Univerzitet u Kragujevcu
TEHNIČKI FAKULTET
Čačak

SPECIFIKACIJA ARHITEKTURE PROCESORA TFACO

Verzija 4.01
Interni izveštaj

Laboratorija za računarsku tehniku

Čačak
2012. godine

SPECIFIKACIJA ARHITEKTURE PROCESORA TFACO

Instrukcije procesora TFCo su fiksne dužine 16 bita. Osnovni format instrukcije je prikazan na slici 1.

Slika 1. Osnovni format instrukcije procesora TFaCo

Shodno usvojenom formatu instrukcije arhitektura procesora TFaCo ima sledeće karakteristike:

- Dužina operacionog dela instrukcije (COP) je **4 bita** (**I_{12} – I_{15}**).
- Funkcija ostalih bitova instrukcije (**I_0 – I_{11}**) zavisi od vrste instrukcije i primenjenog načina adresiranja.
- Dužina procesorske reči je **16 bita**, a istu dužinu ima i **memorijska reč**, čija je dužina prilagođena dužini instrukcije, koja se na taj način može dohvatati u okviru jednog memorijskog ciklusa.

Arhitektura procesora TFaCo je RISC (Reduced Instruction Set Computer) orijentisana pa procesor ima registar fajl od **16 registara,** koji su programski dostupni. Registri **R_0 – R_7** su registri opšte namene, a registri **R_8 – R_{15}** imaju i specijalne namene, kao što je prikazano u tabeli 1.

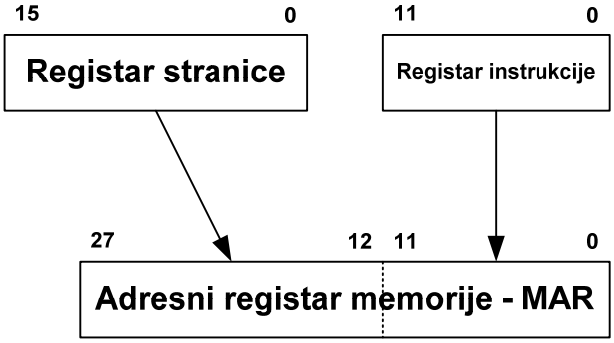
Tabela 1. Funkcije specijalnih registara

R_8	Akumulator (RA)
R_9	Statusni registar (RS)
R_{10}	Registar skoka (RJ)
R_{11}	Registar programske stranice (RC)
R_{12}	Registar stranice podataka (RD)
R_{13}	Indeks registar (RX)
R_{14}	Bazni registar (RB)
R_{15}	Programski brojač (PC)

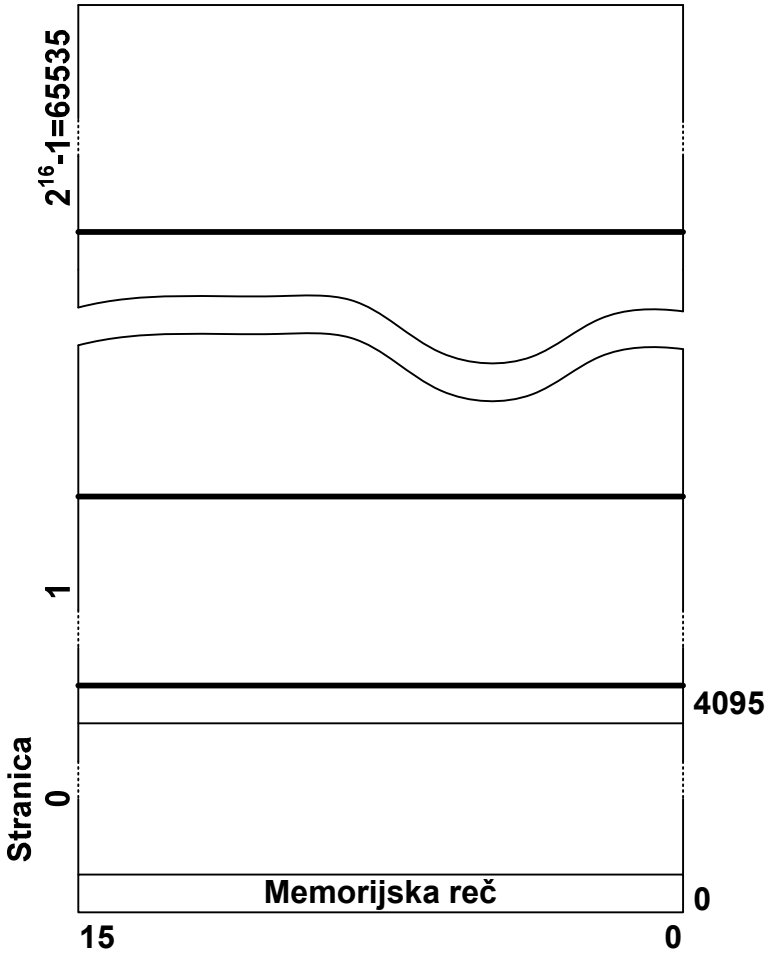
Načini adresiranja

Procesor TFaCo omogućava sledeće načine adresiranja

1. **Direktno memorijsko adresiranje** – Kod ovog načina adresiranja bitovi instrukcije **0 – 11** predstavljaju adresu adresibilne jedinice (memorijske reči dužine 16 bita) unutar stranice od **2^{12}** (4096) lokacija. Adresa stranice je definisana sadržajem **Registra stranice** (R_{11} – **RC** i R_{12} – **RD**) i memorijska adresa se dobija spajanjem sadržaja segment registra i adresnog dela instrukcije. Na taj način memorijska adresa TFaCo ima dužinu **16** (definiše jednu od 2^{16} stranica) plus **12** (definiše memorijsku reč unutar stranice) bita, što ukupno iznosi **28 bita**. Način formiranja izvršne adrese prikazan je na slici 2. Ukupan broj adresibilnih jedinica (memorijskih reči dužine 16 bita) memorije sistema TFaCo iznosi **2^{28}** memorijskih reči ili **256MB**. S obzirom da je memorijska reč dugačka 2 bajta onda je maksimalni kapacitet ove memorije **2^{29}** (**512MB**) bajta. Struktura memorije sistema TFaCo je prikazana na slici 3. Shodno ovome dužina **memorijskog adresnog registra** (MAR) iznosi 28 bitova. Ovaj način adresiranja primenjuje se kod instrukcija kojima se vrši prenos podataka između **akumulatora** i **memorije**.
2. **Registarsko adresiranje** – Ovaj način adresiranja implicira da se sve aritmetičko – logičke instrukcije izvršavaju nad sadržajima registara. Ovde postoje dva podformata, jedan se odnosi na troadresne instrukcije, a drugi su dvoadresne instrukcije u koje spadaju instrukcije kojima se vrši prenos podataka između **akumulatora** i **registara.** Formati instrukcije kod ovog načina adresiranja prikazani su na slici 4.

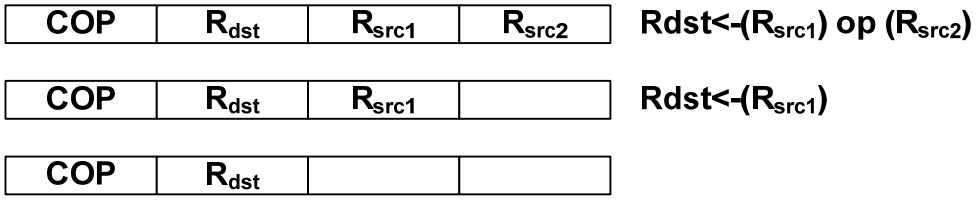


Slika 2. Način formiranja izvršne memorijske adrese

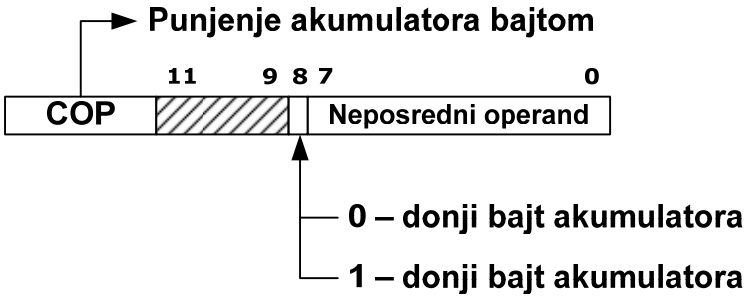


Slika 3. Struktura operativne memorije

3. **Neposredno adresiranje** – Ovaj način adresiranja za punjenje akumulatora neposrednom vrednošću operanda. Usvojeno je da dužina neposrednog operanda bude 8 bita (1 bajt). U tom slučaju bitovi instrukcije I_8 – I_{11} ostaju neiskorišćeni. S obzirmo, da je dužina akumulatora 16 bita ova instrukcija može da se iskoristi da se bira u koji bajt akumulatora će se upisati neposredni operand. Za tu namenu se koristi bit 8. Ako je $I_8 = 0$ tada se neposredni operand upisuje u donji bajt akumulatora, a ako je $I_8 = 1$ tada se neposredni operand upisuje u gornji bajt akumulatora, kao što je prikazano na slici 5.



Slika 4. Registarski načini adresiranja

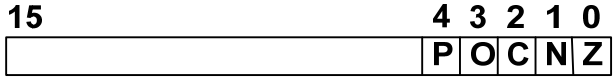


Slika 5. Neposredno adresiranje

4. **Instrukcije skoka** – Skokovi kod TFaCo mogu biti bezuslovni ili uslovni. Uslovni skokovi se realizuju na bazi statusnog registra RS (R_9) u koji se po izvršavanju operacija upisuje odgovarajuća statusna informacija, prema tabeli 1 i slici 6.

Tabela 1. Značenje bitova u registru statusa – R_9

Bit	Oznaka	Značenje
R_9 (0)	**Z – Zero Bit**	
	Z=0	Rezultat operacije različit od 0
	Z=1	Rezultat operacije jednak 0
R_9 (1)	**N – Negative Bit**	
	N=0	Rezultat operacije pozitivan
	N=1	Rezultat operacije negativan
R_9 (2)	**C – Carry Bit**	
	C=0	Nema prenosa iz najstarijeg razreda rezultata
	C=1	Postoji prenos iz najstarijeg razreda rezultata
R_9 (3)	**O – Overflow Bit**	
	O=0	Nema prekoračenja opsega računanja
	O=1	Došlo je do prekoračenja opsega računanja
R_9 (4)	**P – Parity Bit**	
	P=0	Broj 1 u rezultatu paran
	P=1	Broj 1 u rezultatu neparan



Slika 6. Struktura statusnog registra RS (R_9)

Instrukcija skoka ima jedinstveni COP, a test za proveru uslova za izvršenje skoka definiše se postavljanjem odgovarajućeg bita u adresnom formatu u skladu sa sadržajem statusnog registra, kao što je prikazano na slici 7. U slučaju da se postavi bit **J**, tj. **J=1** instrukcija će se izvršiti kao bezuslovni skok. Izvor adrese skoka uvek je sadržaj **registra skoka (R_{10}).**

Slika 7. Format instrukcije skoka

Skup registara i njihova funkcija

Skup registara sastoji se od 16 registara, svaki dužine 16 bita. Skup je podeljen u dva bloka registara

- Registri opšte namene (**R_0 – R_7**)
- Registri specijalne namene (**R_8 – R_{15}**) – svaki registar ima određenu namenu. Registri su programski dostupni tako da u određenim slučajevima mogu da budu korišćeni i kao registri opšte namene

- **R_8 – Akumulator (RA)** – Registar se koristi kao implicitno odredište/izvorište podatka za potrebe razmene informacija sa memorijom.
- **R_9 – Statusni registar (RS)** – U ovaj registar se po okončanju svake operacije upisuje odgovarajuća informacija o statusu izvršene operacije u skladu sa pravilima datim u tabeli 1.
- **R_{10} – Registar skoka (RJ)** – U ovom registru se nalazi adresa (12 bita) po kojoj se vrši prelazak (skok) u programu: Punjenje odgovarajućim sadržajem (adresom skoka) može da se realizuje:
  - Unosom neposrednih operanada
  - Prenosom sadržaja nekog od registara
  - Čitanjem sadržaja memorije preko akumulatora
- **R_{11} – Registar programske stranice (RC)** – Registar koji sadrži početnu adresu jedne od stranica u memoriji u kojima se nalaze instrukcije. Puni se na jedan od načina kao i registar skoka ili izvršavanjem određene arutmetičke operacije.
- **R_{12} – Registar podataka stranice (RD)** – Registar koji sadrži početnu adresu jedne od stranica u memoriji u kojima se nalaze podaci. Puni se na jedan od načina kao i registar skoka ili izvršavanjem određene arutmetičke operacije.
- **R_{13} – Indeks registar (RX)** – Registar koji se koristi za indeksiranje sadržaja nizova pri relativnom adresiranju
- **R_{14} – Bazni registar (RB)** – Koristi se kod relativnog adresiranja, kao registar baze.
- **R_{15} – Programski brojač (PC)** – Definiše adresu instrukcije u slučaju konsekutivnog izvršavanja instrukcija.

Skup instrukcija TFaCo i opis njihovog izvršavanja

COP	Simbolički kod	Sintaksa	Opis
0000	**NOP**	NOP	**Operacija bez dejstva**
0001	**LI**	LI lb, podatak	**Operacija punjenja akumulatora neposrednim operandom** ▪ **lb** – Jednobitna informacija, koji se nalazi na lokaciji **I_8**, i koja definiše u koji bajt se upisuje neposredni operand, koji je dužine 1 bajt. o **lb=0** – podatak se upisuje u donji bajt akumulatora o **lb=1** – podatak se upisuje u gornji bajt akumulatora ▪ **podatak** – 8 – bitna informacija. Pomoću ove naredbe, u opštem slučaju, mogu da se formiraju sledeći podaci u akumulatoru: o Neoznačena veličina dužine 16 bita o Označena veličina dužine 16 bita o Binarni broj dužine 16 bita o Četiri heksadecimalna broja o Dva ASCII karaktera o Jedan Unicode karakter
0010	**LOAD**	LOAD adresa	**Puni akumulator podatkom iz memorije** ▪ **adresa** - memorijska adresa sa koje se čita podatak
0011	**STORE**	STORE adresa	**Pamti sadržaj akumulatora** ▪ **adresa** - memorijska adresa na koju se upisuje podatak
0100	**ADD**		**Sabiranje** r3←(r1)+(r2) – Funkcionalni format instrukcije za sabiranje ▪ r3 – Broj odredišnog registra ▪ r2 – Broj registra drugog operanda ▪ r1 – Bbroj registra prvog operanda
0101	**SUB**		**Oduzimanje** r3←(r1)-(r2) – Funkcionalni format instrukcije za oduzimanje ▪ r3 – Broj odredišnog registra ▪ r2 – Broj registra drugog operanda ▪ r1 – Broj registra prvog operanda

COP	Simbolički kod	Sintaksa	Opis
0110	**MOV**	MOV r3,r1	**Kopiranje sadržaja jednog registra u drugi registar** acc←(r)
0111	**NOT**	NOT r3, r1	**Logička negacija** r3←not(r1)
1000	**OR**		**Logičko ILI** r3←(r1) or (r2) – Funkcionalni format instrukcije za logičko ILI ▪ r3 – Broj odredišnog registra ▪ r2 – Broj registra drugog operanda ▪ r1 – Broj registra prvog operanda
1001	**AND**		**Logičko I** r3←(r1) or (r2) – Funkcionalni format instrukcije za logičko I ▪ r3 – Broj odredišnog registra ▪ r2 – Broj registra drugog operanda ▪ r1 – Broj registra prvog operanda
1010	**XOR**		**Ekskluzivno ILI** r3←(r1) or (r2) – Funkcionalni format instrukcije za ekskluzivno ILI ▪ r3 – Broj odredišnog registra ▪ r2 – Broj registra drugog operanda ▪ r1 – Broj registra prvog operanda
1011	**SHIFT**	SHIFT ind,r,np	**Pomeranje podatka levo/desno** acc←shift(r) ▪ **ind** – Jednobitna informacija koja se nalazi na poziciji $\mathbf{I_8}$ i koja definiše smer pomeranja o **ind=0** - pomeranje levo, o **ind=1** - pomeranje desno ▪ **r** – Broj registra čiji se sadržaj pomera, pri čemu je odredište rezultata uvek akumulator ▪ **np** – Broj bitova za koji se vrši pomeranje. Četvorobitna informacija koja zauzima pozicije $\mathbf{I_0 – I_3}$.
1100	**JUMP**	JUMP maska	**Instrukcija skoka** Adresa skoka se nalazi u registru skoka (RJ). ▪ **maska** – Definiše test za proveru uslova za skok u programu. Dušina maske je 6 bitova o **maska=000001** – provera na nulu (Z=1) o **maska=000010** – provera na negativno (N=1) o **maska=000100** – provera na prenos (C=1) o **maska=001000** – provera na prekoračenje (O=1) o **maska=010000** – provera na parnost (P=1) o **maska=1xxxxx** – bezuslovni skok
1101	**IN**	IN adresa	**Puni akumulator podatkom iz I/O uređaja** ▪ **adresa** - adresa I/O porta sa koga se čita podatak
1110	**OUT**	OUT adresa	**Šalje podatak iz akumulatora u I/O uređaj** ▪ **adresa** - adresa I/O porta na koji se šalje podatak
1111	**HALT**	HALT	**Zaustavljanje programa**

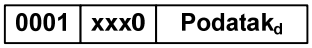
Opisi binarnog formata instrukcija TFaCo

Pri objašnjavanju načina kodovanja instrukcija koristiće se sledeća pravila za označavanje

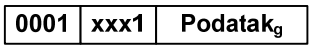
- **0 –** Vrednost bita jednaka 0
- **1 –** Vrednost bita jednaka 1
- **x –** Vrednost bita nije bitna

NOP – Bez operacije

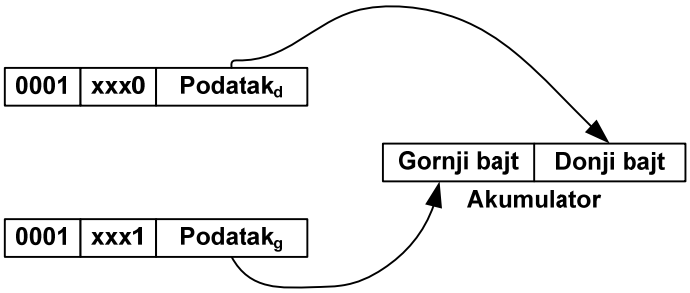
LI – Punjenje akumulatora neposrednim operandom



Upis podatka u donji bajt akumulatora

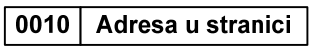


Upis podatka u gornji bajt akumulatora

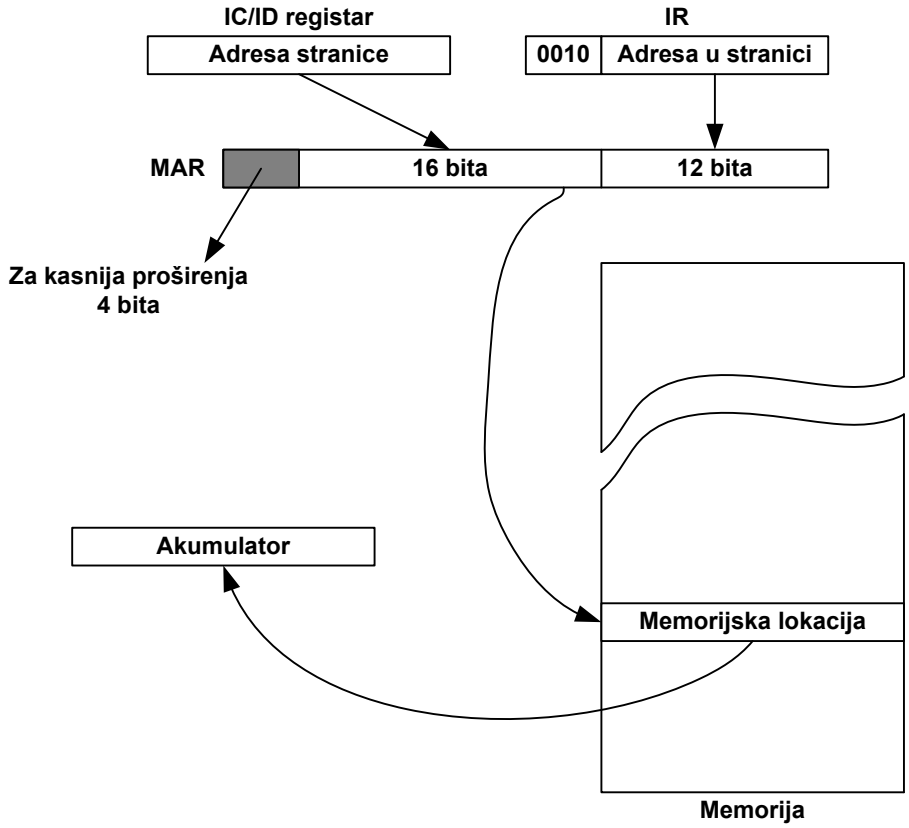


Način izvršenja LI instrukcije

LOAD – Punjenje akumulatora sadržajem memorije



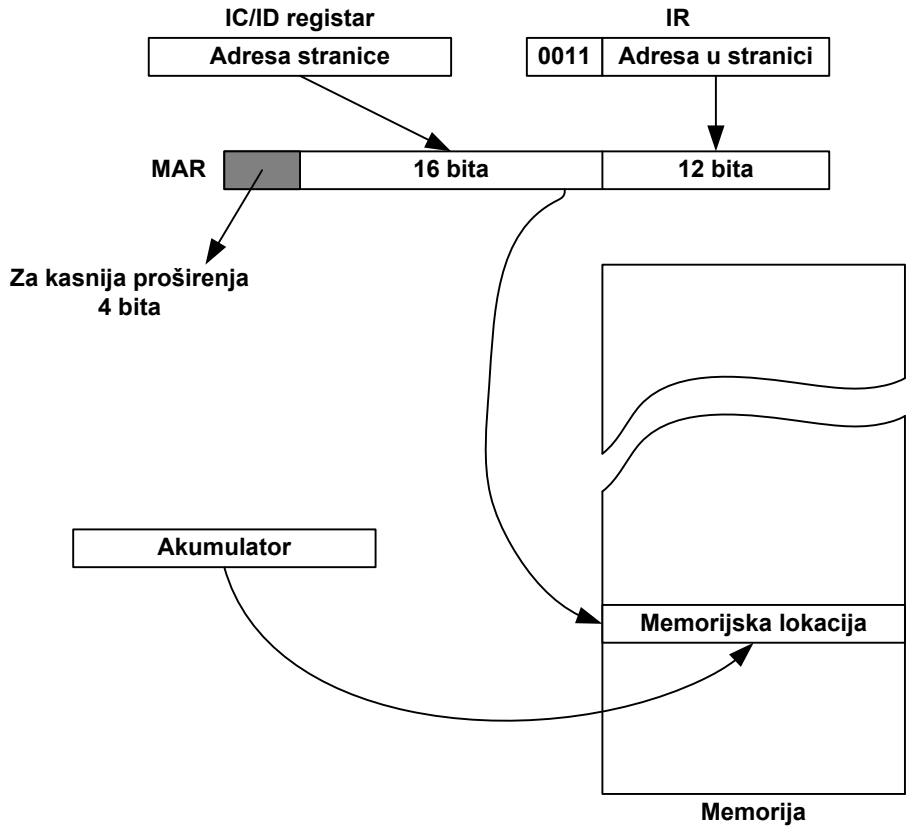
Punjenje akumulatora sadržajem adresirane memorijske lokacije – specificirana adresa odgovara memorijskoj lokaciji sa koje se čita podatak



Način izvršavanja instrukcije LOAD

STORE – Pamćenje sadržaja akumulatora u memoriji

Pamćenje sadržaja akumulatora u adresiranoj memorijskoj lokaciji – specificirana adresa odgovara memorijskoj lokaciji na kojoj se pamti podatak



Način izvršenja instrukcije STORE

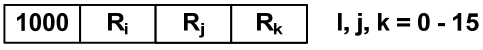
ADD – Sabiranje

Format instrukcije ADD

SUB – Oduzimanje

Format instrukcije SUB

OR – Logičko ILI



Format instrukcije OR

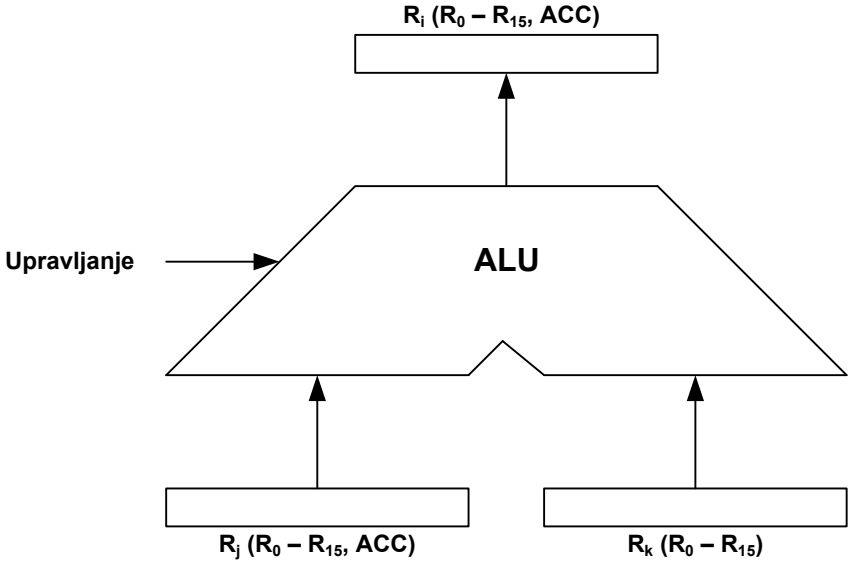
AND – Logičko I

Format instrukcije AND

XOR – Ekskluzivno ILI

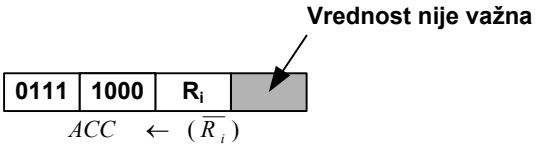
Format instrukcije XOR

Aritmetičko – logičke instrukcije ADD, SUB, OR, AND i XOR izvršavaju se prema šemi na sledećoj slici:

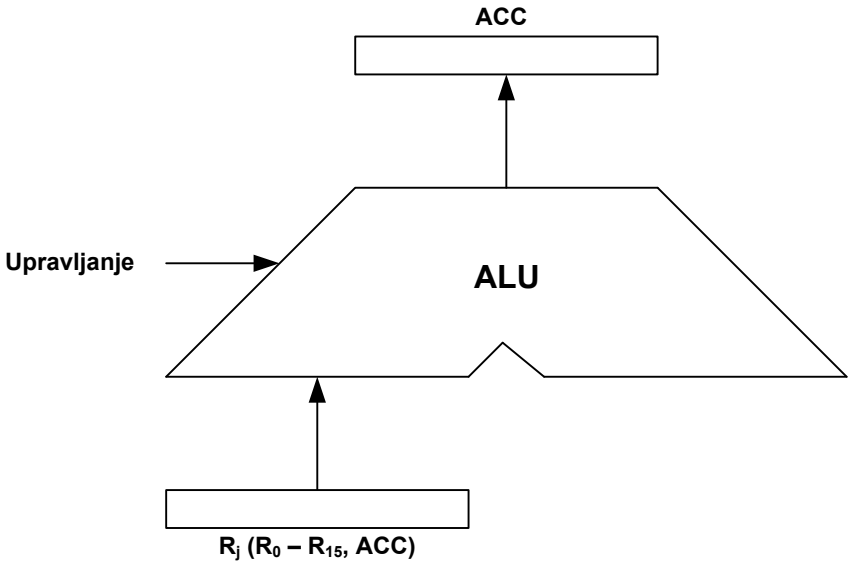


Šema izvršavanja aritmemtičko – logičkih operacija

NOT – Logička negacija

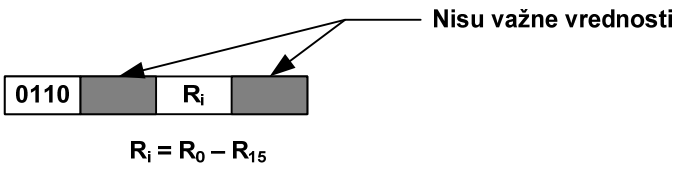


Format instrukcije NOT

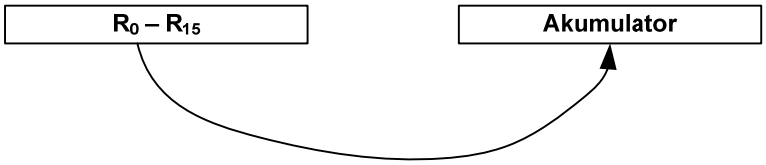


Šema izvršavanja instrukcije NOT

MOV – Kopiranje sadržaja jednog registra u drugi registar

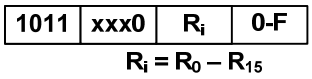


Format instrukcije MVAC

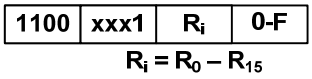


Šema izvrešenja instrukcije MVAC

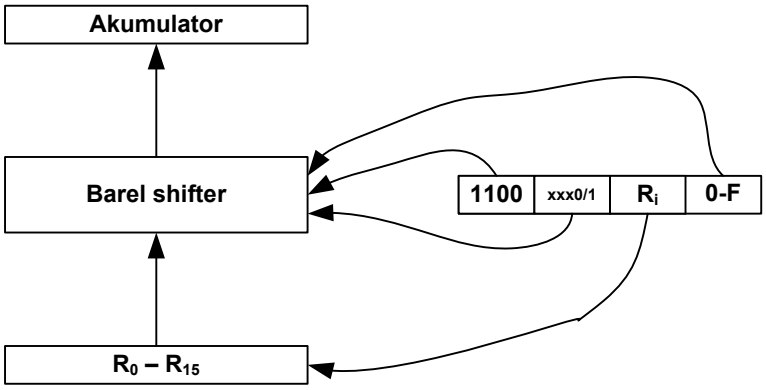
SHIFT – Pomeranje sadržaja registra u levo/desno



Format instrukcije SHIFT kod pomeranja sadržaja u levo

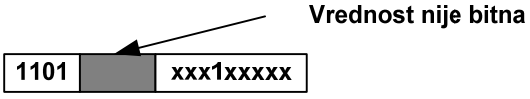


Format instrukcije SHIFT kod pomeranja sadržaja u desno

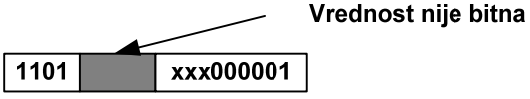


Šema izvršenja instrukcije SHIFT

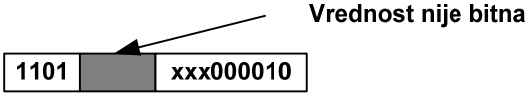
JUMP – Uslovni/bezuslovni skok



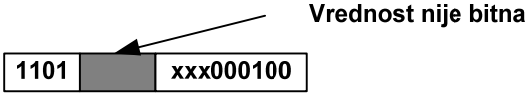
Format instrukcije bezuslovnog skoka



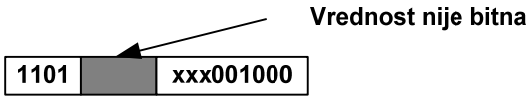
Skok ako je rezultat jednak 0



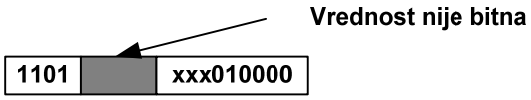
Skok ako je rezultat negativan



Skok ako je generisan prenos

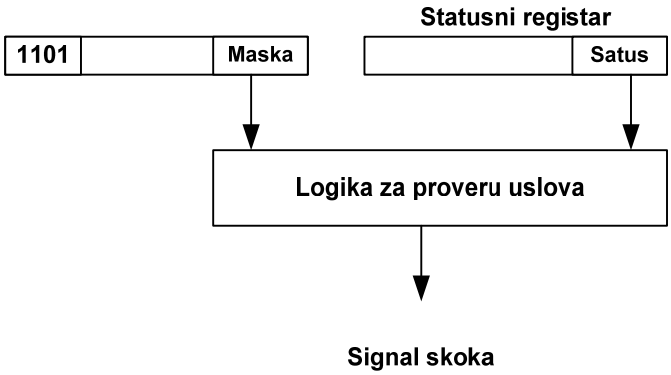


Skok ako je došlo do prekoračenja opsega računanja



Skok po parnosti

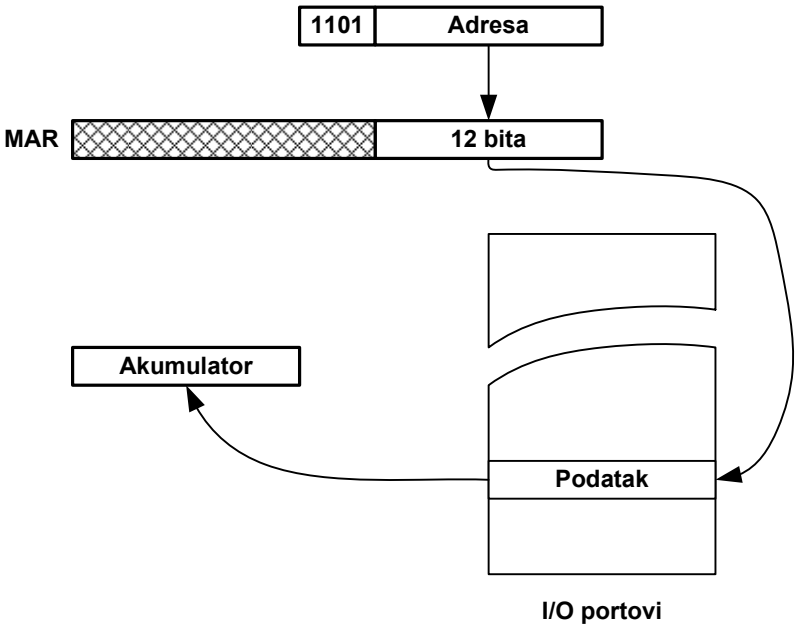
Formati uslovnog skoka



Šema izvršenja instrukcije skoka

IN – Punjenje akumulatora podatkom iz I/O uređaja

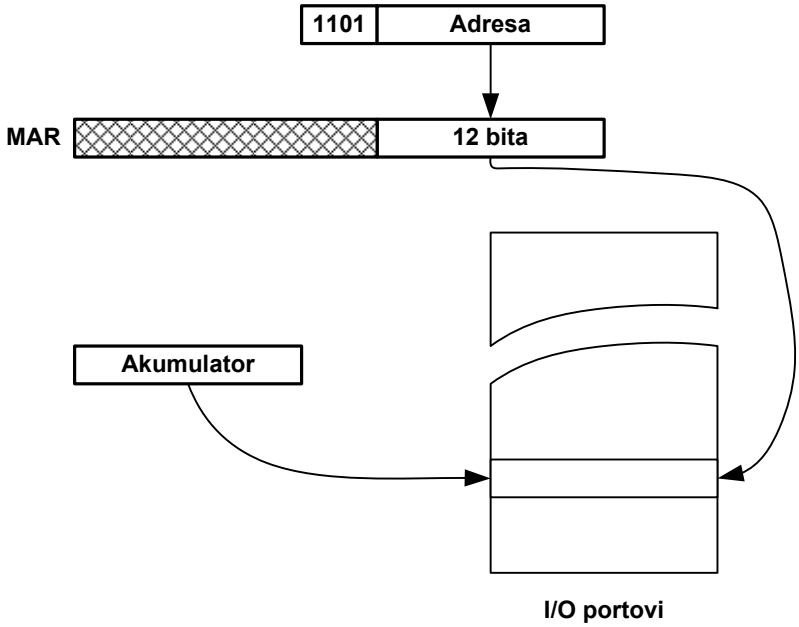
Punjenje akumulatora sadržajem adresiraog I/O porta



Način izvršavanja instrukcije IN

OUT – Slanje sadržaja akumulatora u I/O uređaj

Slanje sadržaja akumulatora na adresirani I/O port



Način izvršenja instrukcije OUT

PRIKAZ MIKROPERACIJA INSTRUKCIJA TFACO

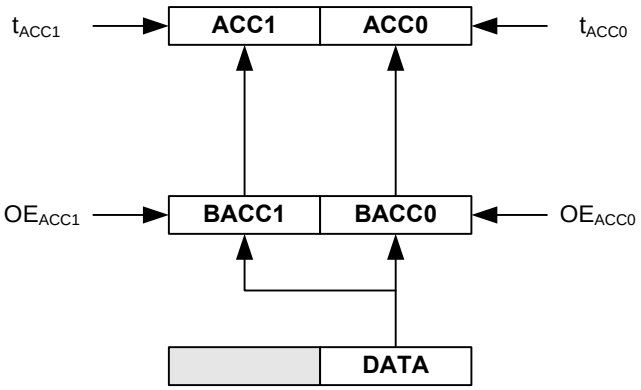
NOP – Operacija bez dejstva

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
		Instrukcija ne menja sadržaj statusnog registra		
	Dekodovanje operacije			
	Čeka se određeni broj procesorskih taktova pre nego se započne sa izvršavanjem sledeće instrukcije			

LI – Punjenje akumulatora neposrednim operandom

Neposredni operand se pre unošenja u akumulator nalazi u registru instrukcije (IR) i dužine je 8 bita. Ovaj podatak može da se upiše u akumulator na nižu ili višu poziciju, što znači da je potrebno izvršiti odgovarajuće demultipleksiranje. U principu ovde može da se napravi proširenje dejstva ove naredbe tako što bi se omogućila operacija umetanja, tj. da se drugi deo sadržaja akumulatora ne menja. Demultipleksiranje se realizuje postavljanjem dvostrukog bafera sa izlazom sa tri stanja na izlaz registra instrukcije IR7 – IR0. U zavisnosti od vrednosti bita **lb** otvara se izlaz bafera koji propušta informacije iz IR na ulaz donjeg ili gornjeg dela akumulatora.

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
	If lb = 0 then OE_{ACC0}	OE_{ACC0} – Upravlja three – state baferom na ulaz u donji bajt akumulatora	BACC0 – Three – state bafer za multipleksiranje neposrednog operanda na ulaz donjeg bajta akumulatora	
	If lb = 1 then OE_{ACC1}	OE_{ACC1} – Upravlja three – state baferom na ulaz u gornji bajt akumulatora	BACC1 – Three – sate bafer za multipleksiranje neposrednog operanda na ulaz gornjeg bajta akumulatora	
ACC←$(IR)_{7-0}$				Na željeno mesto u akumulatoru upisuje se neposredni operand definisan donjim bajtom registra instrukcije.
			Statusni bitovi: Z, N, P	



Upis neposrednog operanda u odgovarajuće bajtove akumulatora

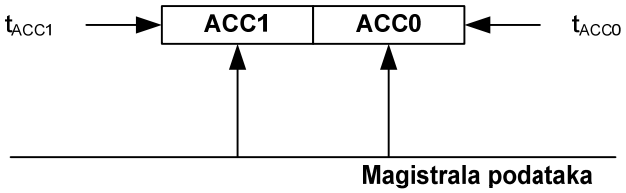
$$OE_{ACC0} = (\overline{IR}_{15}\overline{IR}_{14}\overline{IR}_{13}IR_{12})\overline{lb}$$

$$OE_{ACC0} = (\overline{IR}_{15}\overline{IR}_{14}\overline{IR}_{13}IR_{12})lb$$

Logičke funkcije signala OE_{ACC0} i OE_{ACC1}

LOAD – Punjenje akumulatora sadržajem memorijske lokacije

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
MAR←Adresa				U MAR se upisuje adresa memorijske lokacije čiji se sadržaj upisuje u akumulator[1].
D_{15}–D_0←(Memorija)				Pročitani sadržaje se postavlja na magistralu podataka
$Adresa_{RF}$←AL_{RF}				Selekcija akumulatora u RF (Register File). Adresa u RF (akumulator) se dobija na osnovu koda operacije LOAD i STORE.
ACC←D_{15}–D_0				Podaci magistrale se upisuju u akumulator
			Statusni bitovi: Z, N, P	



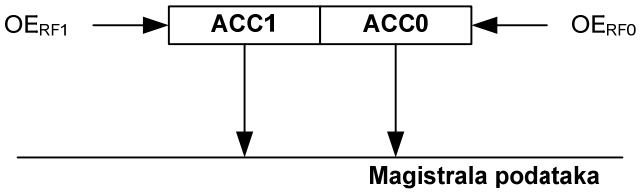
Upis podataka sa magistrale podataka u akumulator

STORE – Prenos sadržaja akumulatora u memoriju

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
MAR←Adresa				U MAR se upisuje adresa memorijske lokacije u koju se upisuje sadržaj akumulatora[2].
$Adresa_{RF}$←AL_{RF}				Selekcija akumulatora u RF (Register File). Adresa u RF (akumulator) se dobija na osnovu koda operacije LOAD i STORE.

[1] Način formiranja memorijske adrese zahteva postojanje posebnog mikroprograma.
[2] Način formiranja memorijske adrese zahteva postojanje posebnog mikroprograma.

D_{15}–D_0←(Akumulator)				Pročitani sadržaje se postavlja na magistralu podataka
Memorija←D_{15}–D_0				Podaci magistrale se upisuju u memoriju. Generiše se signal t_{MW}.
			Statusni bitovi: Z, N, P	



Čitanje podataka iz akumulatora na magistralu podataka

$$OE_{RF1} = OE_{RF0} = ARF_3\overline{A}\overline{R}\overline{F}_2\overline{A}R\overline{F}_1\overline{A}R\overline{F}_0$$

Formiranje signala dozvole izlaska podataka iz RF na magistralu podataka

ADD – Sabiranje

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu
$Adresa_{RFO}$←IR_7 – IR_4 (R2) **$Adresa_{RFC}$←IR_3 – IR_0 (R1)**				Selekcija registara operanada u RF (RF je dual – port memorija)
ALU_A←(R1) **ALU_B←(R2)**				Postavljanje sadržaja registara operanada na ulaz u ALU
ADD: (R1)+(R2)				Izvršavanje operacije sabiranja nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF
R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, N, C, O, P	

SUB – Oduzimanje

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu
$Adresa_{RFO}$←IR_7 – IR_4 (R2) **$Adresa_{RFC}$←IR_3 – IR_0 (R1)**				Selekcija registara operanada u RF (RF je dual – port memorija)
ALU_A←(R1) **ALU_B←(R2)**				Postavljanje sadržaja registara operanada na ulaz u ALU
SUB: (R1)-(R2)				Izvršavanje operacije oduzimanja nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF

R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, N, C, O, P	

OR – Logićko ILI

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu
$Adresa_{RFO}$←IR_7 – IR_4 (R2) **$Adresa_{RFC}$←IR_3 – IR_0 (R1)**				Selekcija registara operanada u RF (RF je dual – port memorija)
ALU_A←(R1) **ALU_B←(R2)**				Postavljanje sadržaja registara operanada na ulaz u ALU
OR: (R1) or (R2)				Izvršavanje operacije logičkog ILI nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF
R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, P	

AND logičko I

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu
$Adresa_{RFO}$←IR_7 – IR_4 (R2) **$Adresa_{RFC}$←IR_3 – IR_0 (R1)**				Selekcija registara operanada u RF (RF je dual – port memorija)
ALU_A←(R1) **ALU_B←(R2)**				Postavljanje sadržaja registara operanada na ulaz u ALU
AND: (R1) and (R2)				Izvršavanje operacije logičkog I nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF
R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, P	

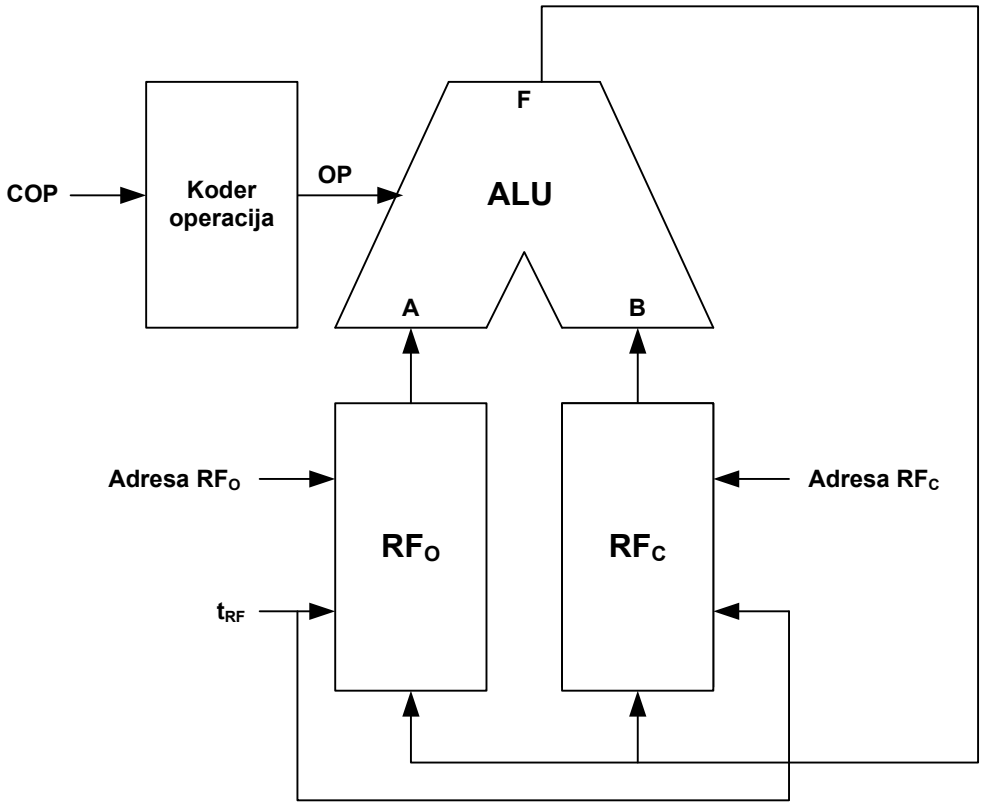
XOR – Ekskluzivno ILI

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu

$Adresa_{RFO}$←IR_7 – IR_4 (R2) **$Adresa_{RFC}$←IR_3 – IR_0 (R1)**				Selekcija registara operanada u RF (RF je dual – port memorija)
ALU_A←(R1) **ALU_B←(R2)**				Postavljanje sadržaja registara operanada na ulaz u ALU
XOR: (R1) xor (R2)				Izvršavanje operacije ekskluzivnog ILI nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF
R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, P	

NOT Logička negacija

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu
$Adresa_{RFO}$←IR_7 – IR_4 (R2)				Selekcija registara operanada u RF (RF je dual – port memorija)
ALU_A←(R1)				Postavljanje sadržaja registara operanada na ulaz u ALU
NOT: not(R1)				Izvršavanje operacije logičke negacije nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF
R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, N, C, O, P	



Izvršavanje aritmetičko – logičkih operacija

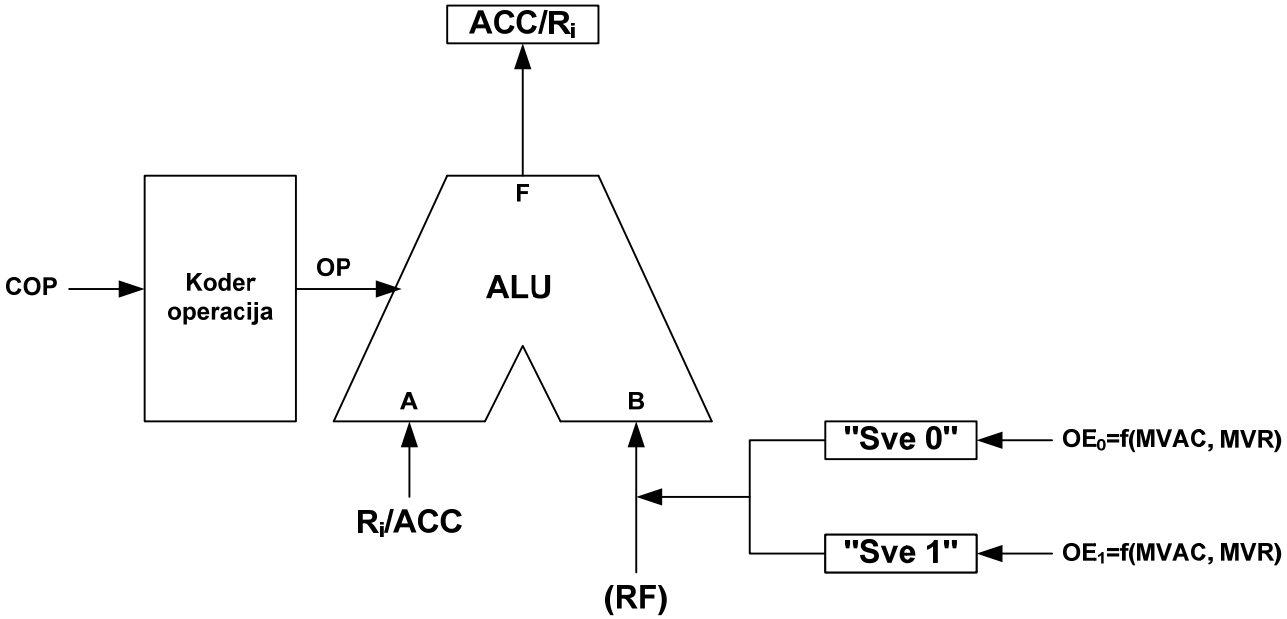
MVAC – Kopiranje sadržaja registra u akumulator
MVR – Kopiranje sadržaja akumulatora u registar

Radi unifikacije načina izvršavanja instrukcija ove dve instrukcije će se realizovati preko izvršavanja neke od aritmetičko – logičkih instrukcija pri čemu će se kao drugi operand koristiti (zavisno koja se instrukcija usvoji za realizaciju) podataka koji ima sadržaj "sve 0" odnosno "sve 1". Moguće varijante su:
MVAC:

1. ACC←(Ri)+0…00
2. ACC←(Ri)–0…00
3. ACC←(Ri) and 1…11
4. ACC←(Ri) or 0…00

Za ove potrebe u strukturu procesora potrebno je uvesti dva specijalna registra (koji nisu programski dostupni) – "Registar sve 0" i "Registar sve 1". Njihovi izlazi će se pojaviti na magistrali podataka u zavisnosti od potreba. U konkretnom slučaju to je određeno kodom operacije.

MAR←Adresa				U MAR se upisuje adresa instrukcije
PC←(PC)+1				Uvećava se sadržaj programskog brojača
IR←(Memorija)				Po adresi u MAR se čita instrukcija
ALU←Upravljački signali (operacija)				Generisanje upravljačkih signala za ALU jedinicu
$Adresa_{RFO}$←IR_7 – IR_4 (R2)				Selekcija registara iz koga se sadržaj kopira u akumulator u RF (RF je dual – port memorija)
ALU_A←(R1) **ALU_B←(R2)–"Registar sve nulu/jedinice"**				Postavljanje sadržaja registara operanada na ulaz u ALU
ADD: (R1)+(R2)				Izvršavanje operacije sabiranja nad (R1) i (R2)
$Adresa_{RFO,RFC}$←IR_{11} – IR_8 (R3)				Selekcija odredišnog registra (R3) u obe polovine RF – u konkretnom slučaju radi se o akumulatoru
R3←Rezultat				Pamćenje rezultata u odredišnom registru
			Statusni bitovi: Z, N, P	



Izvršavanje instrukcija MVAC i MVR