 - 2 komp = konstanta
- 7) RELATIVNO ADR. S POMAKOM
baza reg. = PC
pomak = 1 byte instrukcije $ea = (PC) + R$

Zlatno pravilo uzbudjuje:

- 1) Redukciju skupa instrukcija (jednostavnih)

29.01.2001

1) A₀ je trajno u "0"

A $\Rightarrow \$0F$

\$A000 4C \Rightarrow incA
 \$A001 43 \Rightarrow comA
 \$A002 B7 } staA
 \$A003 20 } $\Rightarrow \$2001$
 \$A004 01 }

Odrediti:

- stanje na sabirnicici
- sadržaj A nakon izvršenja
- adr. i sadržaj memorije

A₀ je trajno u "0", pa je, iako PC ima i neparne vrijed. (sadržaj), na sabirnicama UVJEĆ parne adrese ??

① PC = \$A000 \rightarrow IR = 4C (DEKODER), PC = PC + 1, izvrši $\Rightarrow A = \$10$

② PC = \$A001 \rightarrow na sabirnicici je OPET \$A000 ?? -- $\Rightarrow A = \$11$

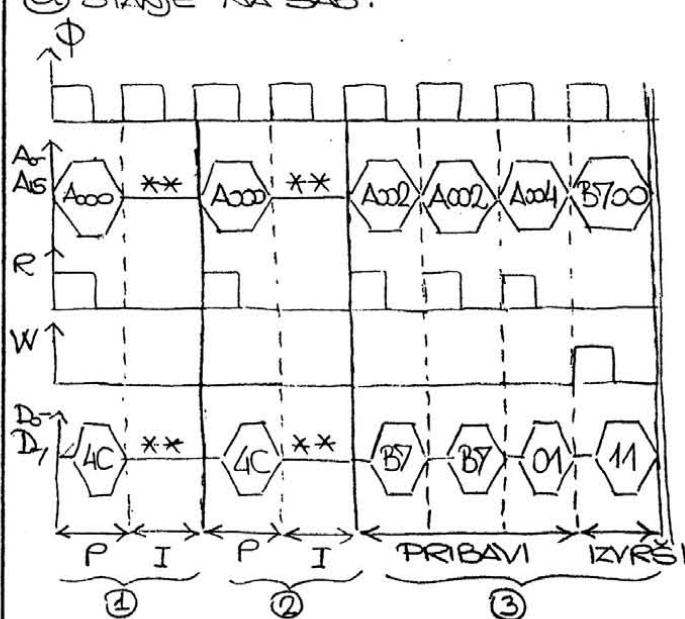
③ PC = \$A002 \rightarrow na sab \$A002 \Rightarrow IR = B7 (DEKODER) 3bocna instr.

PC = \$A003 \rightarrow na sab \$A002 u DC = B7 --

PC = \$A004 \rightarrow -- \$A004 u DC = B7 01

Izvrši: spremi se na adresu B700 (jer A₀ = "0") sadržaj akumulatora; tj. A = \$11

④ STANJE NA SAB:



⑤ sadržaj A: A = \$11

⑥ Memorija:

| | |
|--------|----|
| \$A000 | 4C |
| \$A001 | 43 |
| \$A002 | B7 |
| \$A003 | 20 |
| \$A004 | 01 |
| | ⋮ |
| \$B700 | 11 |

** High Z-CPU se odspoji od vanjske sabirnice

2) Sklop od 128 instrukacija: $2^7=128 \Rightarrow 7$ bitova za op. kod ??

Brže $\Rightarrow 8\Delta T$ } ukupno max 16 taktova: $\Phi_0 - \Phi_{15}$

Sponje $\Rightarrow 16\Delta T$

Nezav. upravl. točke $\Rightarrow 32$ (C₀-C₃₁)

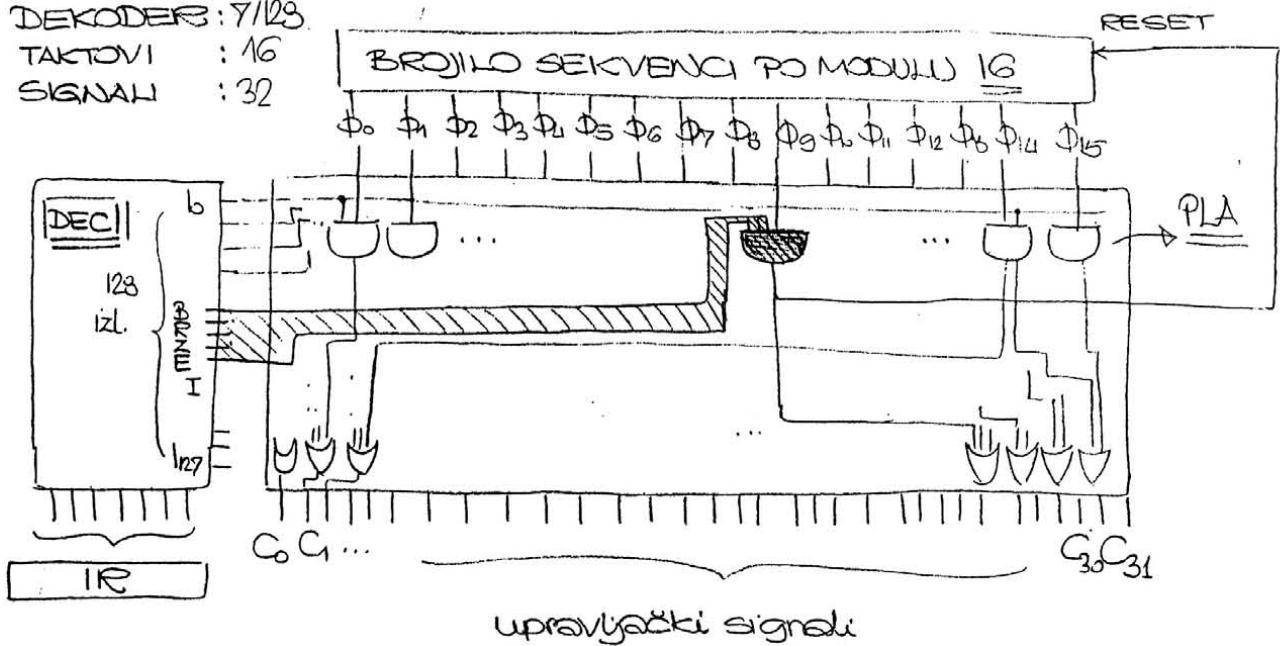
a) log. jednadžba za upravl. signale: $\left\{ C_i = \sum_{j=0}^{15} (\Phi_j \sum_{m=0}^{127} l_m) \quad i=0, \dots, 31 \right\}$

b) Shema?

Sklop. detalji za ubrzanje rada procesora kad su u pitanju brze instrukcije:

DEKODERS: Y/128

TAKT/VI : 16
SIGNALI : 32



Za uvrzanje rada:

Za svaku instr. (bazu, koja traje 8DT, a ne 16DT), njen izlaz I spajamo na I sklop od taka ϕ_0 , i izlaz tog I sklopa spajamo na RESET od BROJILA?

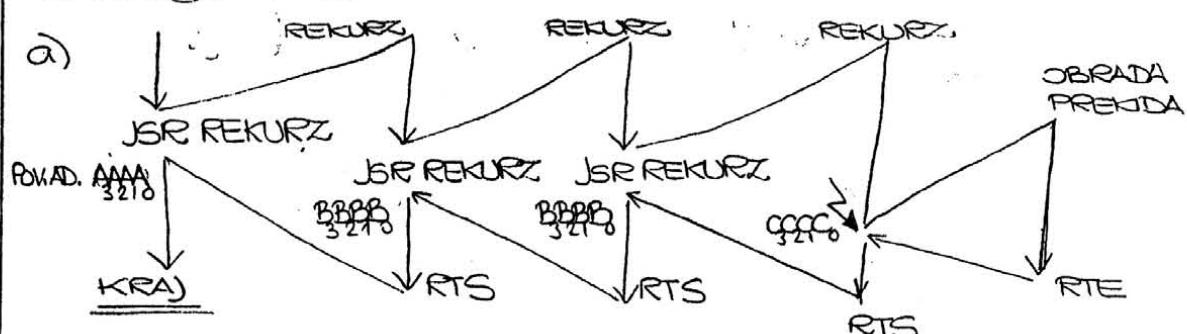
③ MC68000 \Rightarrow USER MODE ADRESA \Rightarrow 4 byte

REKURZJE $1+2=3\times$

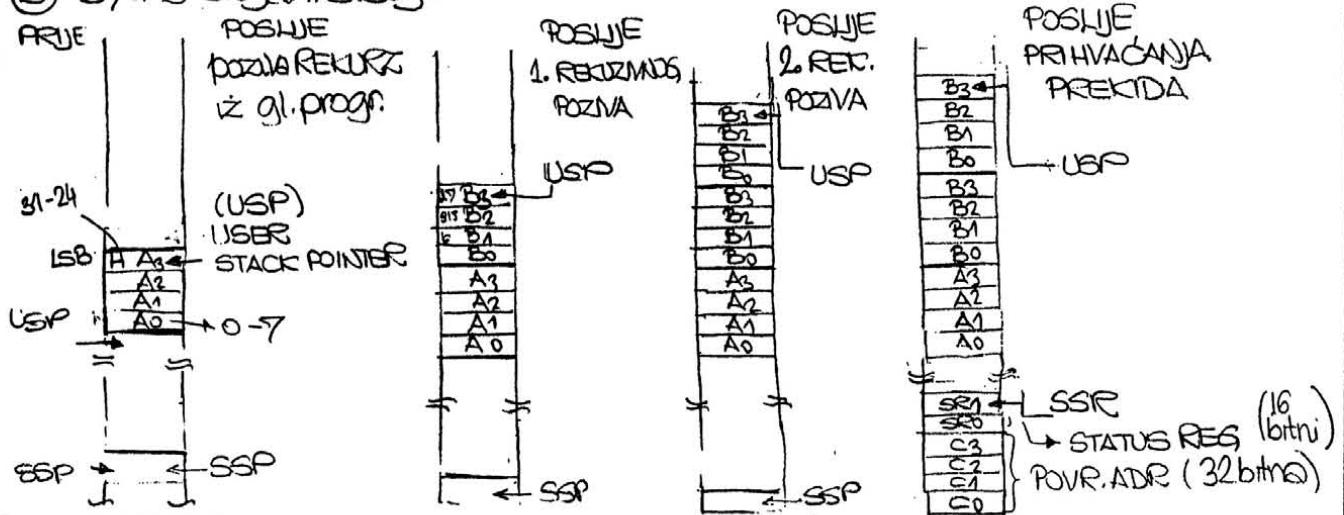
PREKID \Rightarrow SUPERVISOR MODE

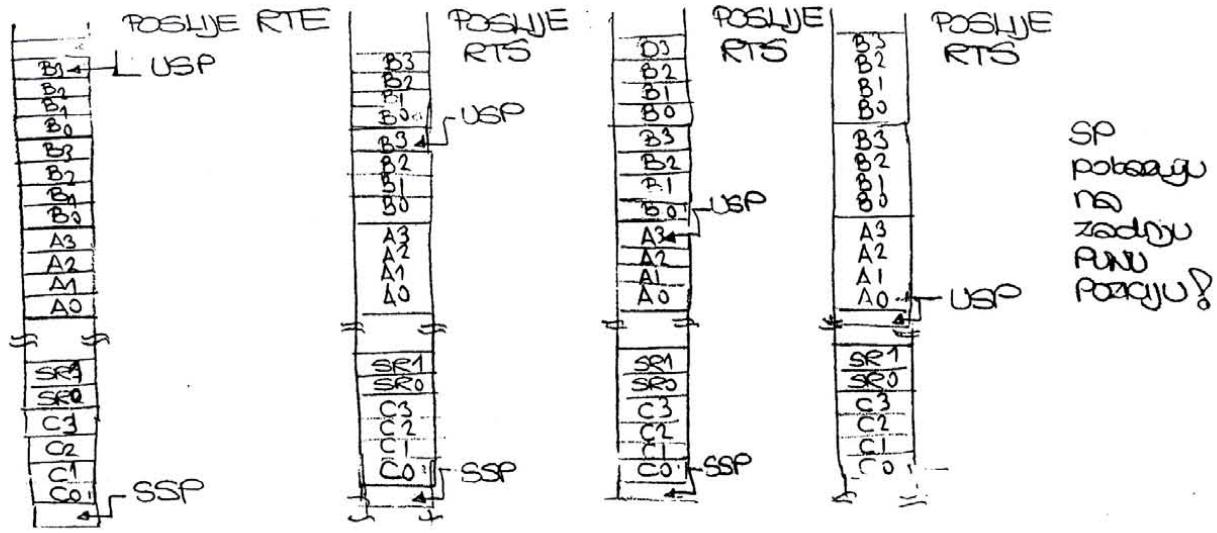
a) grafički

b) stanje stogova



b) Bytno organ.stog:





- (A) μprogramm (dj. tokca, izvršenje po fazama)
sadržaj djela μmem. za izvrši od tbma
tbma : $(B) + 32_{(10)} \rightarrow A$

$$\begin{aligned} 3l &= 7^5 \\ &\underline{00000} \quad \underline{1000000} \\ &\text{CEM} \\ 1l &= 3+4 = 2^3+2^2 \\ &\underline{1100} \end{aligned}$$

tbma :

| | |
|-----------------------------------|-------------|
| $L \leftarrow [0, F(\text{CEM})]$ | $CA = 01$ |
| $R \leftarrow B$ | $CB = 001$ |
| $C = 0$ | $COP = 00$ |
| $MB \leftarrow Q, Q \leftarrow S$ | $CHS = 00$ |
| $A \leftarrow MB$ | $CMB = 001$ |
| $H(1) \leftarrow 0$ | $CAB = 00$ |
| $H(0) \leftarrow 0$ | $CB8 = 00$ |
| nema wtj. | $CST = 00$ |

Dijagram
tbma

$$F2 = 11110010$$

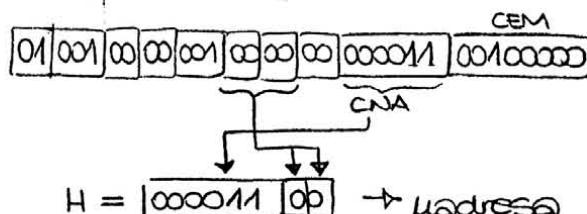
$CNA \leftarrow 000011$

$$\begin{array}{ll} /S * P(1)/ & H \leftarrow 11110010 \\ /S * P(2)/ & F \leftarrow CM(H) \end{array}$$

$$\begin{array}{ll} /CA(01) * P(0)/ & L \leftarrow [0, F(\text{CEM})] \\ /CB(001) * P(0)/ & R \leftarrow B \\ /COP(00) * P(0)/ & Cn = 0 \\ /CHS(00) * P(0)/ & MB = Q = S \\ /CMB(001) * P(0)/ & A \leftarrow MB \\ /CST(00) * P(0)/ & nema wtj. \end{array}$$

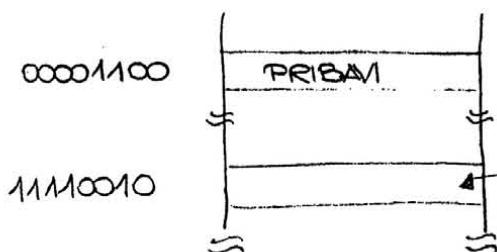
$$\begin{array}{ll} /CAB(00) * P(1)/ & H(1) \leftarrow 0 \\ /CB8(00) * P(1)/ & H(0) \leftarrow 0 \\ /S * P(1)/ & H(1-7) \leftarrow F(CNA) \\ /S * P(2)/ & F \leftarrow CM(H) \end{array}$$

μInstrukcija:



$H = \underline{000011} \quad \underline{0b} \rightarrow$ adresa
syjedade
 μI (PRIBAV)

μmemorija:

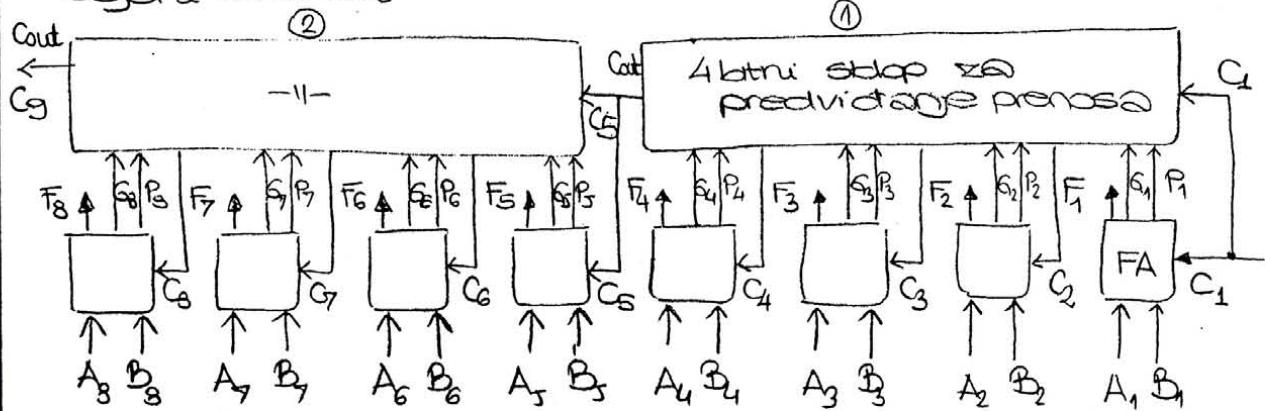


⑤ 8bitno-paral. zbrajalo

Sklop za četiri LookAhead \rightarrow komb 2 4bitna sklopa

SHEMA : LOGIČKE JEDNAŽBE

Ogromni brzinu!



$$\text{log. jednažbe: } C_{i+1} = g_i + p_i \cdot c_i$$

$$C_2 = G_1 + P_1 C_1$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 (G_1 + P_1 C_1) = G_2 + P_2 G_1 + P_2 P_1 C_1$$

$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$$

$$C_5 = G_4 + P_4 C_4 = G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 + P_4 P_3 P_2 P_1 C_1$$

$$C_6 = (\text{ista formula kao za } C_2) = G_5 + P_5 C_5$$

$$\begin{aligned} C_7 &= (\text{analogni s } C_3) \\ C_8 &= (-\text{II}-) \\ C_9 &= (-\text{II}-) \end{aligned}$$

Brzina:

$$\begin{aligned} 1. \text{ 4bitni sklop} &\Rightarrow 2 \text{taj + tun} \\ 2. \text{ 4bitni sklop} &\Rightarrow 2 \text{taj + tun} \end{aligned}$$

$$2 \cdot 2 \text{taj + (tun)} = \text{V}$$

zanim.

Da smo konstrukciju samo 1 (8bitni sklop) $\Rightarrow t = 2 \text{taj + (tun)}$ ^{zanim.}
jer ovako ipak moramo delati 2taj da se izračuna CS od 1. 4bitnog sklopa!

⑥ INKREMENT

A+1

$$\begin{cases} x = A ; C = 1 \\ y = 00..0 (S_1 S_0 = 00) \end{cases}$$

$S_2 S_1 S_0$

0 0 0

C=1

DEKREMENT

A-1

$$\begin{cases} x = A ; C = 0 \\ y = 11..1 (S_1 S_0 = 11) \end{cases}$$

0 1 1

C=0

ZBRAJANJE

A+B

$$\begin{cases} x = A ; C = 0 \\ y = B (S_1 S_0 = 01) \end{cases}$$

0 0 1

C=0

ODUZIMANJE

A+B+1

$$\begin{cases} x = A ; C = 1 \\ y = \bar{B} (S_1 S_0 = 10) \end{cases}$$

0 1 0

C=1

EX-OR

A⊕B

$$\begin{cases} x = A ; C = 0 \\ y = B (S_1 S_0 = 01) \end{cases}$$

① 0 1

C = X don't care

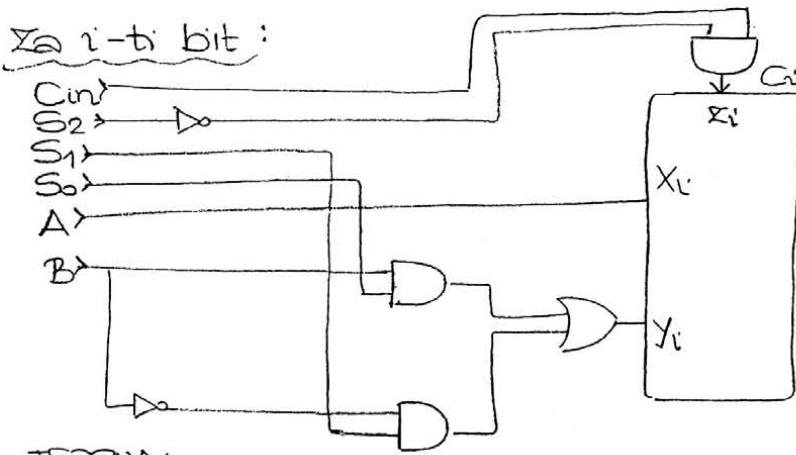
NE

\bar{A}

$$\begin{cases} x = A ; C = 0 \\ y = 11 (S_1 S_0 = 11) \end{cases}$$

① 1 1

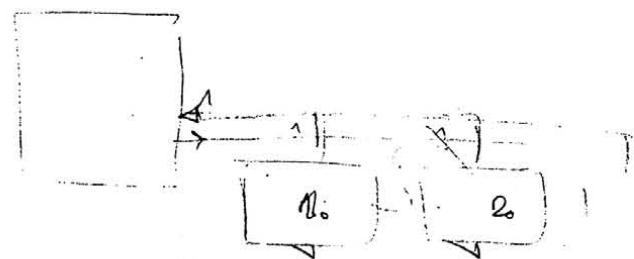
C = X



Da se tražilo
resursirati još i
log. op. I i III,
trebalo bi postaviti
komb. sklopove i
na vraz X_i ?

TEORIJA:

- ⑦ Hazardi u protičnoj strukturi
Resursni konflikti load, store (I ponstupni mem. modul)
i ječenja (ubacivanje zapornih vrata, čekanje)
- ⑧ Adresno (Denningsov) preslikavanje. Log. pogreška. Šta?
- ⑨ PREKID, obrada prekida
- ⑩ Pribučna memorija (BLOC, ZNAČKA, BLOČNI PRIKLJUČAK)



(2) CU (8-instr. μ P)

PRIJAVA : ϕ_1

ϕ_2, ϕ_3 $AR \leftarrow PC$
 ϕ_4 $DR \leftarrow READ(AR)$
 $PC \leftarrow PC + 1$
 $IR \leftarrow DR$

C_{10}
 C_3
 C_9
 C_M

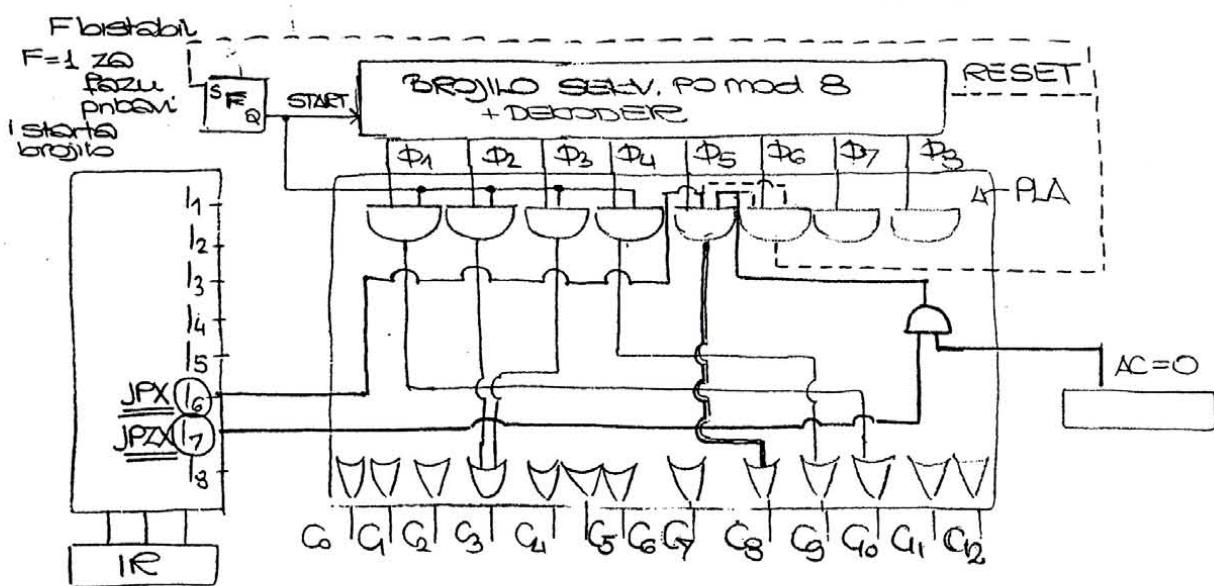
Traje
 $4\Delta T$

IZVRŠI ($JP X \rightarrow$ bezuvjetno grananje):

ϕ_5 $PC \leftarrow DR$ C_8

IZVRŠI ($JPZ X \rightarrow$ uvjetno grananje \rightarrow Jump If zero)
 ϕ_5 Ako $A=0$ $PC \leftarrow DR$ $C_8, + AC=0$

IZVRŠI
TRAJE
 $1\Delta T$



- PRIJAVA
- IZVRŠI $JP X$
- IZVRŠI $JPZ X$

Za RESETIRANJE BROJILA nekori ϕ_5 (jer je instr. ved izvršena, koristi se ϕ_6 : ----)

Arhitektura i organizacija računala

Druga kontrolna zadaća

(4) Za model mikroprogramiranog procesora napisati mikroprogram za fazu IZVRŠI strojne instrukcije TTBA (*Transfer Two's Complement B To A*). Zastavice Z i N se postavljaju ovisno o rezultatu izvođenja instrukcije. Operacijski kôd instrukcije TTBA je 01110000. Prva mikroinstrukcija za fazu PRIBAVI nalazi se na adresi 00001100. Format mikroinstrukcije prikazan je na slici 1. Potrebno je:

- nacrtati dijagram toka mikroprograma,
- prikazati tijek izvođenja mikroprograma po fazama signala vremenskog vođenja u jeziku sličnom CDL-u,
- prikazati sadržaj mikroprogramske memorije.

(3) 8-instrukcijskom modelu procesora pridodane su dvije vrlo spore instrukcije koje zahtijevaju $10\Delta t$ i dva dodatna upravljačka signala C13 i C14. Nactati sklopovsku izvedbu upravljačke jedinice i točno označiti sve parametre na koje utječe gornja preinaka.

(3) Nacrtati 3-bitovno (paralelno) potpuno zbrajalo temeljeno na poluzbrajalima i točno označiti signale potrebne za izvedbu sklopa za predviđanje bita prijenosa. Uz to prikazati preinaku ulaza zbrajala potrebnu za ostvarivanje operacija: F=A+B, F=A+B+1, F=A+B', F=A-B, F=A-1, F=A i F=A+1.

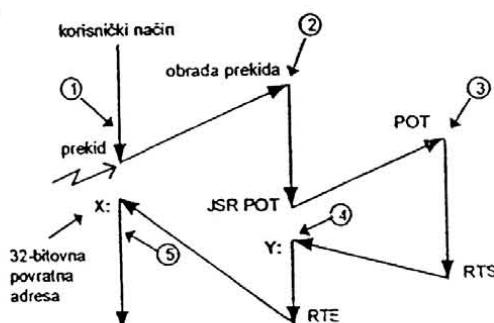
(4) Uz pretpostavku da su za svaki stupanj 4-bitovnog potpunog zbrajala raspoloživi potrebni signali, izvesti sklop za predviđanje bita prijenosa. Napisati logičke formule, nacrtati sklop i ocijeniti ubrzanje u odnosu na klasičnu izvedbu potpunog zbrajala.

(5) Opisati postupak realizacije logičke operacije I (temeljen na logičkoj operaciji ekvivalencije) koji se koristi u "standardnom" postupku oblikovanja aritmetičko-logičke jedinice.

(6) Za slijed događaja prikazan na slici 2 nacrtati stanja stogova i kazala stogova za računalo temeljeno na procesoru MC68000. Statusni registar je 16-bitovni, programsko brojilo je 32-bitovno, a memorija je bajtno organizirana. Početne vrijednosti kazala stogova su SSP=A01F, USP=010A.

| CA | CB | COP | CSH | CMB | CAB | CBB | CST | CNA | CEM |
|---|-----------------------------|-----------------------------|---|------------------|-----|-----|-----|---------------------------------|-----|
| 2 | 3 | 2 | 2 | 3 | 2 | 2 | 2 | 6 | 6 |
| CA: | | CB: | | COP: | | | | CSH: | |
| 00 nema prijenosa | | 000 nema prijenosa | | 00 suma uz C=0 | | | | 00 MB $\leftarrow Q$, Q=S | |
| 01 L(15-6, 5-0) \leftarrow 0, F(CEM) | | 001 R \leftarrow B | | 01 suma uz C=1 | | | | 01 MB $\leftarrow Q$, Q= shr S | |
| 10 L(15-10, 9-0) \leftarrow F(CEM), 0 | | 010 R \leftarrow B' | | 10 ne koristi se | | | | 10 MB $\leftarrow Q$, Q= shl S | |
| 11 L \leftarrow A | | 011 R \leftarrow PC | | 11 ne koristi se | | | | 11 MB $\leftarrow IN$ | |
| | | 100 R \leftarrow SR | | | | | | | |
| CMB: | CAB: | CBB: | CST: | | | | | | |
| 000 nema prijenosa | 00 H(1) \leftarrow 0 | 00 H(0) \leftarrow 0 | 00 nema utjecaja na SR | | | | | | |
| 001 A \leftarrow MB | 01 H(1) \leftarrow 1 | 01 H(0) \leftarrow 1 | 01 SR(15) \leftarrow ZT | | | | | | |
| 010 B \leftarrow MB | 10 H(1) \leftarrow SR(15) | 10 H(0) \leftarrow SR(14) | 10 SR(14) \leftarrow MB(15) | | | | | | |
| 011 PC \leftarrow MB | 11 H(1) \leftarrow SR(14) | 11 H(0) \leftarrow MB(15) | 11 SR(15) \leftarrow ZT; SR(14) \leftarrow MB(15) | | | | | | |
| 100 SR \leftarrow MB | | | | | | | | | |
| 101 OUT \leftarrow MB | | | | | | | | | |

Slika 1. Mikroprogramska riječ



Slika 2. Slijed događaja

Zadatke sastavio: prof. dr. sc. S. Ribarić

34

Samo je jedan Mali Ivica!

29.01.2000

1) TTBA ($A \leftarrow \bar{B} + 1$) \rightarrow OP.KOD 0110000
 Z, N ovise o rezultatu
 PRIBAVI ... 00001100

- a) dijagram boka
- b) bjet izvođenja po fazama
- c) sadržaj memorije

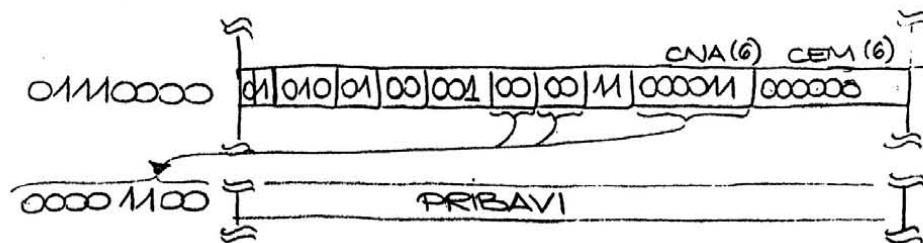
| | | | |
|----|--|---|---|
| a) | $L \leftarrow Q$ [$L(15-6,5-0)$] $\leftarrow 0, F(CEM)$ $R \leftarrow \bar{B}$ $Cin \leftarrow 1$ Rezultat bez pomaka Rez. u A ($A \leftarrow MB$) Utjecaj na stanica Syad. adresu fixna: (PRIBAVI) | $CA = 01$; $CEM = 0$?? $CB = 010$ $CP = 01$ $CSH = 00$ $CMB = 001$ $CST = 11$ $CAB = 00 *$ $CBB = 00 **$ $CNA = 000011$ | $MB \leftarrow Q + \bar{B} + 1$ $A \leftarrow MB$ $H = 0000110000000000$ ** |
| | | | |

b) /S * P(1)/ $H \leftarrow 01110000$
 /S * P(2)/ $F \leftarrow CM(H)$

$/CA(01)*P(0)/$ $L(15-6,5-0) \leftarrow 0, F(CEM)$
 $/CB(010)*P(0)/$ $R \leftarrow \bar{B}$
 $/CP(01)*P(0)/$ $C = 1$
 $/CSH(00)*P(0)/$ $MB \leftarrow Q; Q = S$
 $/CMB(001)*P(0)/$ $A \leftarrow MB$
 $/CST(11)*P(0)/$ $SR(15) \leftarrow XT$
 $SR(14) \leftarrow MB(15)$

 $/CAB(00)*P(1)/$ $H(1) \leftarrow 0$
 $/CBB(00)*P(1)/$ $H(0) \leftarrow 0$
 $/S * P(1)/$ $H(7-2) \leftarrow CNA(000011)$
 $/S * P(2)/$ $F \leftarrow CM(H)$

c) memorija:

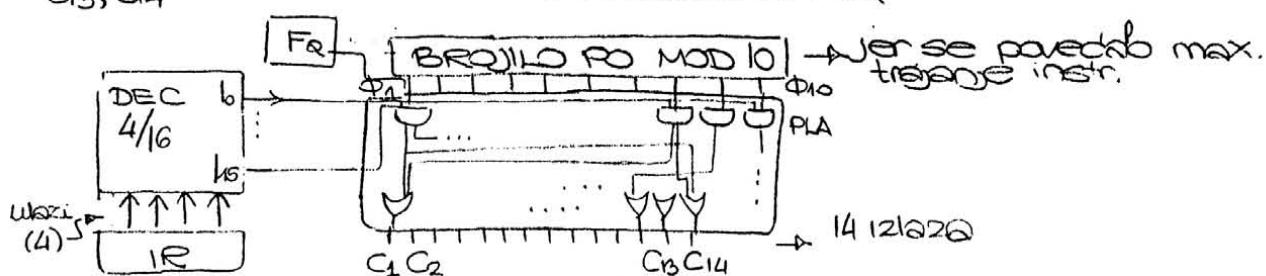


2) 8+2 instrukcije = 10 instr. \rightarrow DEKODER 4/16

10 DT

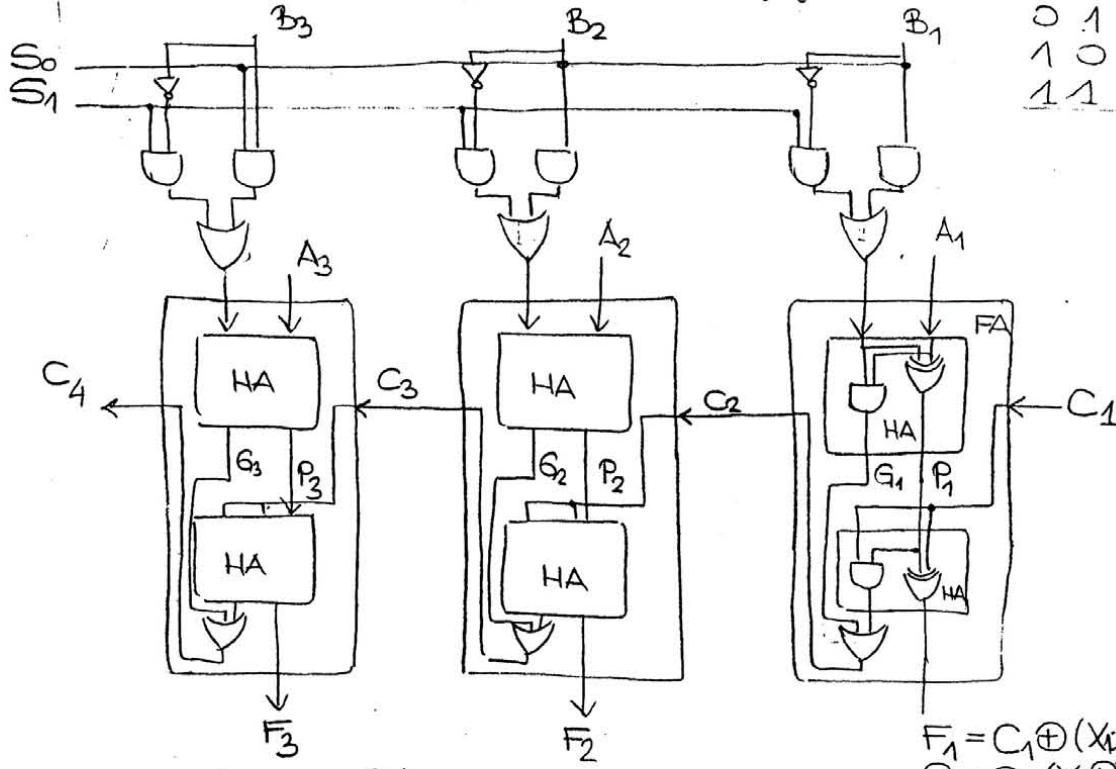
C₁₃, C₁₄

\Rightarrow Brojilo sekvenci po mod 10
 \Rightarrow 14 izlaza iz PLA



3) 3bitni FA na HA

Signalni su previdanje prenosova? (G_i, P_i)



| $S_1 S_0$ | Y |
|-----------|-----|
| 0 0 | 0 |
| 0 1 | 1 |
| 1 0 | 1 |
| 1 1 | 1 |

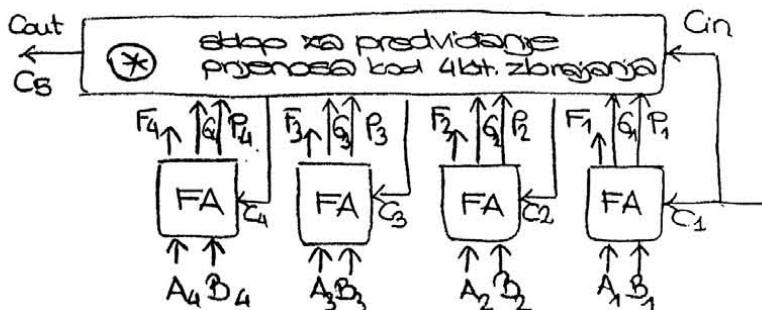
$$F_1 = C_1 \oplus (x_1 \oplus y_1)$$

$$C_2 = C_1 (x_1 \oplus y_1) + x_1 y_1$$

| $A+B$ | $S_1 S_0$ | C_1 |
|-------------|-----------|-------|
| $A+B$ | 0 1 | 0 |
| $A+B+1$ | 0 1 | 1 |
| $A+\bar{B}$ | 1 0 | 0 |
| $A-B$ | 1 0 | 1 |
| $A-1$ | 1 1 | 0 |
| A | 1 1 | 1 |
| $A+1$ | 0 0 | 1 |

ili 00 0

④ 4bitni:



log. formule ...

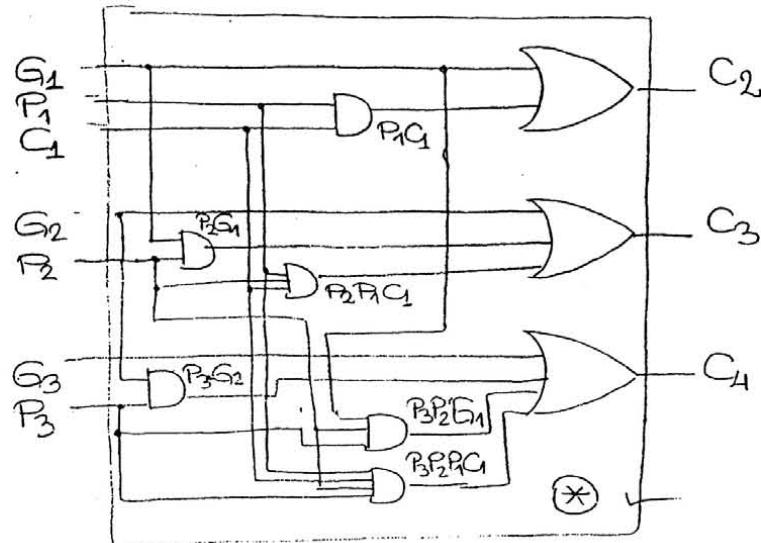
sklop \Rightarrow po log. formulema I i ILI vrata

$$C_2 = G_1 + P_1 C_1$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 C_1$$

$$C_4 = G_3 + P_3 C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$$

sklop... realizacija :



⑤ I: Log. oper:

Znamo: (iti biti) da vrednosti $F_i = X_i \oplus Y_i \oplus Z_i$

Z_i "ubijemo" sa signalom S_2 , tako da je $F_i = X_i \oplus Y_i$

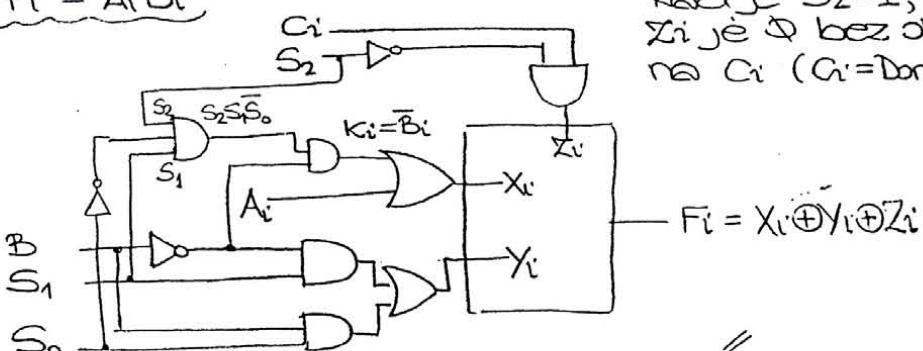
na Y_i stavimo \bar{B}_i
na X_i - " - $A + K_i$

$$F_i = (A_i + K_i) \oplus \bar{B}_i = (A_i + K_i) \bar{B}_i + (\bar{A}_i + K_i) \bar{B}_i = A_i \bar{B}_i + K_i \bar{B}_i + \bar{A}_i K_i \bar{B}_i$$

K_i je \bar{B}_i pa $K_i \bar{B}_i$ i $K_i B_i \Rightarrow 0$
i imamo $F_i = A_i B_i$

Realizacija:

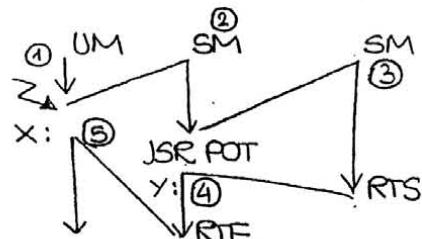
| $S_1 S_0$ | Y_i |
|-----------|-----------|
| 00 | 0 |
| 01 | B |
| 10 | \bar{B} |
| 11 | 1 |



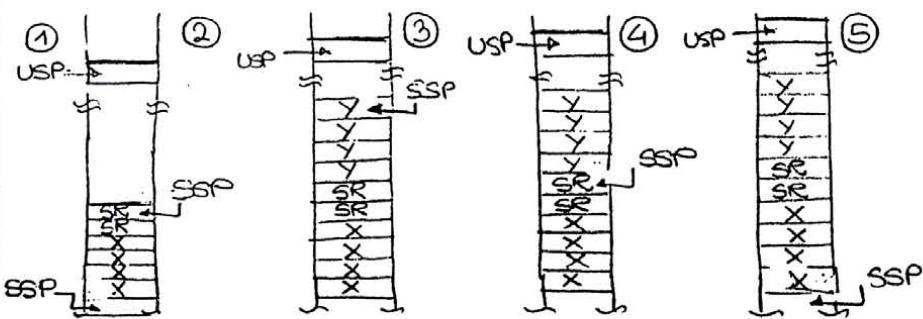
kad je $S_2 = 1$,
 Z_i je 0 bez obzira
na C_i (C_i = Don't Care)

5. $\Rightarrow S_2 S_1 S_0 = 110$ imamo Log. oper. I //

⑥ SR \rightarrow 2 byte
PC \rightarrow 4 byte
SSP = A01F
USP = 010A



update se ne konča!
User Stack Pointer
SSP \rightarrow Sistem Stack Pointer



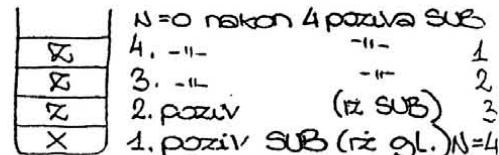
19. siječnja 1999.

Arhitektura i organizacija računala 2. kontrolna zadaća

- ① (5) Za model mikroprogramiranog procesora (slika 1) napisati mikroprogram za fazu IZVRŠI instrukcije BN (Branch if Negative), čije se izvođenje može opisati na sljedeći način: ako je sadržaj akumulatora A negativan tada $PC \leftarrow PC - 8$; inače $PC \leftarrow PC + 8$. Format mikroinstrukcije prikazan je na slici 2. Procesor je 16-bitovni, a najmanje značajan bit procesorske riječi je B0.
 - a) nacrtati dijagram toka mikroprograma,
 - b) prikazati izvođenje mikroprograma u obliku poput jezika CDL,
 - c) prikazati sadržaj mikroprogramske memorije.
- ② (4) Nacrtati jedan stupanj potpunog zbrajala i označiti signale koji se rabe u izvedbi sklopa za predviđanje bita prijenosa. Projektirati 5-bitovni sklop za predviđanje bita prijenosa. Procijeniti faktor ubrzanja rada zbrajala ako je kašnjenje na logičkim vratima $t_g = 10$ ns. Faktor ubrzanja neka je omjer između vremena dobivanja rezultata za "klasičnu" izvedbu (za najnepovoljniji slučaj) te vremena potrebnog za dobivanje rezultata uz uporabu sklopa za predviđanje bita prijenosa.

3. (4) Zadan je programski odsječak s rekurzivnom procedurom:

```
    . . . ; glavni program
    . . .
N:=4
CALL SUB
X:   . . .
    . . .
SUB  . . .
    . . .
    ↳ N:=N-1
    IF (N>0) THEN
        CALL SUB
    Z:   . . .
        . . .
        RET
```



Treba nacrtati i opisati stanja stogova tijekom izvođenja programskog odsječka..

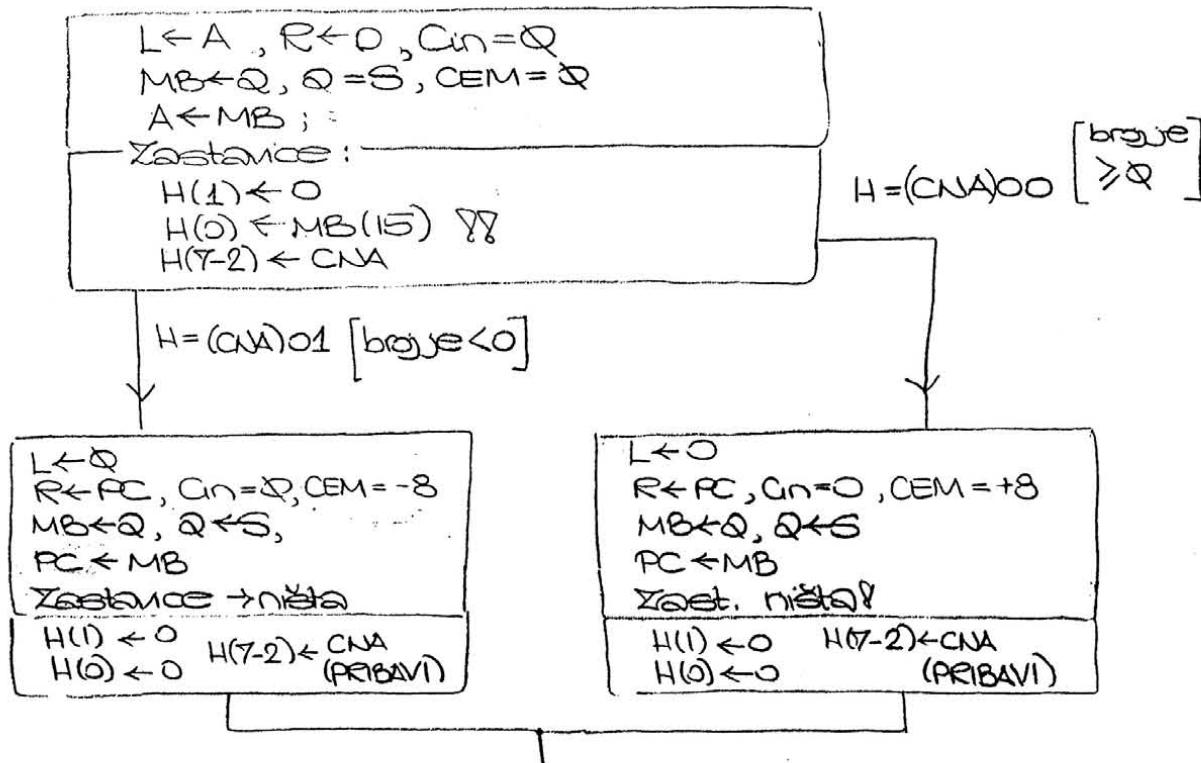
4. (5) Na temelju modifikacije osnovnog sklopa 2-bitovne aritmetičko-logičke jedinice u skladu s uobičajenim pristupom realizirati logičke operacije I, ILI, NE i ISKLJUČIVO ILI.
5. (2) Navesti i objasniti vrste formata upravljačke riječi (mikroriječi).

19.01.1999

1) BN (Branch if Negative)

ako $A < 0$ tada $PC \leftarrow PC - 8$
inče $PC \leftarrow PC + 8$

a) Dijagram toka



b) Izvođenje: [Ovdje sam pretp. da u $B \neq Q$, inče je potrebno]
[Izvrsi 1 instrukciju koja to i izvršava $B \leftarrow Q$]

\S * P(1) /
\S * P(2) /

H ← XXXXXXXX
F ← CM(H)

izvrši ?

\CA(M) * P(0) /
\CB(00) * P(0) /
\COP(00) * P(0) /
\CSH(00) * P(0) /
\CMB(00) * P(0) /
\CST(00) * P(0) /

L ← A
R ← B (=Q)
C = 0
MB ← Q, Q = S
nema prenosa
nema učit. na SE

\CAB(00) * P(1) /

H(1) ← 0

\CBB(11) * P(1) /

H(0) ← MB(15)

\S * P(1) /

H(7-2) ← CNA (yyyyyy)

\S * P(2) /

F ← CM(H)

⇒ Ako je $A < 0$, u F je 1. m.s adresu 00000001
 $A \geq 0$, u F - 11 - -11 - 00000000

Slijede 2 m.s, 1. m.s adresi 00000010 (A<0)
2 - " - 00 (A>0)



1

$A < 0$

$\backslash CA(10) * P(0)$
 $\backslash CB(000) * P(0)$
 $\backslash COP(0) * P(0)$
 $\backslash CSH(00) * P(0)$
 $\backslash CMB(010) * P(0)$
 $\backslash CST(00) * P(0)$
 $\backslash CAB(01) * P(1)$
 $\backslash CBB(00) * P(1)$
 $\backslash S * P(1)$
 $\backslash S * P(2)$

$L \leftarrow F(CEM), Q$
 nema pojava Q (R=Q)
 $C = 0$
 $MB \leftarrow Q, Q = S$
 $B \leftarrow MB$
 nista
 $H(1) \leftarrow 1$
 $H(0) \leftarrow 0$
 $H(7-2) \leftarrow \text{yyyyyy}$
 $F \leftarrow CM(H)$

$L \leftarrow O, FF(CEM)$
 $R \leftarrow B$
 $C = 0$
 $MB \leftarrow Q, Q = S$
 $A \leftarrow MB$
 nista
 $H(1) \leftarrow 1$
 $H(0) \leftarrow 1$
 $H(7-2) \leftarrow \text{yyyyyy}$
 $F \leftarrow CM(H)$

2

$\backslash CA(01) * P(0)$
 $\backslash CB(001) * P(0)$
 $\backslash COP(00) * P(0)$
 $\backslash CSH(00) * P(0)$
 $\backslash CMB(001) * P(0)$
 $\backslash CST(00) * P(0)$
 $\backslash CAB(01) * P(1)$
 $\backslash CBB(01) * P(1)$
 $\backslash S * P(1)$
 $\backslash S * P(2)$

$L \leftarrow A$
 $R \leftarrow PC$
 $C = 0$
 $MB \leftarrow Q, Q = S$
 $PC \leftarrow MB$
 nista
 $H(0) \leftarrow 0$
 $H(1) \leftarrow 0$
 $H(7-2) \leftarrow \text{zzzzzz}$
 $F \leftarrow CM(H)$

3

$\backslash CA(11) * P(0)$
 $\backslash CB(01) * P(0)$
 $\backslash COP(00) * P(0)$
 $\backslash CSH(00) * P(0)$
 $\backslash CMB(011) * P(0)$
 $\backslash CST(00) * P(0)$
 $\backslash CAB(00) * P(1)$
 $\backslash CBB(00) * P(1)$
 $\backslash S * P(1)$
 $\backslash S * P(2)$

© xxxxxxxxxx

$A > 0$
 $\text{yyyyyyy} 00$
 $\text{yyyyyyy} 01$
 $A < 0$
 $\text{yyyyyyy} 10$
 $\text{yyyyyyy} 11$

| CA | CB | COP | CSH | CMB | CAB | CBB | CST | CNA(6) | CEM(5) |
|----|-----|-----|-----|-----|-----|-----|-----|--------|----------|
| 1 | 001 | 00 | 00 | 001 | 00 | 00 | 00 | yyyyyy | 00000000 |
| 10 | 000 | 00 | 00 | 010 | 01 | 00 | 00 | yyyyyy | 11111111 |
| 01 | 001 | 00 | 00 | 001 | 01 | 01 | 00 | yyyyyy | 11111000 |
| 11 | 001 | 00 | 00 | 001 | 00 | 00 | 00 | zzzzzz | 00000000 |

PRIBAVI SLJEDEĆU INSTRUKCIJU

1. U $L \leftarrow 11111111 00000000 + Q + Q \rightarrow u B$
2. U $L \leftarrow 00000000 111111000 + B + Q \rightarrow u A$
3. U $L \leftarrow A + PC + Q \rightarrow PC$

$$-B \Rightarrow 11110111$$

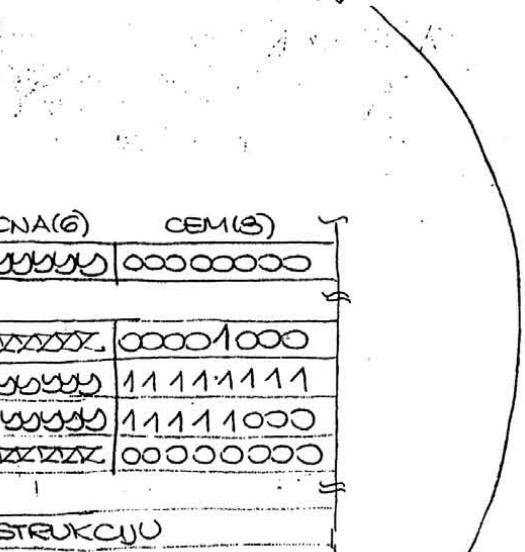
$$\frac{1}{11111000}$$

$A \geq 0$

$L \leftarrow O, F(CEM)$
 $R \leftarrow PC$
 $C = 0$
 $MB \leftarrow Q, Q = S$
 $PC \leftarrow MB$
 nista
 $H(1) \leftarrow 0$
 $H(0) \leftarrow 0$
 $H(7-2) \leftarrow \text{zzzzzz}$
 $F \leftarrow CM(H)$

$\exists A \geq 0$ je lako, CEM = 8
 on se smješta u dajući byte L registru (1 μI)
 No za $A < 0$, moramo imati
 ODUZIMANJE:
 $A - B \equiv A + B + 1$ (16 bitno)
 tj. 1 spremi 8 bitova L
 moraju biti sve 111...11
 TREBAJU NAM 3 μI

1. U gornji byte $L \leftarrow 111111 00000000$
2. U donji byte $L \leftarrow 00000000 111111000$
3. Izbrisi S PC



$$R \quad C$$

$$\uparrow \quad \uparrow$$

$$11111111$$

$$00000000 111111000$$

1. U $L \leftarrow 00000000 00001000 + 0 + 0 \Rightarrow u B$
2. U $L \leftarrow 00000000 00000000 + E + 1 \Rightarrow u A$
3. U $L \leftarrow A + PC + O \Rightarrow PC$

2) FA:

| A | B | Cin | Cout | F |
|---|---|-----|------|---|
| X | Y | Z | | |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

$$F = (X \oplus Y) \oplus C_{in}$$

$$C_{out} = (X \oplus Y) \oplus C_{in} + XY$$

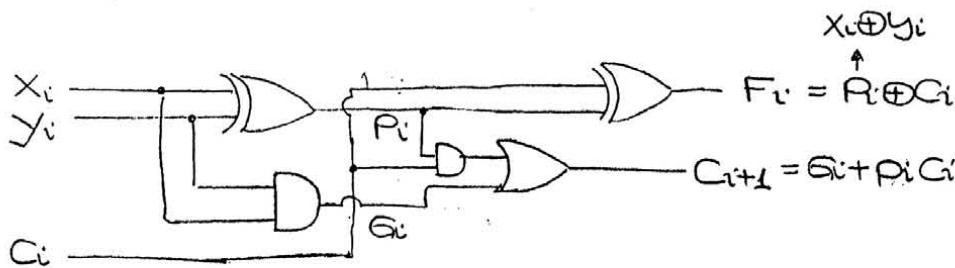
HA:

$$G_i = A_i B_i$$

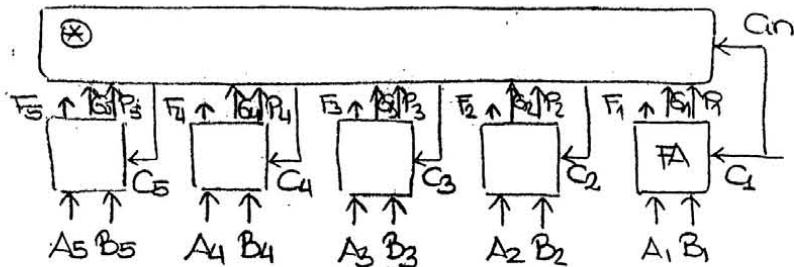
$$P_i = A_i \oplus B_i$$

$$C_{i+1} = G_i + P_i C_i$$

$$F_i = P_i \oplus C_i$$



5-bitani:



$$C_1 = G_1 + P_1 C_1$$

$$C_2 = G_2 + P_2 G_1 + P_2 P_1 C_1$$

$$C_3 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_1$$

$$C_4 = G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 + P_4 P_3 P_2 P_1 C_1$$

$$C_5 = G_5 + P_5 G_4 + P_5 P_4 G_3 + P_5 P_4 P_3 G_2 + P_5 P_4 P_3 P_2 G_1 + P_5 P_4 P_3 P_2 P_1 C_1$$

REALIZACIJA

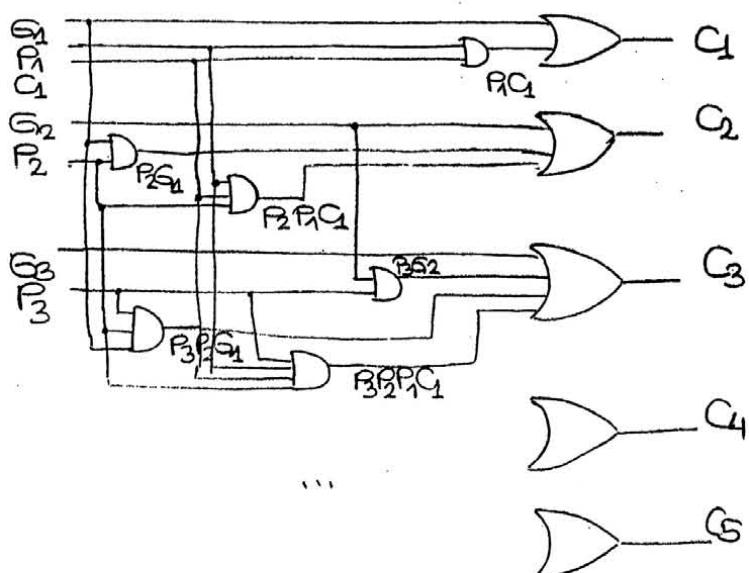
$$t_{gj} = 10 \text{ ns}$$

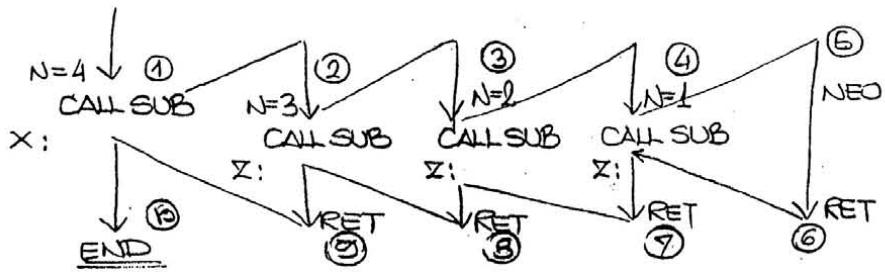
Brzine:

BEZ CARRY LOOK AHEAD

$$t_{uk} = 2t_{gj} \cdot 5$$

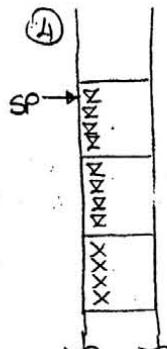
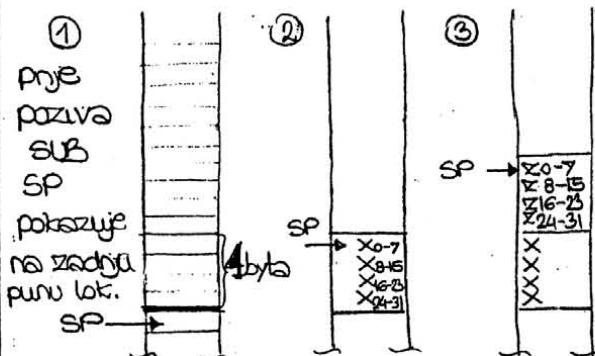
SA
t_{uk} = 2t_{gj}



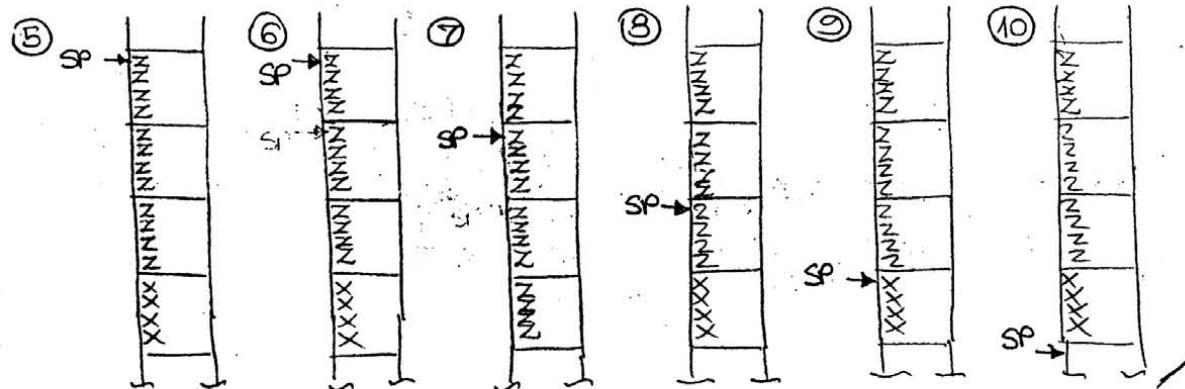


Pripremke:

- memorija
- bitno organizirana
- PC 4 bytean



- ① poje poziva SUB
② 1. poziv SUB, ne doguje je pov.adr. g. pog
③ 1.REK. poziv, ne doguje pov. adr. od SUB
- ! ! !
itd...



A) 2bitov. ALU : I, ILI, NE , EX-ILI $F_i = X_i \oplus Y_i \oplus Z_i$

EX-1H : Ako $Z_i = Q$ $\Rightarrow F_i = X_i \oplus Y_i$ $\Leftrightarrow A = X_i$
 $B = Y_i$

| S ₁ S ₀ | Y ₁ |
|-------------------------------|----------------|
| 00 | 0 |
| 01 | B |
| 10 | B |
| 11 | 1 |

Real. smo
EX-ILI

NE : - " - $F_i = X_i \oplus Y_i = \overline{XY} + \overline{XY}$ $\Leftrightarrow A = X_i$
 $M \dots 1 = Y_i$
 $\bar{Y} = 0$

| S ₁ S ₀ | Y ₁ |
|-------------------------------|----------------|
| 00 | 0 |
| 01 | B |
| 10 | B |
| 11 | 1 |

Real. smo
NE

ILI : - " - $F_i = X_i \oplus Y_i = \overline{XY} + \overline{XY}$ $\Leftrightarrow A+B = X_i$
 $Q = Y_i$

| I |
|---|
|---|

ILI

I : - " - $F = AB + \overline{B}B + \overline{A}B$ $\Leftrightarrow A \oplus B = X_i$
 $B = Y_i$

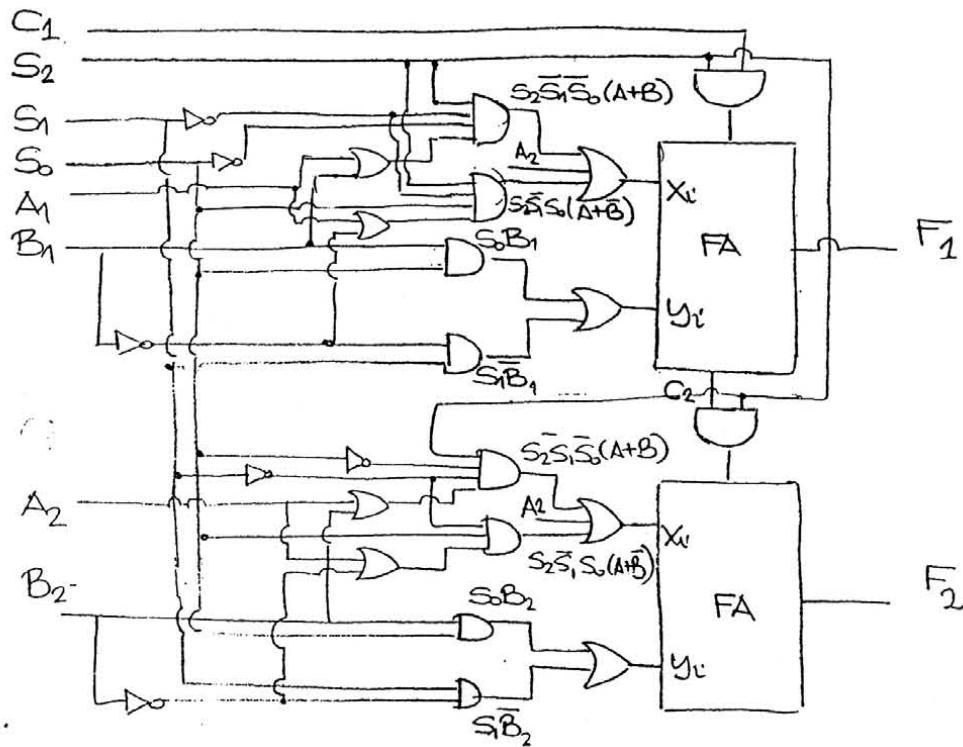
| I |
|---|
|---|

$S_2 = 1 \Rightarrow Z = Q$

| S ₂ S ₁ S ₀ | F |
|--|----------|
| 1 0 1 | \oplus |
| 1 1 1 | NE |
| 1 0 0 | ILI |
| 1 0 1 | I |

→ s tim da na X_i učinimo $A+B$
 $A+\overline{B}$





5) Vrste formata / uRječi:

a) uRječi s izravnim upravljanjem: VLIW
A uoperaciji odgovara 1 bit u uRječi

b) uRječi s grupiranjem bitova

→ uoper. (istodobne) se grupiraju i povežu (temelju se na istm bitom → vlastita na prostoru)

→ samostalne uoperac. se kodiraju "svim" poljem bitova

c) uRječi s višestrukim formatom

A uIndr. u uprogri specifidira max 4 uoperacije

Npr.

FORMAT 0

IF F = 0 then mop ∈ {TRANSFER}
and interpret (opd1 source; opd2 result)

| | | | |
|---|-----|------|------|
| 2 | 6 | 5 | 5 |
| F | mop | opd1 | opd2 |

FORMAT 1

IF F = 1 then mop1 ∈ {ADD, SUB, OR, AND...}
mop2 ∈ {R/W}
mop3 ∈ {INC-PC}
& if. if mop1 ∈ {SHL, SAR} then interpret (modifier = amount)

| | | | | | |
|---|------|------|------|----------|---|
| 2 | 4 | 1 | 1 | 6 | 4 |
| F | mop1 | mop2 | mop3 | modifier | |

FORMAT 2

IF F = 2 then mop ∈ {MASK}
& interpret (bd1 as bound¹)
(bd2 as bound²)

| | | | |
|---|-----|-----|-----|
| 2 | 4 | 6 | 6 |
| F | mop | bd1 | bd2 |

FORMAT 3

IF F = 3 then brehop ∈ {BN, BPZ, BU...}
& interpret (addr as branch adres)

| | | | |
|---|--------|------|-------|
| 2 | 3 | 9 | 4 |
| F | brehop | addr | |

d) Vertikalno/horizontalsko uprogramiranje

↳ nekoliko paralelnih razodjivih uoperacija

isključivo su jedna uoperacija

PISMENI ISPIT IZ ARHITEKTURE I ORGANIZACIJE RAČUNALA I
09.04.1997.

1.(4) Opisati formate mikroinstrukcija.

2.(5) Napisati mikroprogram za instrukciju:

TCBA - prijenos jediničnog komplementa akumulatora B u akumulator A.

Zastavice neka se postavljaju u zavisnosti od sadržaja koji se prenosi.

| CA | CB | COP | CSH | CMB | CAB | CBB | CST | CNA | CEM |
|--------------------------|----|-----|--------------------|-----|-----|------------------|-----|---------------------------------|-----|
| 31 | 30 | 27 | 25 | 23 | 20 | 18 | 16 | 14 | 8 |
| | | | | | | | | | |
| CA: | | | CB: | | | COP: | | CMB: | |
| 00 nema prijenosa | | | 000 nema prijenosa | | | 00 suma uz C=0 | | 000 nema prijenosa | |
| 01 L(15-8,7-0)←0, F(CEM) | | | 001 R←B | | | 01 suma uz C=1 | | 001 A←MB | |
| 10 L(15-8,7-0)←F(CEM),0 | | | 010 R←B* | | | 10 ne koristi se | | 010 B←MB | |
| 11 L←A | | | 011 R←PC | | | 11 ne koristi se | | 011 PC←MB | |
| | | | 100 R←SR | | | | | 100 SR←MB | |
| | | | | | | | | 101 DOUT←MB | |
| CSH: | | | CAB: | | | CBB: | | CST: | |
| 00 MB←Q, Q=S | | | 00 H(1)←0 | | | 00 H(0)←0 | | 00 nema utjecaja na SR | |
| 01 MB←Q, Q=shr S | | | 01 H(1)←1 | | | 01 H(0)←1 | | 01 SR(15)←ZT | |
| 10 MB←Q, Q=shl S | | | 10 H(1)←SR(15) | | | 10 H(0)←SR(14) | | 10 SR(14)←MB(15) | |
| 11 MB←IN | | | 11 H(1)←SR(14) | | | 11 H(0)←MB(15) | | 11 SR(15)←ZT ; SR(14)←MB(15) | |

Slika uz zadatku 2: mikroprocesorska rječ modela CPU

3.(3) Nacrtati organizaciju upravljačke jedinice 8-instrukcijskog procesora. Pokazati izvedbu brojila sekvenci po modulu 8.

4.(4) Za pojednostavljeni model mikroprocesora sa dodanim akumulatorom B nacrtajte stanje na sabircicama tijekom izvođenja instrukcije:

TAB - prijenos sadržaja akumulatora A u akumulator B.

Instrukcija se nalazi na adresi 1000_H a njen operacijski kod je 29_H.

5.(4) Opisati Denningov model rješenja preslikavanja logičkog u fizički adresni prostor. Ukazati na neologičnost u modelu i ponuditi rješenje.

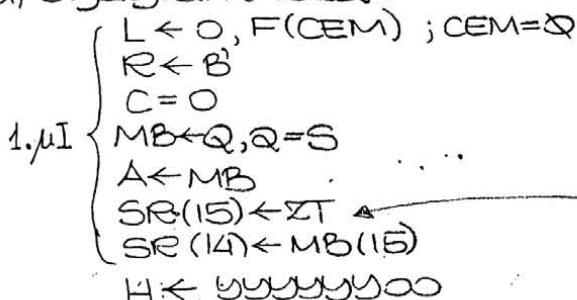
09.04.1987

1) TEORIJA (Vidi 5. zdrž. ROKA 19.01.1980)

2) TCBA \rightarrow $A \leftarrow \bar{B}$

zastavice se postavljaju.

a) dijagram tokoa:



$ZT(\text{FLAGS}) = 1$ kad $A = Q$
 $ZT = 0$ kad je $A \neq Q$
 a skripti je
 kompl $\leftarrow ZT' = 0$ kad $A = 0$ (što je
 $ZT' = 1$ - - - $A \neq 0$ istočvar)

Problem je u ZT ? U skripti na predavanju piše:

| ZT' | MB | pa ljudi pogrešno misle da ZT i Z (zastavica ZB) |
|-------|------|---|
| 0 | 0 | su međusobno komplementarni, a nisu jer |
| 1 | 1 | tabela je za ZT' (ZT komplement ZT), pa je $ZT = Z(\text{FLAGS})$ |

b) program:

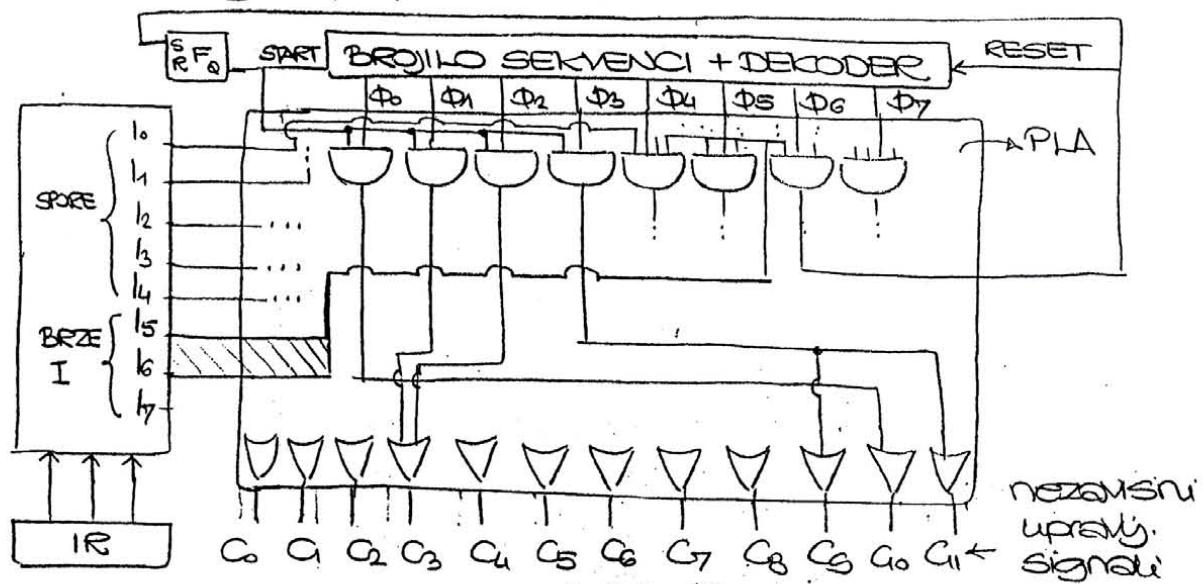
| | | |
|-----------------|--|------------|
| /S * P(1)/ | H $\leftarrow \text{xxxxxxxxxx}$ | |
| /S * P(2)/ | F $\leftarrow CM(H)$ | 1. μizvrsi |
| <hr/> | | |
| /CA(01)*P(0)/ | L (15-8,7-0) $\leftarrow 0, F(CEM) ; CEM = Q$ | |
| /CB(010)*P(0)/ | R $\leftarrow \bar{B}$ | |
| /COP(00)*P(0)/ | C = 0 | |
| /CSH(00)*P(0)/ | MB $\leftarrow Q, Q = S$ | |
| /CMB(100)*P(0)/ | SR $\leftarrow MB$ | |
| /CST(11)*P(0)/ | SR(15) $\leftarrow ZT ; SR(14) \leftarrow MB(15)$ | 1. upnjava |
| <hr/> | | |
| /CAB(00)*P(1)/ | H(1) $\leftarrow 0$ | |
| /CBB(00)*P(1)/ | H(0) $\leftarrow 0$ | |
| /S * P(1)/ | H(7-2) $\leftarrow F(CNA) ; CNA = \text{yyyyyyyy}$ | |
| /S * P(2)/ | F $\leftarrow CM(H)$ | |

a) sadržaj memorije

| CA | CB | COP | CSH | CMB | CAB | CBB | CST | CNA(6) | CEM(8) |
|-----------------------------------|-----|-----|-----|-----|-----|-----|-----|----------|----------|
| 01 | 010 | 00 | 00 | 100 | 00 | 00 | 11 | yyyyyyyy | oooooooo |
| PRIBAVI KOD SLJEDEĆE INSTRUKCIJE. | | | | | | | | | |
| : | | | | | | | | | |

3) Organizacija CU:

8-instrukcijski µP \Rightarrow DEKODER 3/8
Izvedba brojila po modulu 8



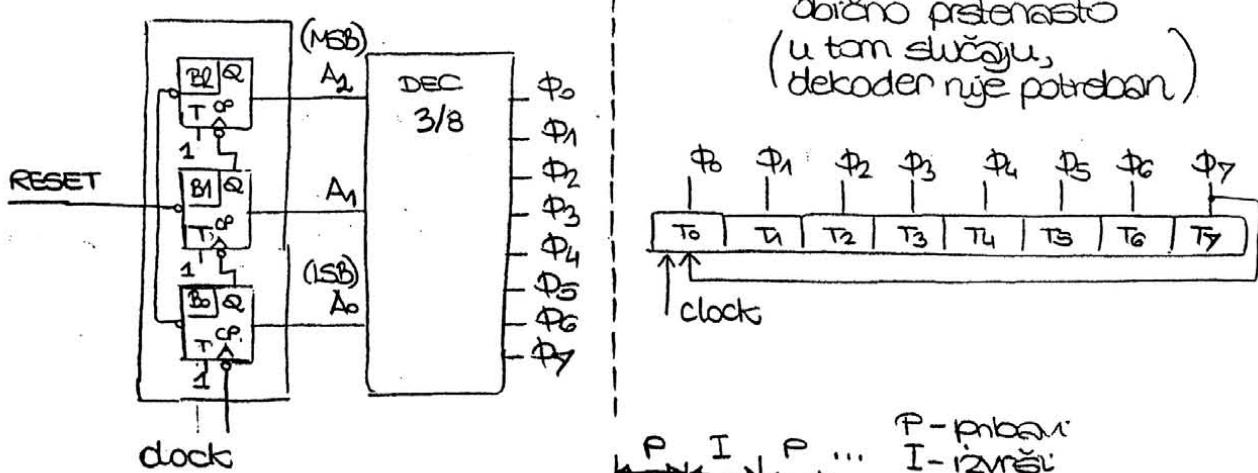
Prva 4 ϕ su za PRIBAVI (Csignali ⑩, ③, ⑨, ⑪)

Druge 4 ϕ su za izvrši (kod sporih instr.) \rightarrow TRAJU SIT

- - 2 ϕ za izvrši (kod brzih - -) \rightarrow TRAJU GDT

Kod BRZIH se u ϕ_8 RESETIRA BROJILO \Rightarrow POSTAVI FETCH BISTABIL opet u 1 (jer u fazi izvrši on postaje 0)

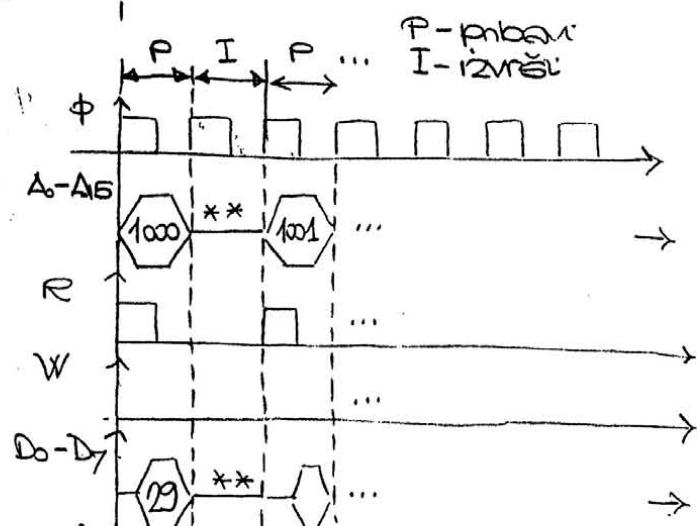
Izvedba BROJILA:



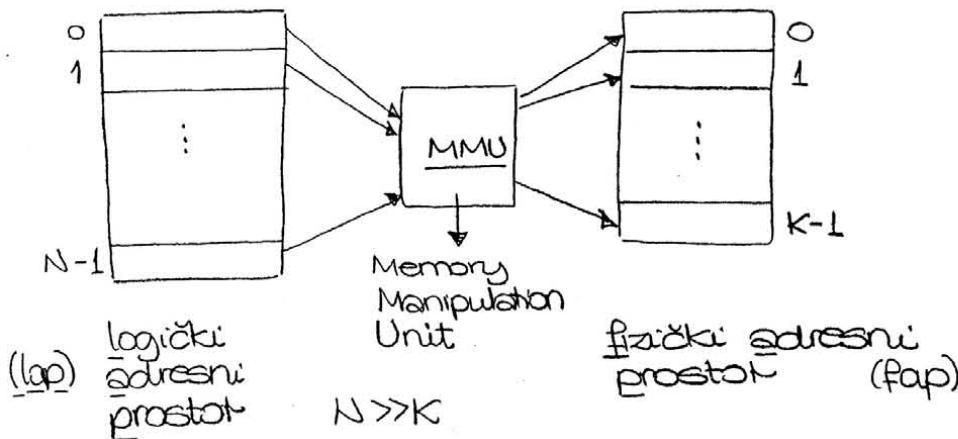
4) Stanje na sabirnicama:

ADRESA OP KOD
 1000_H $29_H \rightarrow TAB$

Čim se dekodira (1bitna) instrukcija, faza PRIBAVI završava i ide se na izvrši, a budući da se interno prebacuje sadržaj reg. iz A u B, µP se odspaja od adresab (HIGH X **)



5.) Denningov model adresnog preslikavanja:



Neka je u l^ap logička adresa "a".

Denningov model kaže da za l^aog.adr postoji u MMU element if(a) koji sadrži:

"a", fizičku adresu podatka (u gl.memoriji) ako se podatak adresiran log.adresom tamo nalazi

ili "a", fizičku adresu podatka (u sekundarnoj memoriji), ako se podatak NE NALAZI u gl.memoriji

(može, u jednost. izvedbi, umjesto "a" biti "Q", što označava) (da se podatak ne nalazi u gl.memo.)

Nalogičnost:

Ako za l^aog.adr. \exists elem. u MMU, to znači da ima točno N elemenata MMU tj. veličina MMU (koja se nalazi u gl.mem., ili broj prirođenoj, avno o izvedbi) je = ogromnom l^ap-u?

Rješenje nalogičnosti:

l^ap i fas se podijele u blokove, i onda MMU sadrži samo toliko elemenata koliko ima:

ili: blokova u l^ap-u (1 izvedba: izravno preslikavanje)

ili - " - - " - fas-u (2 izvedba: asocijativno - " -)

Samo je jedan Mali Ivica!

(1) Za model mikroprogramiranog procesora (slika 1) napisati program za fazu izvrsi strojne instrukcije NOP (No Operation). Ta instrukcija ne radi nista, vec samo troši vrijeme - vjerujec da je izvrsi traje dva perioda signala vremenskog vodjenja. Format mikroinstrukcije prikazan je na slici 2.

- a) nacrtati dijagram toka mikroprograma
- b) prikazati izvođenje mikroprograma u obliku poput jezika CBL ?
- c) prikazati sadržaj mikroprogramskog memorije

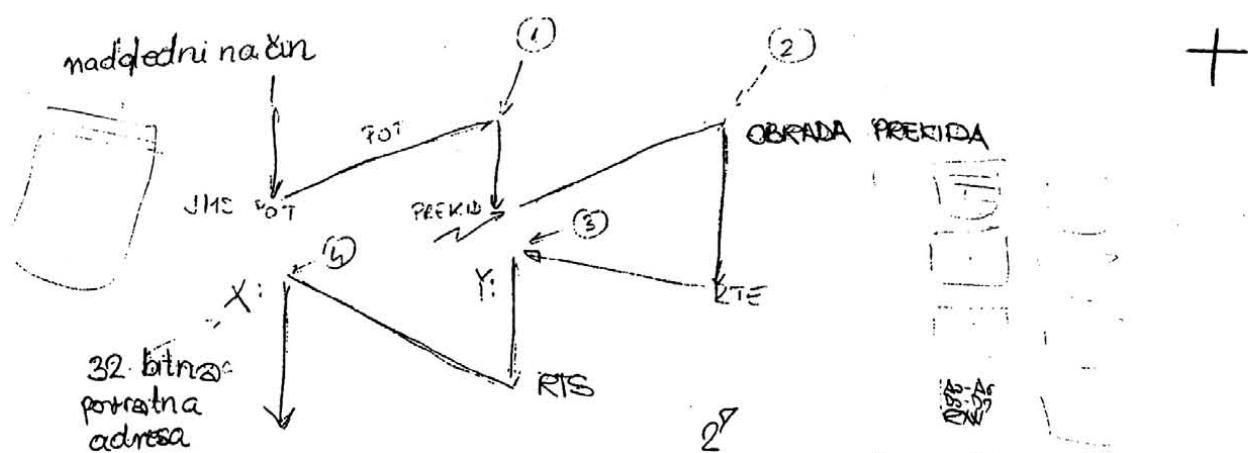
(2) Nacrtati stanje na sabirnici računala izgrađenog na temelju pojednostavljenog modela mikroprocesora i to za sljedeći programski odsjek (dok se on ne okonča).

Adresa

| | |
|-------|---|
| 0100H | DEC \$0101 ; dekrementiraj (Op. kod B9) |
| 0103H | BNE \$FC ; granaj ako nije jednak null (Op. kod 4E) |

Sadržaj memorije lokacije 0105H je 02H

Za sljed događaja sa slike 3 prikazati stanje stogova i karakta stoga sa računalo na bazi mikroprocesora HCG8000 (i to za točke označene na slici) stog ima bajtnu adresiranost. Komentirati što bi se dogodilo ako se umjesto instrukcije RTE zabilježi upotrebљiva instrukcija RTS :



(3) Upotrebom građevine komponente RAM 1Kx8 bayova odlikovat memoriju modela kapaciteta $\frac{1}{2}K \times 8$ bayova. Model računa se postavi na početne adrese **1000H**. Koristiti spispuno adresno delodjeljivanje. Građevina L411 komponenti linearno adresira je, upotrebjavši učit R/W, da učita iz sklopnika (CS2, CS1, CS0, OE, CS1, CS0, OE) ulaze AD-16 i D-12.?

Adresni sklopnik:

| | | | |
|-----|------|------|------|
| 000 | 0000 | 0000 | 0000 |
| A15 | A12 | A11 | A8 |

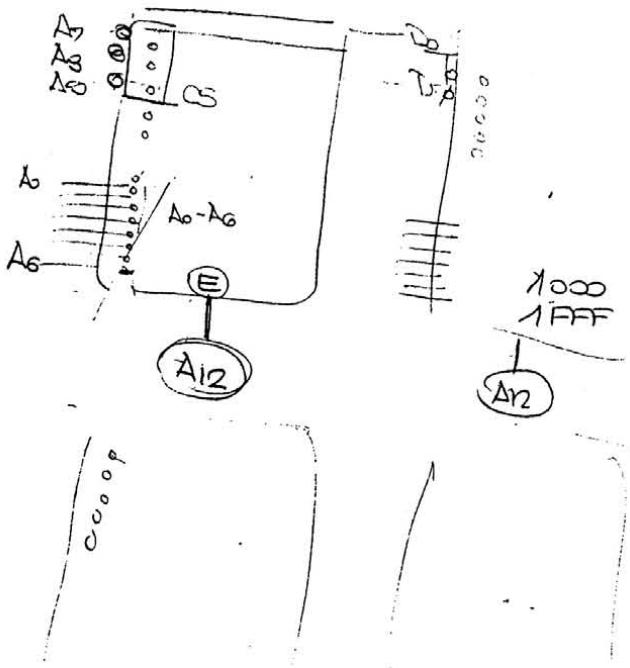
$$\begin{array}{r} 1000 \\ \times 8 \\ \hline 1000 \end{array}$$

101
000

XX
X =

1000
FFFF

A7
A8
A9



12

12

12

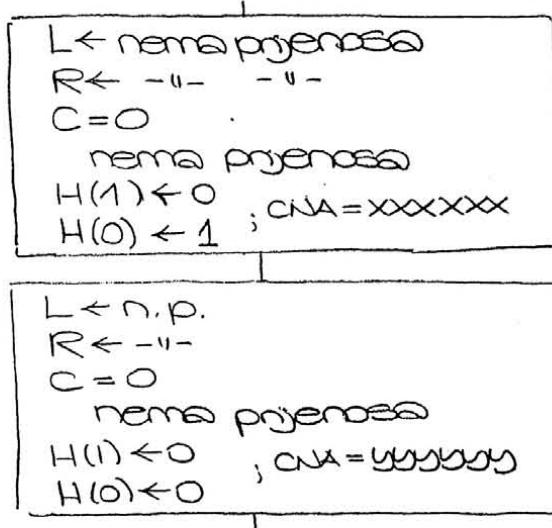
52

12

28.01.1999

1.) NOP → Faza izvrsi traje 2 PERIODE SIGNALA!
(tj. INSTR. NOP ima 2 instrukcije ??)

a) dijagram:



b) program:

```

/CA(∞)*P(0) / nema projenosa
/CB(∞)*P(0) / -||-
/COP(∞)*P(0) / C=0
/CSH(∞)*P(0) / MB←Q, Q=S
/CMB(∞)*P(0) / nema projenosa
/CST(∞)*P(0) / nema utjecaj
/CAB(∞)*P(1) / H(1)←0
/CBB(∞)*P(1) / H(0)←1
/S*P(1) / H(7-2)←xxxxxx
/S*P(2) / F←CM(H)
/CA(∞)*P(0) / nema projen.
/CB(∞)*P(0) / -||-
/COP(∞)*P(0) / C=0
/CSH(∞)*P(0) / MB←Q, Q=S
/CMB(∞)*P(0) / nema Pn
/CST(∞)*P(0) / -||- utjec.
/CAB(∞)*P(1) / H(1)←0
/CBB(∞)*P(1) / H(0)←0
/S*P(1) / H(7-2)←yyyyyy
/S*P(2) / F←CM(H)

```

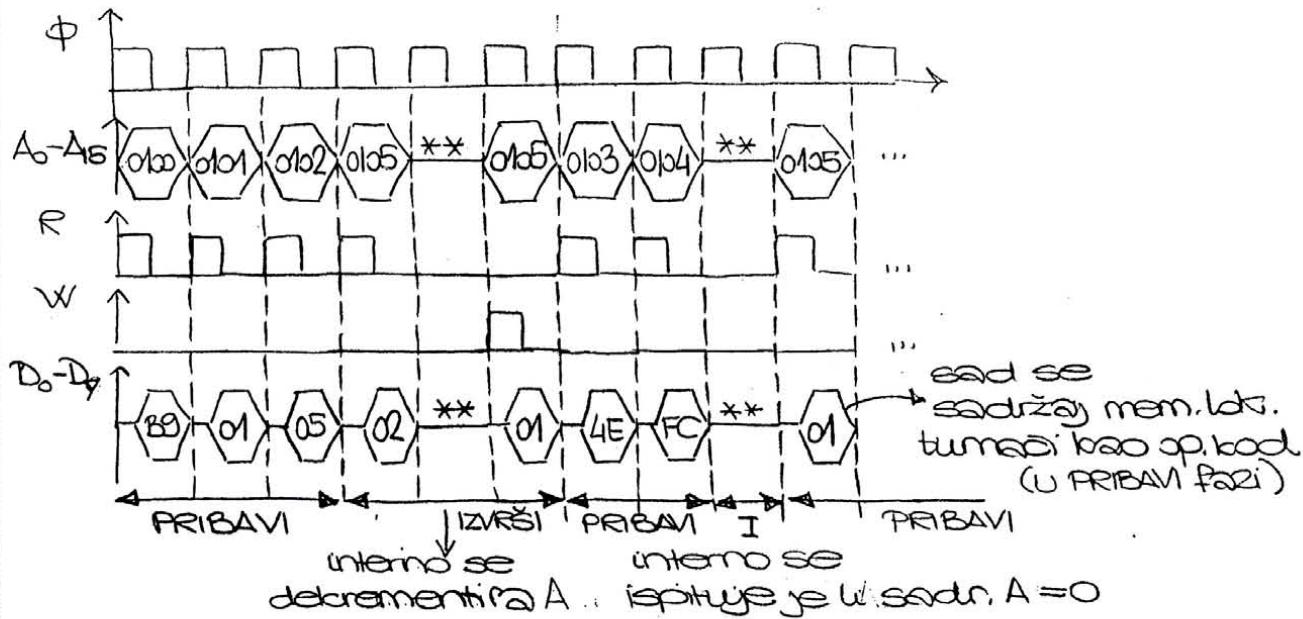
c) memorija:

| | CA | CB | OP | CSH | CMB | C&CBB | CST | CNA(6) | CEM(8) |
|-------------------------------|----|----|----|-----|-----|-------|-----|--------|----------|
| xxxxxx00 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | xxxxxx | 00000000 |
| xxxxxx01 | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ | yyyyyy | 00000000 |
| PRIBAVI KOD SJED. INSTRUKCIJE | | | | | | | | | |

2.) Adresa

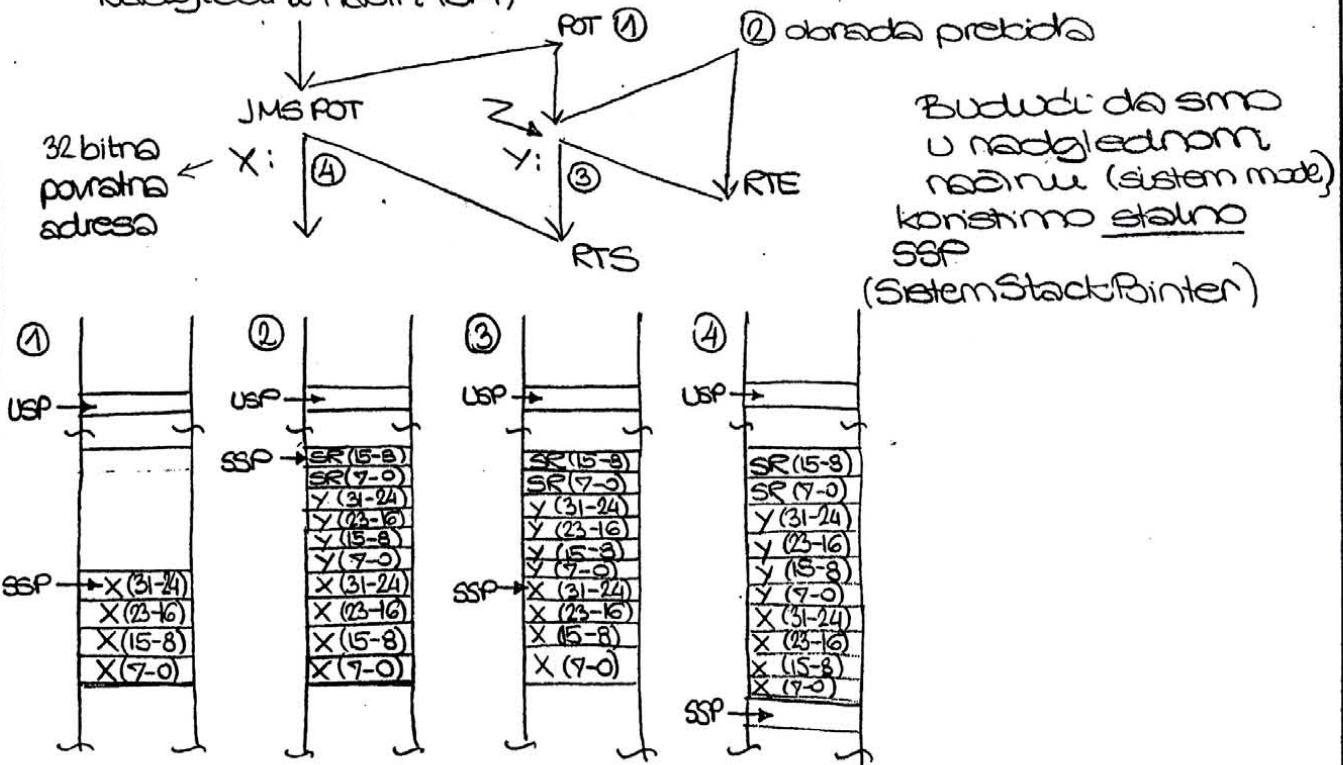
Kod

| | | |
|------|------------|-----------------------------|
| 0100 | DEC \$0105 | (op.kod B9) |
| 0103 | BNE \$FC | (Branch if ≠ 0) (op.kod 4E) |
| 0105 | 02 | |



3.) Stanje stogova? MC68000 \Rightarrow SIPS PONTER POKAZUJE NA zadnju punu lokaciju
Bytno adresiran stog
Ako umjesto RTE upotrijebimo RTS?

nadgledni način (SM)

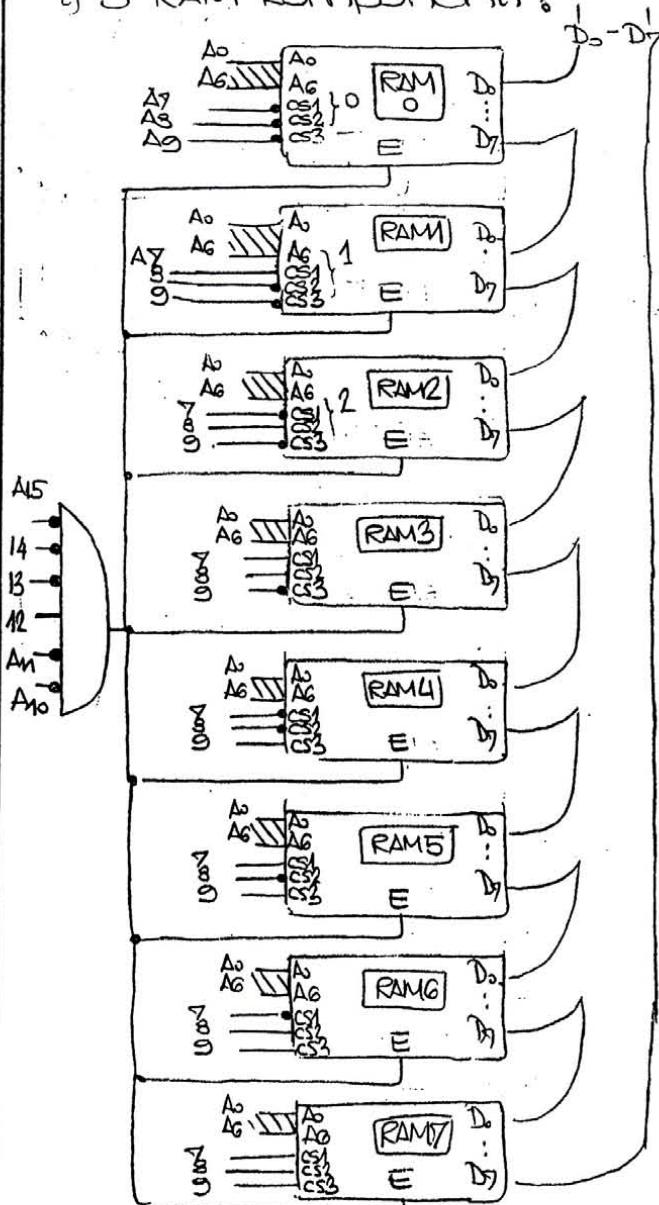


Da smo u mjeri RTE upotrijebili RTS, ne bi se izazvala iznimka, jer smo u smode-u, ved bi se instrukcija izvela: sa stoga (ssp) bi se 'skinulo' 4 byte koji bi se tumačili kao povratna adresa potprograma.

Dakle u PC bi stamku: $(SR(15-8)|(SR(7-0)|PC(31-24)|PC(23-16))$ i tražili instr. s te adresi u memoriji.

- ④ Upotreboom građevne komponente RAM 128×8 byte oblikovati mem. modul kapaciteta $1K \times 8$ byteva. Modul neka se javlja na poč. adresi 1000_H . Konstanti potpuno adresano dekodiranje. Građevna RAM komponenta ima linearno adresiranje, upravljački ulaz R/W, 5 waze za izbor čipa (CHIP SELECT: $\overline{CS}_0, CS_1, CS_2, CS_3, \overline{CS}_4$) i waze A₀-A₆ i D₀-D₇.

1 komponenta: 128×8 byte = $2^7 \times 8$ byte
Mi trebamo $1K \times 8$ byte = $2^{10} \times 8$ byte = $2^3 \cdot 2^7 \times 8$ byte = $8 \cdot (128 \times 8$ byte)
tj 8 RAM komponenti?



Poč. adresa : 1000_H
Kam. adresa : $1000_H + 2^{10}$

$$2^{10} = \frac{0000010000000000}{0 \quad 4 \quad 0 \quad 0} \quad 1400_H$$

$$+ 1 \quad 0 \quad 0 \quad 0$$

A₀-A₆ su XXX...X (Dni adresiraju hijeci)
(u pojedinačnom RAM-u)

| A ₇ | A ₈ | A ₉ | A ₁₀ | A ₁₁ | A ₁₂ | A ₁₃ | A ₁₄ | A ₁₅ | Aktivan RAM | preko CS ₁ |
|----------------|----------------|----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-------------|-----------------------|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RAM0 | |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | RAM1 | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RAM2 | |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RAM3 | |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RAM4 | |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | RAM5 | |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | RAM6 | |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | RAM7 | |

adresiraju module

A₅ A₄ A₃ A₂ A₁ A₀
P.A : 0001 0000 0000 0000
Z.A : 0001 0011 1111 1111

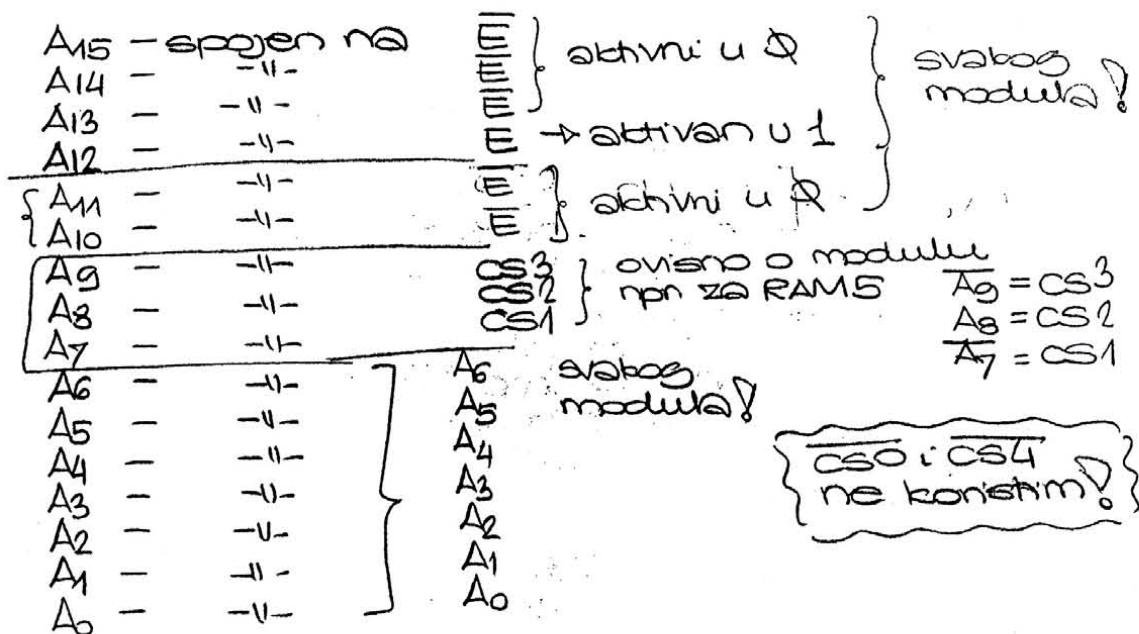
Bududi da A₁₅ A₁₄ A₁₃ A₁₂ A₁₁ A₁₀ moraju biti 0 0 0 1 0 0 da bi RAM bio aktivan, njih demeo spojiti na

ENABLE waze:

| | | |
|---------------------|-----|---|
| A ₁₅ = E | (0) | tek 1Z0 ovu komb. je RAM aktivan? |
| A ₁₄ = E | (0) | |
| A ₁₃ = E | (0) | |
| A ₁₂ = E | (1) | |
| A ₁₁ = E | (0) | |
| A ₁₀ = E | (0) | |

OKRENI 8

Pojašnjenje:



MODUL R₀ (AKTIVAN ZA $A_9=0$ $A_8=0$ $A_7=0$) 128 lokacija!
adresira od $0001\ 00\ 00\ 0000 - 0001\ 00\ 00\ 0111\ 1111$

MODUL R₁ (AKTIVAN ZA $A_9A_8A_7=001$) 128 ✓
adresira $0001\ 00\ 00\ 1000\ 0000 - 0001\ 00\ 00\ 1111\ 1111$

-" R₂ (aktivan $A_9A_8A_7=010$) 128 ✓
 $0001\ 00\ 01\ 0000\ 0000 - 0001\ 00\ 01\ 0111\ 1111$

-" R₃ $0001\ 00\ 01\ 1000\ 0000 - 0001\ 00\ 01\ 1111\ 1111$

-" R₄ $0001\ 00\ 10\ 0000\ 0000 - 0001\ 00\ 10\ 0111\ 1111$

-" R₅ $0001\ 00\ 10\ 1000\ 0000 - 0001\ 00\ 10\ 1111\ 1111$

-" R₆ $0001\ 00\ 11\ 0000\ 0000 - 0001\ 00\ 11\ 0111\ 1111$

-" R₇ $0001\ 00\ 11\ 1000\ 0000 - 0001\ 00\ 11\ 1111\ 1111$

Adresni prostor je ravnomjerno raspodijeljen
(1000-1400) na 8 RAM modula.

Kapacitet (ukupni) = 8×128 rječi od po 8 byte
= $1K \times 8$ byte //

3.09.1999

 $2^{10} = 1024$ AFC 000
000 400KONAČNA
ADRESA(4) RAM 128×8 bit moduli (gradivne jedinice)Oblikuj mem. modul kapaciteta $1K \times 16$ bita \rightarrow 16bitna DATA BUS

Poč. adresa AFC 000 (H)

 $\Rightarrow 6 \cdot 4 = 24$ bitna adres. sab.

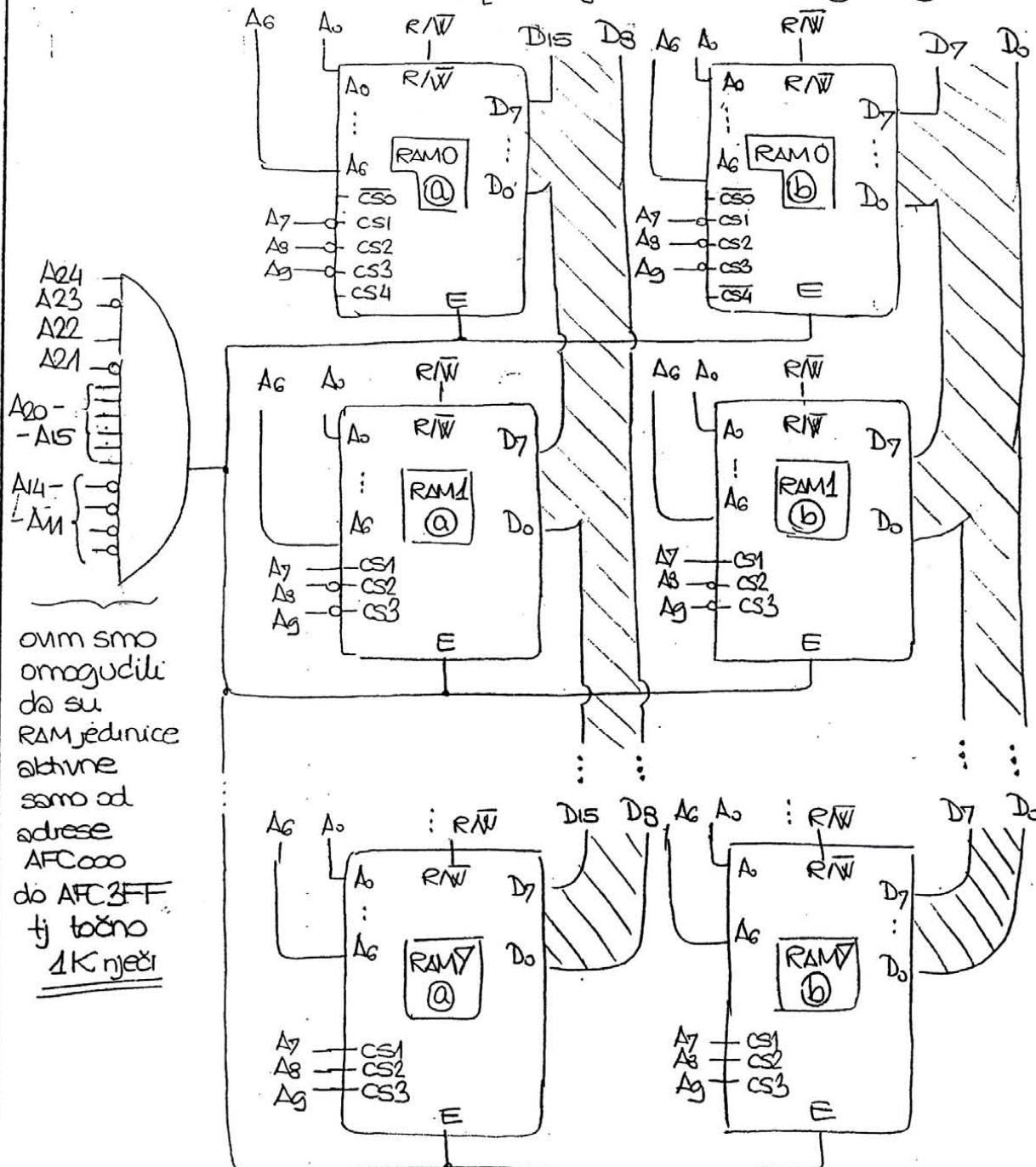
V RAM: linearne adresiranje, R/W,

 $\overline{CS0}, \overline{CS1}, \overline{CS2}, \overline{CS3}, \overline{CS4}$
 $A_7 - A_6 ; D_0 - D_7$

$128 \times 8 = 2^7 \times 8$

Trebaemo $1K \times 16 = 2^{10} \times 2 \cdot 8 = 8 \cdot [128 \times 8] \cdot 2 \rightarrow 16$ RAM grad. jedinice

| FIKSNO | | A ₇ A ₆ | | A ₆ - A ₀ | | PA | |
|--------|------|-------------------------------|------|---------------------------------|------|------|------|
| 1 | 0110 | 1111 | 1100 | 0000 | 0000 | 0000 | 0000 |
| 2 | 1010 | 1111 | 1100 | 0011 | 1111 | 1111 | KA |



Kombinacije

A₇A₆A₅ biraju per modul?

@ dis modula pohranyje D15-D8,

a(b) -" - " - D7-D0

1. kontrolna zadaća iz predmeta
"ARHITEKTURA I ORGANIZACIJA RAČUNALA"

11. prosinca 2000.
grupa A

(1) (2) Napisati program za Turingov stroj koji obavlja operaciju dekrementiranja broja zapisanog na vrpci. Broj je predložen kao troznamenkasti u brojevnom sustavu s bazom 3. Glava za čitanje i pisanje nalazi se na poziciji najznačajnije znamenke.

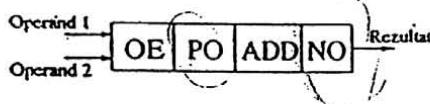
(2) (3) Pretpostavite da je pojednostavljenom modelu mikroprocesora pridodan još jedan registar - akumulator B. Za tako preinačen model nacrtajte stanja na vanjskoj sabirnici za programske odsječak prikazan na slici 1. Programski odsječak je pohranjen u memoriji s početnom adresom \$0100 (slika 2). Napišite stanja registara DC, PC, IR, A, B, te sadržaje promijenjenih memorijskih lokacija nakon izvođenja programa, uz početni sadržaj memorijskih lokacija prikazan na slici 3.

LDA A \$1000 ; op. kod = \$86
LDA B \$1001 ; op. kod = \$96
ABA ; A + B - A, op. kod = \$91
STA A \$1002 ; op. kod = \$87

Slika 1.

PC = 010A
DC = 1002
IR = 87
A = 00
B = F1
1000 OF
1001 F1
1002 AA
1003 96
1004 10
1005 01
1006 91
1007 87
1008 10
1009 02

Slika 3.



Slika 4.

Slika 2.

(3) Nacrtajte binarnu čeliju BC temeljenu na RS bistabilu i uporabite je kao građevnu komponentu za oblikovanje memorijskog modula (u skladu s von Neumannovim modelom memorije) kapaciteta 4 4-bitne riječi. Uz pretpostavku da je adresna sabirnica širine samo 8 bitova, oblikujte adresni dekoder za taj modul, tako da se modul jednoznačno javlja s početnom adresom FC! *Vidi*

(3) Aritmetička jedinica za zbrajanje brojeva s pomičnim zarezom realizirana je kao protočna jedinica s cetiri protočna segmenta:

1. protočni segment za oduzimanje eksponenta (OE);
2. protočni segment za poravnavanje (PO);
3. protočni segment za zbrajanje (ADD);
4. protočni segment za normalizaciju (NO).

Grafički takvu jedinicu možemo predložiti slikom 4. Odredite efektivno vrijeme obrade za jedan par operanada ako se zbraja niz od $n = 4000$ parova operanada i ako su vremena obrade u pojedinim segmentima:

$$\begin{aligned} a) t_{OE} &= t_{PO} = t_{ADD} = t_{NO} = 60 \text{ ns.} \\ \rightarrow b) t_{OE} &= 40 \text{ ns; } t_{PO} = 70 \text{ ns; } t_{ADD} = 50 \text{ ns; } t_{NO} = 80 \text{ ns.} \end{aligned}$$

Ocijenite ubrzanje obrade (izraženo omjerom vremena) za protočnu izvedbu (slučaj a) u odnosu na jedinicu realiziranu kao neprotočna s ukupnim vremenom obrade 240 ns.

(2) Nacrtati Turingov stroj, definirati ga kao n-torku i objasniti k-tu konfiguraciju stroja.

6. (2) Navesti osnovne značajke skalarnog i superskalarnog RISC procesora te tipične predstavnike.

7. (1) Navesti uobičajene mjeru za izražavanje performanse procesora. Objasnite mjeru SPEC mark(s).

(2) Nacrtati detaljan dijagram stanja PRIBAVI i IZVRŠI (interpretacijski dijagram).

9. (2) Za SRISC (Simple RISC) model procesora napisati programske odsječak koji će zbrojiti sadržaje memorijskih lokacija M1 i M2 i rezultat pohraniti na memorisku lokaciju M3. Uz pretpostavku da je SRISC ostvaren kao protočni procesor s protočnim segmentima IF, ID, EX, ME i WB prikazati modificirani kod programske odsječke (kojeg će generirati programski prevodilac) u cilju dobivanja ispravnog rezultata. Obratite pozornost da se operandi upisuju u registre tek u segmentu WB.

~~LD 21, M1~~
~~LD 22, M2~~
~~ADD 23, 21, 22~~
GT 23, M3

Samo je jedan Mali Ivica!

PISMENI ISPIT IZ ARHITEKTURE I ORGANIZACIJE RAČUNALA I
21.02.1997.

1) Za pojednostavljeni model mikroprocesor odredi:

a) Stanje na sabirnicama za

LDA \$10FF; napunj A = ~~00~~ [\\$]
BZ TARGET; Branch if zero TARGET
NOP;

...
TARGET: INC \$10FF

0100: B6; LDA;
0101: 10;
0102: FF;
0103: 26; BZ
0104: 08; TARGET
0105: 01; NOP
...
TARGET 7C; INC
10
FF
10FF: 00

b) Konačni sadržaj registara

c) Završni sadržaj memorije ako proc. koristi
drugi osnovni način uređenja slijeda byteova.

1 NAČIN: BIG ENDIAN

2 NAČIN? Vredna obrnuta od njega
veda adresa = byte s vredom težinom.

2. Nacrtaj dijagram toka i mikroprogram za fazu izvrši instrukcije SHLF (posmak akumulatora ulijevo za 4). Izvođenje instrukcije ne mijenja zastavice reg. stanja. Kako se na temelju op. koda instrukcije određuje adresa prve mikroinstrukcije?

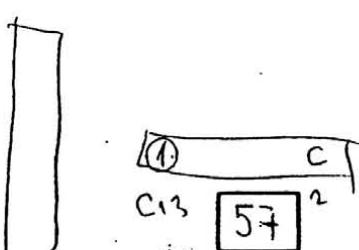
3) Nacrtati kombinacijsko sklopolje upravljačke jedinice za 8-instrukcijsko računalo za fazu pribavi.

4) Objasnitи zbrajalo s predviđanjem bita prijenosa i nacrtati 4-bitnu izvedbu.

5) Za model protočne jedinice od M segmenata, procijenite vrijeme obrade para operanada za $N \gg 1$ (N je broj parova. Trajanje obrade u i-tom segmentu je T_{si} , ali ne vrijedi da je $T_{s1}=T_{s2}=\dots=T_{smax}$ (trajanja obrada nisu jednaka).

Svi zadaci nose 4 BODA.

611



Samo je jedan Mali Ivica!

PISMENI ISPIT IZ ARHITEKTURE I ORGANIZACIJE RAČUNALA I
07.02.1997.

1. Na slici 1. prikazan je "slijepi" dijagram stanja na sabirnicama za vrijeme sabirničkog ciklusa potvrde prekida za MC 68000. Detaljno opišite obradu prekida (**2 BODA**) i na "slijepom" dijagramu označite pojedine signale i događaje (**3 BODA**).
2. Model 8-instrukcijskog procesora (sl.2) preinačen je tako da mu je pridodana instrukcija LSHIFT (posmak ulijevo). Na razini mikrooperacije ona se aktivira dodatnim upravljačkim signalom C13 (LEFT-SHIFT AC).
 - a) Nacrtati strukturu upravljačke jedinice i točno označiti na koje sklopove utječe (i kako) gornja preinaka (**3 BODA**).
 - b) Nacrtati logičku strukturu I-ILI, odnosno PLA kojom se ostvaruje faza IZVRŠI za instrukciju LSHIFT (**2 BODA**).

2) Za model mikroprocesora nacrtati stanja na sabirnicama tijekom izvođenja programskog odsječka (**2 BODA**):

LDA A \$0201
COM A ;jedični komplement

Programski odsječak započinje na adresi A000H i ima oblik prikazan na slici 3. Odgovoriti na pitanja:

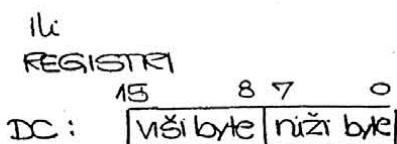
- a) Kakvo je stanje zastavica C, Z i N u statusnom registru nakon izvođenja instrukcije COM A ? (**1 BOD**)

PITATI 2) b) Kakvo uređenje slijeda bajtova koristi model mikroprocesora. (**1 BOD**)
BIG INDIAN : na manjoj adresi se pohranjuju bitovi veće težine

4. Opišite uobičajeni pristup oblikovanju aritmetičko-logičke jedinice (3 koraka), a zatim prikažite kako se oblikuje aritmetička sekcija za n stupnjeva aritmetičke jedinice. Nacrtajte sklop za pribavljanje bita operanda B i objasnite (ukratko) njegovu namjenu. (**4 BODA**)

5. Shematski prikažita dvije osnovne organizacije RAM modula i definirajte logičku i fizičku riječ. Odgovorite: Kakav je odnos između logičku i fizičke riječi u gornjim organizacijama. (**2 BODA**)

SUKA 1: knjiga "Naprednije arhitekture računala"
pol. 7. "Prekidni sustav mikroprocessora"
str.



I. KOLOKVIJ IZ ARHITEKTURE I ORGANIZACIJE RAČUNALA I

11.12.1996

1. Za pojednostavljeni model mikroprocesora nacrtajte stanje na sabircicama tijekom izvođenja sljedećeg programskog odsječka:

OPET: DEC A ; umanji sadržaj akumulatora
 BNE OPET ; granaj na OPET ako je rezultat prethodne operacije različit od nule (zastavica Z=1)

Početni sadržaj akumulatora A je 01H. Programski odsječak je pohranjen u memoriji s početnom adresom 1000H:

1000H: 4AH ; operacijski kod DEC A
 1001H: 26H ; op.kod BNE
 1002H: FDH ; relativna adresa

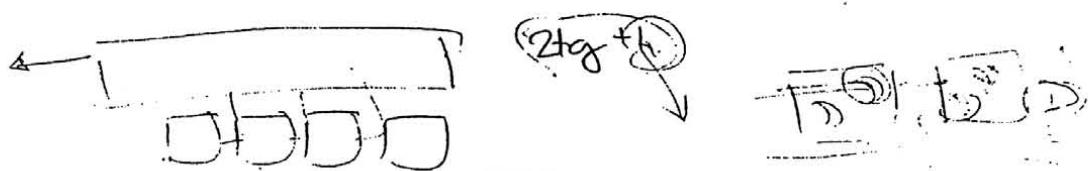
2. Skup od 8 instrukcija za model procesora čiju smo upravljačku jedinicu ostvarili sklopovski, proširujemo instrukcijama SHL M (posmak uljevo sadržaja memoriske lokacije), i COM M (jedinični komplement sadržaja memoriske lokacije M). Objasnite na koje sve komponente modela procesora utječe uvođenje tih dvaju instrukcija, te skicirajte potrebne izmjene.

3. Na primjeru jednostavnog rekurzivnog programa pokažite kako stožna struktura podržava njegovu prikazivanje.

PRIMATI

4. Navedite i objasnite osam arhitektonskih značajki, te navedite njihove vrijednosti za klasifikaciju arhitektura procesora CISC i RISC.

5. Nacrtati i objasniti detaljan dijagram stanja izvođenja instrukcija i na temelju njega predložiti protočne segmente upravljačke jedinice.



$$\text{sa : } \frac{2tg + tn}{2 \cdot n \cdot tg + tn} = \frac{3tg}{(2n+1)tg} = \frac{2n+1}{3} \rightarrow tn + 2tg + tg$$

$$\text{F sa : } \frac{2tg + tg + tn}{2 \cdot n \cdot tg + tn} = \frac{4+tg}{(2n+1)tg}$$

1111

59

Samo je jedan Mali Ivica!

TEORIJA:

- 1) P(2) u uprogramirajuju služi za
 - a) faze u izvrši
 - ⇒ b) faze u upravljanju
 - c) prenos se adresa stred. μI
- 2) Stogovi u MC68000 su
 - a) USP i SSP 16 bitovni
 - b) USP 16 bitni, SSP 32 bitni
 - ⇒ c) USP i SSP 32 bitovni
- 3) Komb. sklopovi → brojilo do n u sklopovskoj izvedbi
služi za
 - a) detekciju preliva
 - ⇒ b) -||- preliva i podliva
 - c) -||- podliva
 - d) realizaciju PUSH i POP
- 4) Za rezultirati stog dubine 32 16-bitne ječi potrebna su
 - ⇒ a) 16 32-bitnih registera
 - b) 32 16-bitne -||-
 - c) obnuta FIFO struktura
- 5) Stack Pointer za MC68000 za rastudij stog ide prema
 - a) rastudijem adresama
 - ⇒ b) podejedinim -||-
 - c) avisno o zastavci GS
- 6) Navedi tehnike za oblikovanje mješići:
 - a)
 - b)
 - c)
 - d) (moraš napisati formate mješići)
 - grupiranje bitova
 - horz/vert. uprogramiranje
 - krovno
 - višestruke mješići
- 7) Napiši formule za zbrojalo (FA):

$$F =$$

$$C =$$
- 8) Bločni povezivac se sastoji od $\frac{i}{(\text{značaj})} \frac{j}{(\text{blok})}$
- 9) Povratak iz konstruktorog načina rada dešava se:
 - a) samo RESET naredbom
 - b) -||- RTE -||-
 - c) -||- iznimbom
 - d) -||- izvodenjem privilegirane naredbe
- 10) Obrada iznimke za MC68000 se dešava
 - a) U konstr. načinu rada
 - b) U nadglednom -||- -||-
 - c) avisno u kojem modu je izninka nastala

Samo je jedan Mali Ivica!

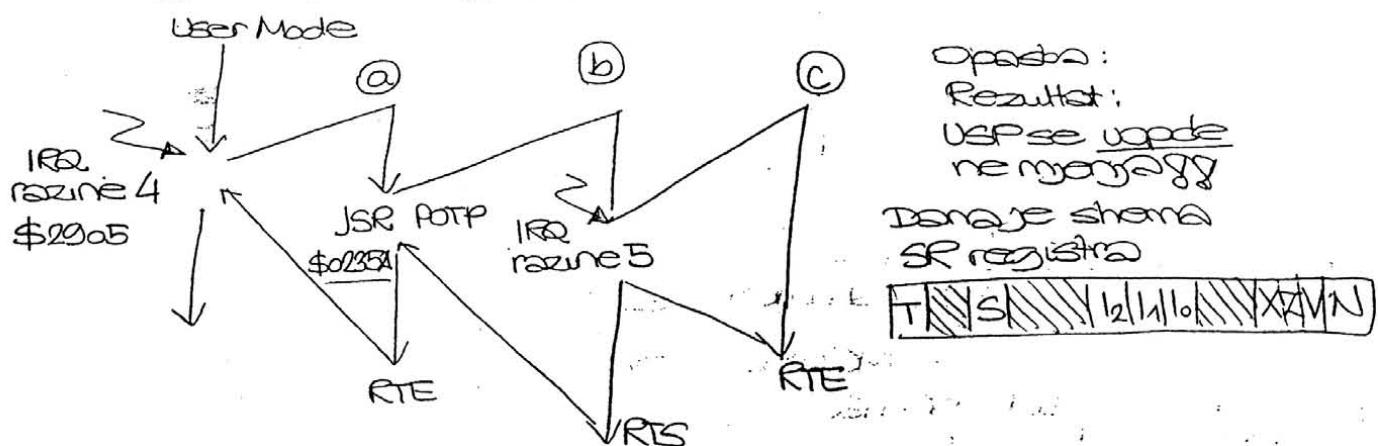
PRAKTIKUM

- 1) Upravlja jedinicu za μ SS instr.
Dodataj je instr. DEC M i C13 upravljački signal koji decrementeira MDR.
 - Nacrtaj realizaciju upravljačke jedinice sa svim izmjenama koje donosi nova DEC M instr.
 - Nacrtaj na PLA za fazu zvrsi te instr.
 - Ispiši log. jednadžbu

- 2) Shemski za MC68000:

USP na adresi A20348H (nismo zapamtili)
SSP -11- -1L C02A0AH

Nacrtaj sljedeća stogovima: SR registre 20 a), b), c); d)



- 3) Realizirati 3bitno paral. zbrojalo (pomoći 2 HA svaki FA)
[da se svaki sklop vidi]

- 4) Realiziraj sklop 2x CARRY LOOK AHEAD
Izvedi jednadžbe za C_4, C_3, C_2 , i onda slobur @
spoji sa sklopm(b) (shematski)

- 5) Realiziraj MUXevirna sklop 2x posmat (4bitovni) koji radi slijedeće (tablica)

| S ₂ | S ₁ | S ₀ | |
|----------------|----------------|----------------|-------------------------|
| 0 | 0 | 0 | nema pomaka |
| 0 | 0 | 1 | pomak za 1 udesno |
| 0 | 1 | 0 | -11- |
| 0 | 1 | 1 | -11- 1 uljevo |
| 1 | 0 | 0 | -11- 2 -11- |
| 1 | 0 | 1 | sve jedinice posmatrati |
| 1 | 1 | 0 | sve nule posmatrati |
| 1 | 1 | 1 | njezinu se |

- 5) Realiziraj instrukciju:

Ako $B > -8$ onda $PC = PC + 8$

Ako $B = -8$ onda $PC = PC + 16$

Ako $B < -8$ onda $PC = PC$

Premašće 30 bitima, tako da OEM ima 6 bitova.

Samo je jedan Mali Ivica!

11.) Vektorski prekid u μP je dobar jer vektor :

- a) generira ga μP kad se desi zahtjev za prekid
- b) generira ga uređaj koji zahtjeva prekid, s tim da svaki novi prekid generira vektor za I vidi
- c) - II - - II - - I - - II - - II - - II -, s tim da je vektor za A prekid jednoznačno određen
- d) nalazi se u v brzoj cache memoriji i prelazi u PC na zahtjev za prekid

12.) Upisi kolike sve podjele blokova memorije postaju :

- a)
- b)
- c) [na stranice
na segmente
na - II - s stranicama]

13.) Kad potpuno asocijativno preslikavanje, blok iz gl. memorije se preslikava u :

- a) blok cache-a na adresi $j = 2^i / B_p$
- b) - II - - II - - II - $j = i \bmod B_p$
- c) bilo koji tko je slobodni blok cache-a
- d) na bilo koju adresiranu adresu

14.) Za podržavanje rekurzije, potrebna je

- a) stogova struktura (LIFO + Pointer)
- b) FIFO struktura
- c) registri za pohranu min. konteksta

15.) Za realizaciju upravlј. jedinice sklopovljem, konstisti se komb. sklop pravilne strukture :

- a) PAL
- b) PLA
- c) PLEA
- d) PBI

Samo je jedan Mali Ivica!