UNIVERZITET
METROPOLITAN
BEOGRAD

CS120 - ORGANIZACIJA RAČUNARA

Centralni procesor i magistrale

Lekcija 05

PRIRUČNIK ZA STUDENTE

CS120 - ORGANIZACIJA RAČUNARA

Lekcija 05

CENTRALNI PROCESOR I MAGISTRALE

UVOD

Program čini skup instrukcija koje su smeštene u memoriji.

U ovoj lekciji će biti pokazan najjednostavniji mogući procesor gde je moguće objasniti principe rada procesora iako se radi o veoma softificiranom delu računara (ako ne i najkomplikvaniji).

Najpre su opisani tipovi operacione jedinice i upravljačke jedinice.

Zatim, odrađen je primer adresiranja na uprošćenom mdelu CPU-a, radi lakšeg opisa principa rada.

U nastavku su opisani tipovi prenosa posataka CPU-a, kao i šta čini program u pogledu instrukcija.

Program čini skup instrukcija koje su smeštene u memoriji. CPU tipično čita (pribavlja) instrukcije iz memorije po jednu u datom trenutku, izvršava svaku instrukciju, a zatim pribavlja narednu. Proces se ponavlja beskonačno dugo.

Konačno obrađena je ogranizacija računarskog sistema sa aspekta magistrala.

Magistrala se koristi za povezivanje dva ili većeg broja sistemskih elemenata. Ona predstavlja skup linija (veza). Skup veza se deli od strane sistemskih elemenata, ali se takođe i koristi od strane sistemskih elemenata za međusobnu komunikaciju. Pojam magistrala obično ukazuje da su veze paralelne, pri čemu se duž istog puta prenosi po nekoliko signala.

Poglavlje 1

Integrisano kolo centralnog procesora

JEDNOSTAVNI MODEL CPU-A

Programsko brojilo (programski brojač) - registar PC - sadrži adresu sledećeg bajta koji će biti pribavljen u narednom ciklusu.

Jednostavni modelna kojem će biti opisan princip rada prikazan je na *Slici* . Model mikroprocesora ima *akumulator* A koji koji se upotrebljava za privremeno skladištenje jednog od operanada. Akumulator učestvuje pri izvršavanju aritmetičkih i logičkih opracija na podacima, te ima i središnju ulogu u prenosu podataka unutar mikroračunara ili sklopa mikroračunara.

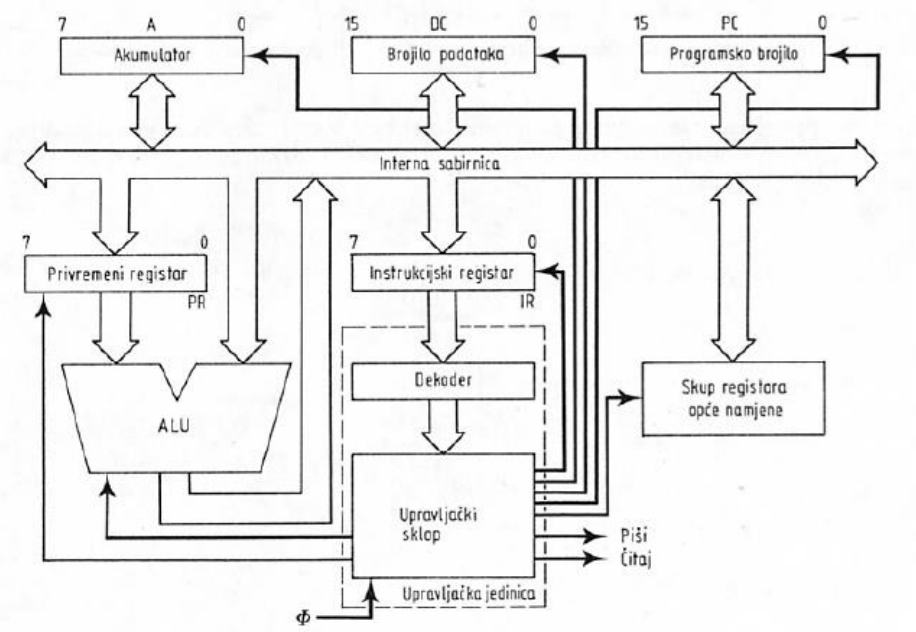
Programski brojač - registar PC - sadrži adresu sledećeg bajta koji će biti pribavljen u narednom ciklusu.

Operacijski kod instrukcije upisuje se u ***instrukcijski registar*** - registar IR. U 16-bitnom registru podataka, registru *DC*, sadržana je adresa memorijske lokacije u kojoj se nalazi operand.

Izvođenje svake instrukcije deli se na:

- fazu pribavljanja instrukcije -PRIBAVI (fetch),

- fazu izvršavanje instrukcije -IZVRŠI (execute).



Slika 1.1 Pojednostavljeni model mikroprocesora. [Izvor: Autor]

OPIS PRINCIPA RADA JEDNOSTANOG CPU-A

Memorijski sklop dekodira postavljenu adresu u cilju pristupa do odgovarajuće memorijske reči.

Mikroprocesor za vreme faze *PRIBAVI* postavlja sadržaj programskog brojila preko ***interne sabirnice*** na spoljnu *adresnu sabirnicu*. Ujedno šalje i odgovarajuće upravljačke signale na spoljnu upravljačku sabirnicu (u našem slučaju pojednostavljeni model imće samo dva upravljačka signala: *ČITAJ i PIŠI*).

Memorijski sklop dekodira postavljenu adresu (prisutnu na adresnoj sabirnici) u cilju pristupa do odgovarajuće memorijske reči. Za nekoliko stotina *ns* (npr. 500 *ns*) sadržaj specificirane memorijske lokacije pojaviće se na spoljnoj sabirnici podataka. Taj se sadržaj skladišti u instrukcijskom registru IR, i to je operacijski kod instrukcije. Za vreme faze PRIBAVI mikroprocesor upotrebljava svoju internu logiku i povećava sadržaj programskog brojila.

U fazi *IZVRŠI* upravljačka jedinica, u skladu s operacijskim kodom koji je skladišten u instrukcijskom registru, stvara niz upravljačkih signala. Rezultat tog niza signala su odgovarajući prenosi podataka, odnosno operacije (npr. aktiviranje pojedinih sklopova unutar aritmetičko-logičke jedinice, izvršavanje (izvođenja) zadate instrukcije).

GENERATOR TAKTA

Operacije unutar mikroprocesora (često se nazivaju mikrooperacijama) sinhronizovane su generatorom takta.

Operacije unutar mikroprocesora (često se nazivaju mikrooperacijama) sinhronizovane su generatorom takta. Perioda generatora takta može biti, u zavisnosti od tipa mikroprocesora, od 100 *ns* do nekoliko μs.

Signali generatora takta mogu se sastojati od jednog ili više signala (to je onda višefazni generator takta, npr. mikroprocesor M6800 ima signale φ1 i φ2). Za pojednostavnjeni model izabran je jednofazni generator takta (*Sl.6*). φ je obično oznaka za signal generatora takta.

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

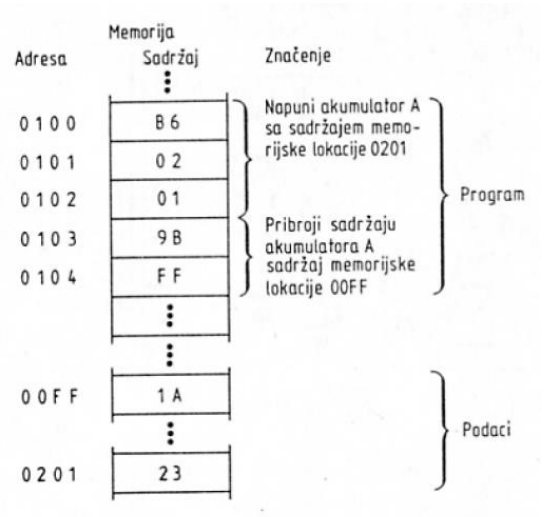
Poglavlje 2

Opis principa rada CPUa

POČETNO STANJE PRE IZVRŠENJA NAREDBI

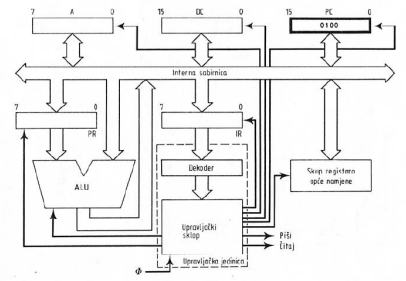
Početni uslov (sadržaj registara), su samo oni registri koji učestvuju u izvođenju programa.

Slika 1 opisuje početno stanje operativne memorije, sa adresama i sadržajem.



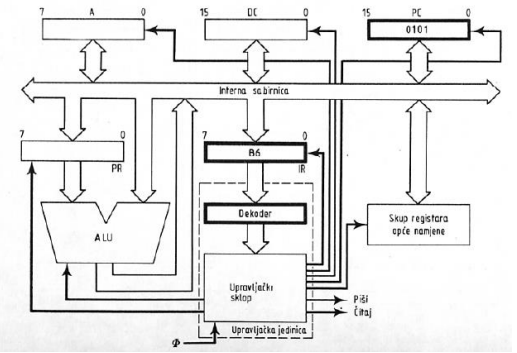
Slika 2.1 Adrese i sadržaji memorijski lokacija na prikazani u heksadekadnom sistemu. [Izvor: Autor]

Početni uslov (sadržaj registara) prikazuje *Slika 2* . Na slici su označeni samo oni registri koji učestvuju u izvođenju programa. U programskom brojači postavljena je adresa prve instrukcije.



Slika 2.2 Početni uslovi. [Izvor: Autor]

Slika 3 prikazuje stanje nakon faze *FETCH* u instrukcijskom registru skladišten je operacijski kod instrucije, sadržaj programskog brojača povećan je za 1.



Slika 2.3 Stanje nakon faze FETCH: Prvi ciklus prve instrukcije. [Izvor: Autor]

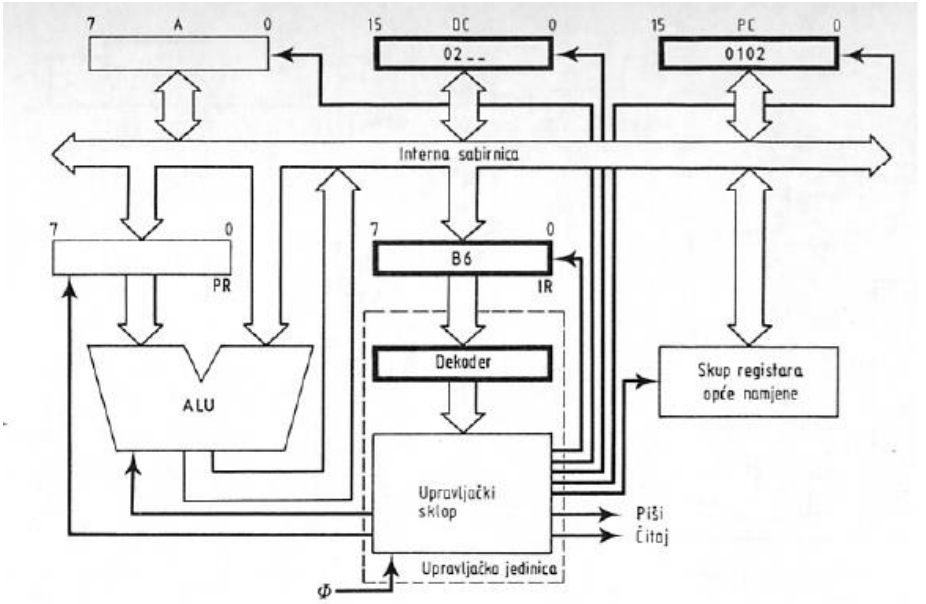
SADRŽAJ REGISTARA

Sadržaj instrukcijskog registra 10110110 (B6-heksadekadno) dekodiran je kao: napuni akumulator A sadržajem memorijske lokacije čija je adresa sadržana u sledeća dva bajta.

Sadržaj instrukcijskog registra ***10110110*** (***B6***-heksadekadno) dekodiran je kao: *napuni akumulator A sadržajem memorijske lokacije čija je adresa sadržana u sledeća dva bajta*. Mikroprocesor pribavlja sledeći bajt postavljanjem sadržaja programskog brojila na adresnu sabirnicu i generiranjem upravljačkog signala *READ*. Pribavljeni bajt se smeštava u brojilo podataka.

Slika 4 prikazuje stanje nakon pribavljanja značajnijeg bajta adrese operanda (**02**..); programsko je brojilo uvećano za 1. Mikroprocesor pribavlja treći bajt instrukcijske reči-manje značajni bajt adrese operanda (..**01**) i smešta ga u brojač podataka.

Programski brojač uvećava se za 1 i pokazuje na sledeću instrukciju 9B.

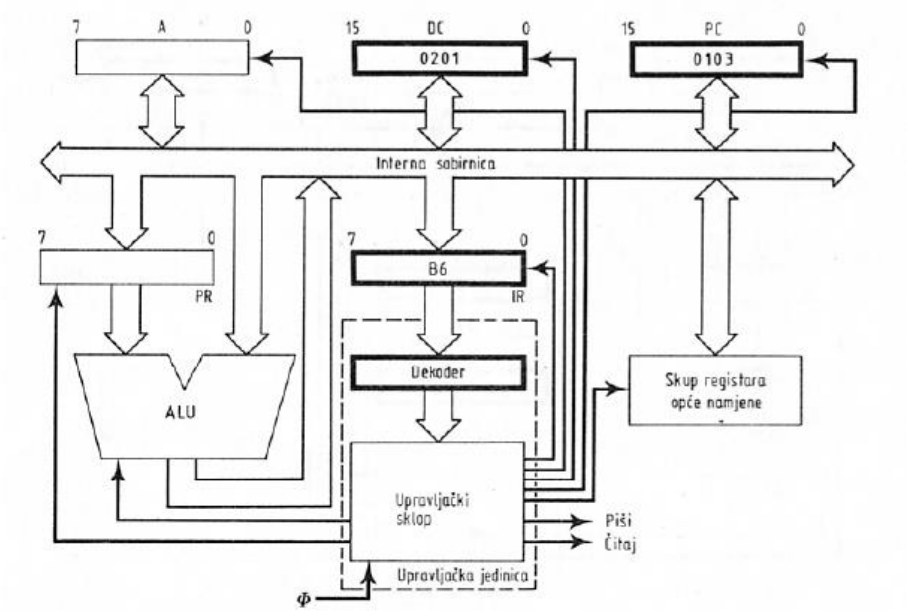


Slika 2.4 Stanje nakon pribavljanja značajnijeg bajta adrese operanda: Drugi ciklus prve instrukcije. [Izvor: Autor]

INSTRUKCIJSKA REČ

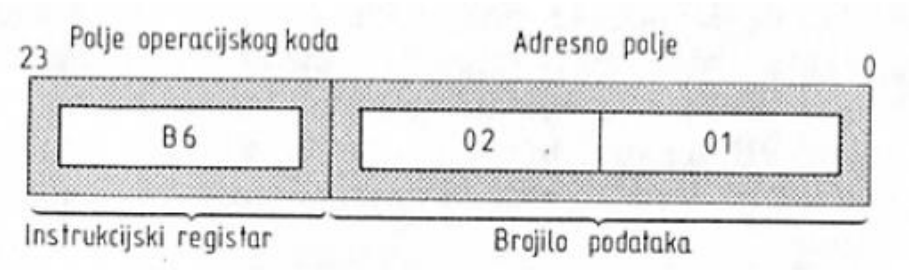
Instrukcijska reč može biti različita od procesorke reči.

Slika 5 prikazuje stanje nakon pribavljanja manje značajnog bajta adrese operanda.



Slika 2.5 Stanje nakon pribavljanja manje značajnog bajta adrese operanda: Treći ciklus prve instrukcije. [Izvor: Autor]

Ako se na trenutak pretpostavi da instrukcijski registar IR (0-7) i brojilo podataka DC (0-15) čine jedan 24-bitni registar, može se instrukcijski registar smatrati poljem operacijskog koda, a brojilo podataka adresnim poljem (Slika 6).



Slika 2.6 Instrukcijska reč sastavljena iz tri bajta. [Izvor: Autor]

SADRŽAJ REGISTARA NAKON PRIBAVLJANJA KOMPLETNE INSTRUKCIJE

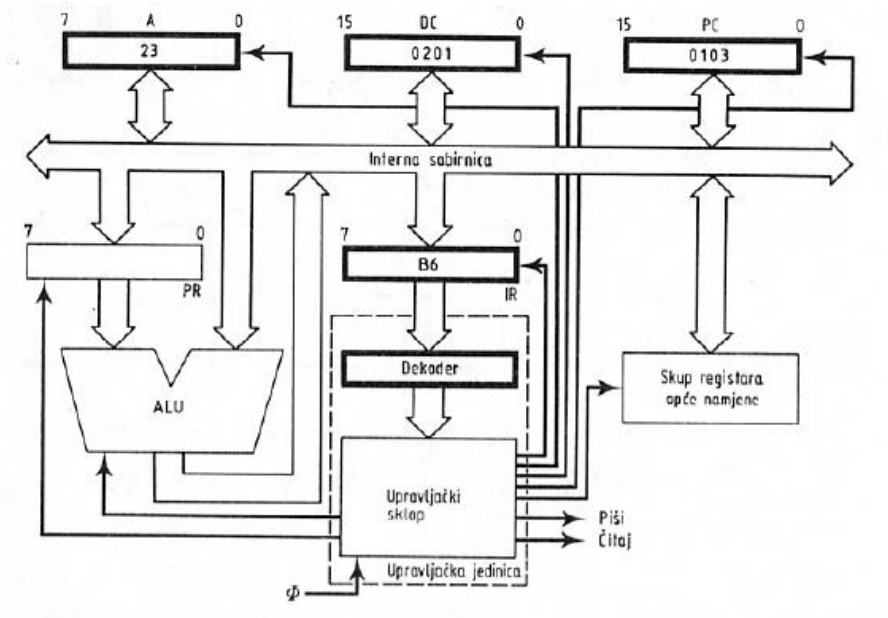
Postupku pribavljanja kompletne instrukcijske reči mikroprocesora su bila tri ciklusa (3x1 bajt), dok bi računar sa dužinom reči od 24 bita taj isti postupak obavio u jednom ciklusu

Postupku pribavljanja kompletne instrukcijske reči mikroprocesora su bila tri ciklusa (3x1 bajt), dok bi računar sa dužinom reči od 24 bita taj isti postupak obavio u jednom ciklusu.

Mikroprocesor pribavlja operand postavljanjem sadržaja brojila podataka na adresnu sabirnicu i generisanjem upravljačkog signala *READ*.

Slika 7 prikazuje konačni rezultat izvođenja prve instrukcije: akumulator A napunjen je sadržajem memorijske lokacije 0201.

Sadržaj programskog brojača ***nije povećan***, jer je pribavljen operand a ne instrukcija - mikroprocesor je bio u fazi *EXECUTE*.

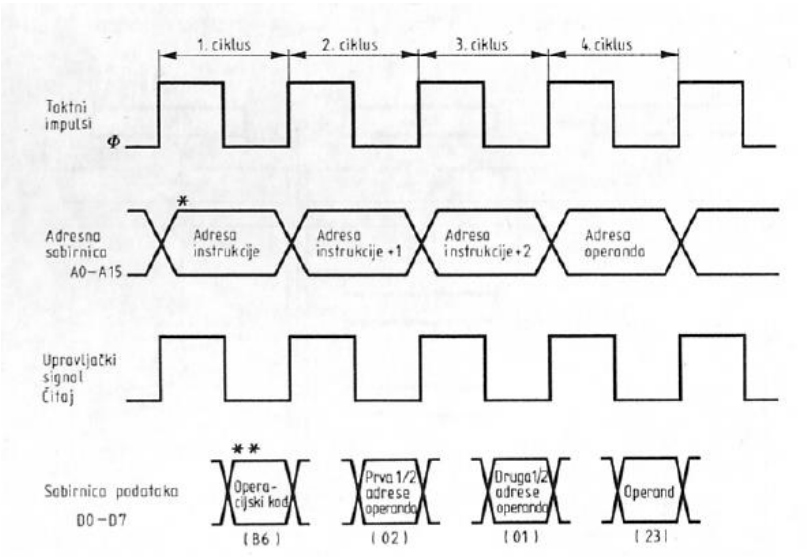


Slika 2.7 Stanje nakon izvođenja prve instrukcije: Četvrti ciklus prve instrukcije. [Izvor: Autor]

VREMENSKI DIJAGRAM IZVOĐENJE PRVE INSTRUKCIJE

Instrukcija se izvodi u četiri ciklusa (periode) generatora takta

Slika 8 prikazuje pojednostavnjeni vremenski dijagram izvođenja prve instrukcije. Instrukcija se izvodi u četiri ciklusa (periode) generatora takta φ.



Slika 2.8 Pojednostavljeni vremenski dijagram izvođenja prve instrukcije. [Izvor: Autor]

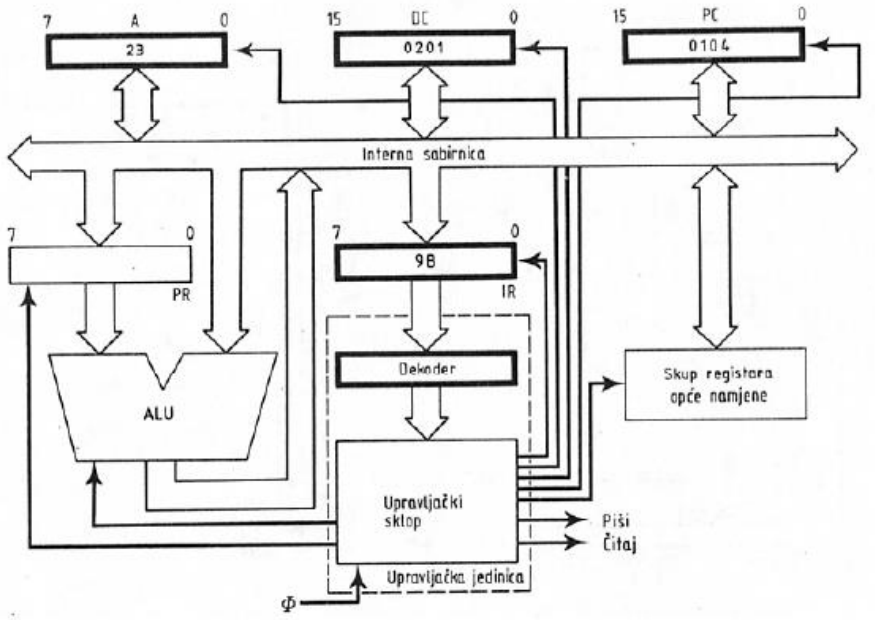
TOK DRUGE INSTRUKCIJE

U instrukcijskom registru smešten je operacijski kod druge instrukcije.

Nastavlja se dalje izvođenje programa. *Slika 9 (sl.* strana) prikazuje stanje nakon pribavljanja prvog bajta druge insrukcijske reči. U instrukcijskom registru smešten je operacijski kod druge instrukcije.

Sadržaj programskog brojača povećan je za 1.

Instrukcijski kod 9B, uz specificiranje operacije (prihvatanja operanda sadržaju akumulatora A), određuje i način adresiranja - sledeći bajt je manje značajan bajt adrese operanda.

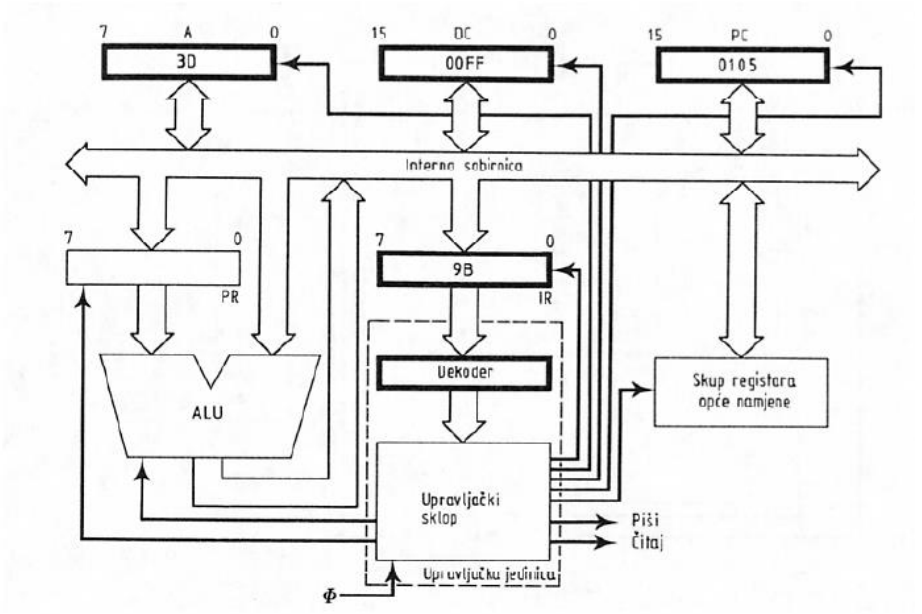


Slika 2.9 Stanje nakon pribavljanja prvog bajta druge instrukcije: Prvi ciklus druge instrukcije. [Izvor: Autor]

STANJE NAKON DRUGE INSTRUKCIJE

Programski brojač uvek se povećava za 1

Slika 10 (sl. strana) prikazuje stanje nakon pribavljanja drugog bajta instrukcijske reči. Programski brojač povećan je za 1, a u brojač podataka smeštava se adresa operanda.



Slika 2.10 Prikaz konačnog izvođenja instrukcije. [Izvor: Autor.]

KONAČNO STANJE POSLE OBE INSTRUKCIJE

Vidimo da se razmatranja na jednostavnom modelu u ovom primeru podudaraju s izvođenjem u stvarnim mikroprocesorima.

slika 12 prikazuje vremenski dijagram izvođenja druge instrukcije.

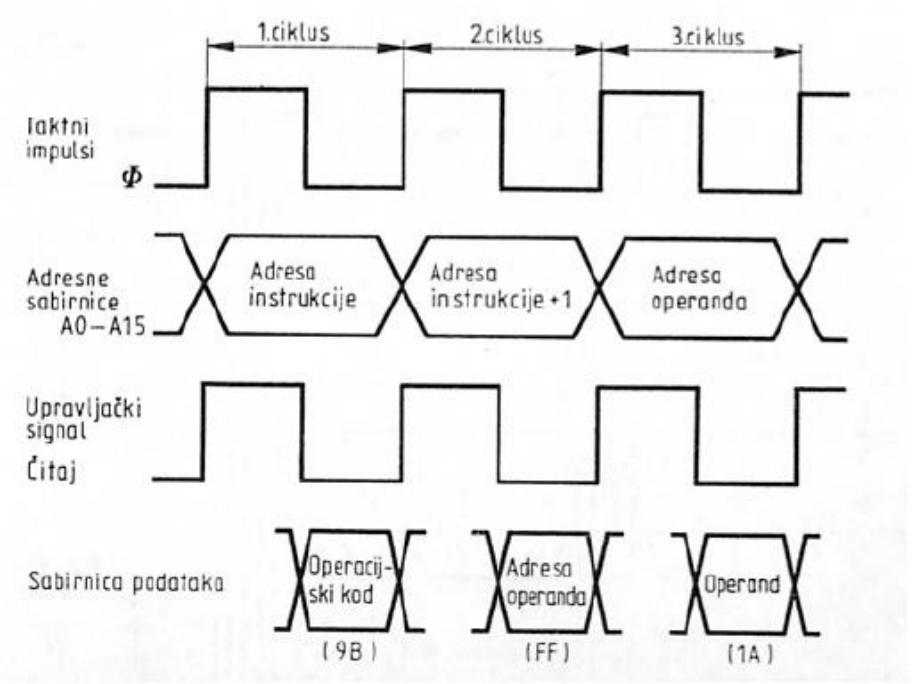
Tabela na slici 11 prikazuje stanje na sabirnici podataka i adresnoj sabirnici za model mikroprocesora prilikom izvođenja zadatog programa sa *Slike 1*. U mnemoničkom kodu program bi imao (za mikroprocesor M6800) sledeći oblik:

```
LDAA $0201
ADDA $FF
```

Stanje na sabirnici podataka i adresnoj sabirnici

Instrukcija	Broj ciklusa	Ciklus	Signal (Čitaj)	Adresne sabirnice	Sabirnice podataka
B6	4	1	1	Adresa instrukcije	Operacijski kod
		2	1	Adresa instrukcije +1	Adresa značajnijeg bajta operanda
		3	1	Adresa instrukcije +2	Adresa manje značajnog bajta operanda
		4	1	Adresa operanda	Operand
9B	3	1	1	Adresa instrukcije	Operacijski kod
		2	1	Adresa instrukcije +1	Adresa operanda
		3	1	Adresa operanda	Operand

Slika 2.11 Stanje na sabirnici podataka i adresnoj sabirnici. [Izvor: Autor]



Slika 2.12 Vremenski dijagram izvođenja druge instrukcije. [Izvor: Autor]

Broj ciklusa potreban mikroprocesoru M6800 za izvođenje prve instrukcije je 4, a druge 3 ciklusa.

Poglavlje 3

Magistrale računarskog sistema

INTERNO POVEZIVANJE CPU-A

Broj i funkcija internih registra R0 do Rn-1 menja se od mašine do mašine.

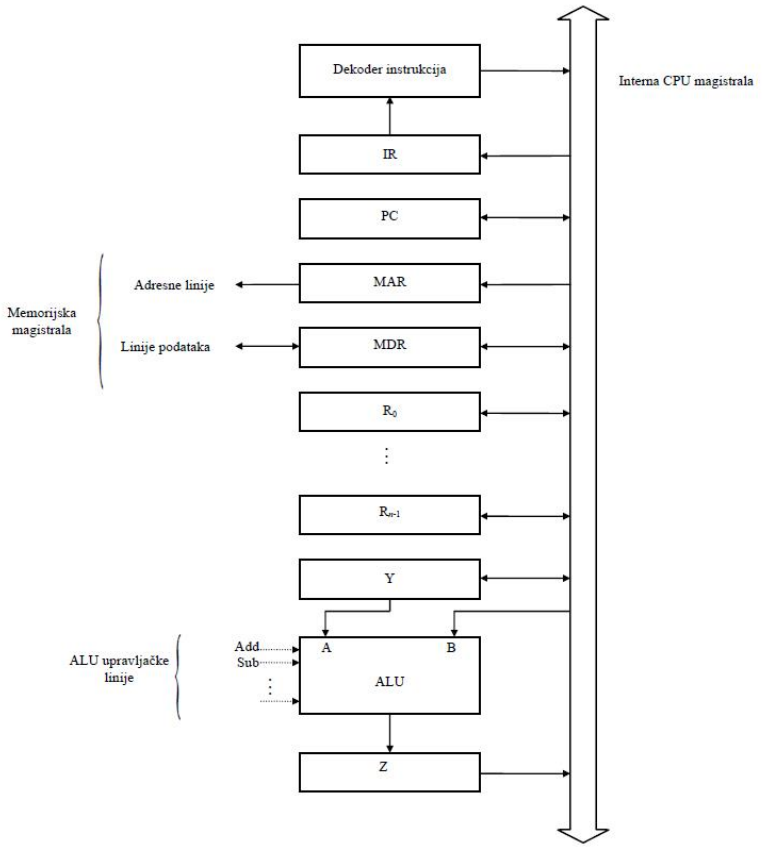
Interno povezivanje blokova CPU-a

Blokovi CPU-a prikazani na slici 1 mogu se organizovati i međusobno povezati na različite načine. Jedna takva organizacija je prikazana na slici6 . U ovom slučaju ALU i svi interni registri su povezani preko jedinstvene zajedničke magistrale. Magistrala je naravno interna i ne sme se poistovetiti sa spoljnom magistralompreko koje se CPU povezuje sa memorijom i U/I uređajima.

Broj i funkcija internih registra R_0 do R_{n-1} menja se od mašine do mašine. U najvećem broju slučajeva koriste se od strane programera kao registri opšte namene, a neki od njih mogu biti određeni za specijalnu namenu kao što su indeksni registri ili pokazivači magacina.

Registri Y i Z su obično transparentni programeru. Drugim rečima programer ne treba da vodi računa o njihovom postojanju, pošto se njima nikad direktno ne obraća bilo kojom instrukcijom. Oni se unutar CPU-a koriste za interno smeštanje (čuvanje) podataka u toku izvršenja neke instrukcije.

Registri Y i Z se nikad direktno ne koriste tako da jedna instrukcija smešta podatke a druga instrukcija ih kasnije koristi.



Slika 3.1 Interni putevi podataka CPU-a organizovanog oko jedinstvene magistrale. [Izvor: Autor]

TIPOVI PRENOSA PODATAKA KOD CPU-A

Sa tačke gledišta prenosa podataka, tj. mesta odakle se pribavljaju izvorni operandi i mesta gde se smeštaju odredišni operandi, instrukcije se mogu podeliti u tri kategorije

Tipovi prenosa CPU-a

Sa tačke gledišta prenosa podataka, tj. mesta odakle se pribavljaju izvorni operandi i mesta gde se smeštaju odredišni operandi, instrukcije se mogu podeliti u sledeće tri kategorije:

a) **registar-memorija** - ostvaruje se prenos registar↔memorija.

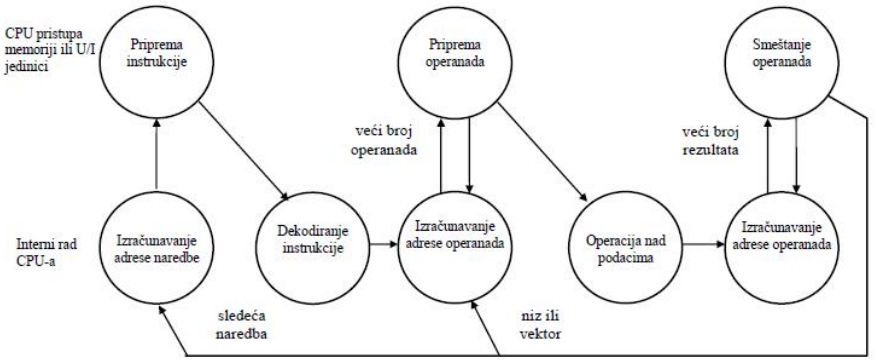
b) **registar-registar** - ostvaruje se prenos registar↔registar.

c) **memorija-memorija** – najpre se ostvaruje prenos memorija→registar za privremeno čuvanje podataka; opciono se obavlja ALU operacija i rezultat smešta u registar za privremeno čuvanje podataka, a u trećem koraku se rezultat smešta u memoriju.

Dijagram stanja ciklusa instrukcija

U toku ciklusa izvršenja pojedinih instrukcija može da se javi potreba za većim brojem obraćanja memoriji. Takođe, umesto obraćanja memoriji, instrukcija može da specificira U/

I aktivnost. Imajući u vidu ova dodatna razmatranja, na slici je prikazan detaljniji pogled na ciklus instrukcije.



Slika 3.2 Dijagram stanja ciklusa instrukcije. [Izvor: Autor]

TIPIČNI DIJAGRAM TOKA AKTIVNOSTI CPU-A U TOKU IZVRŠENJA JEDNE NAREDBE

Izvršenje instrukcije se grubo može podeliti na dve faze: fazu pripreme naredbe i fazu izvršenja naredbe. Prva faza je identična za sve instrukcije, dok faza izvršenja zavisi od tipa instrukcije.

Slika 3 je data u obliku dijagrama stanja. Za bilo koji ciklus instrukcije neka od stanja ne moraju postojati, a druga se mogu izvršavati veći broj puta. Stanja imaju sledeće značenje:

• **Izračunavanje adrese instrukcije (*iac*)** - određuje se adresa instrukcije koja treba da se izvrši kao naredna. Obično se svodi na dodavanje jedinice adresi prethodne instrukcije.

• **Priprema (pribavljanje) instrukcije (*if*)** - instrukcija iz specificirane memorijske lokacije se čita i smešta u CPU.

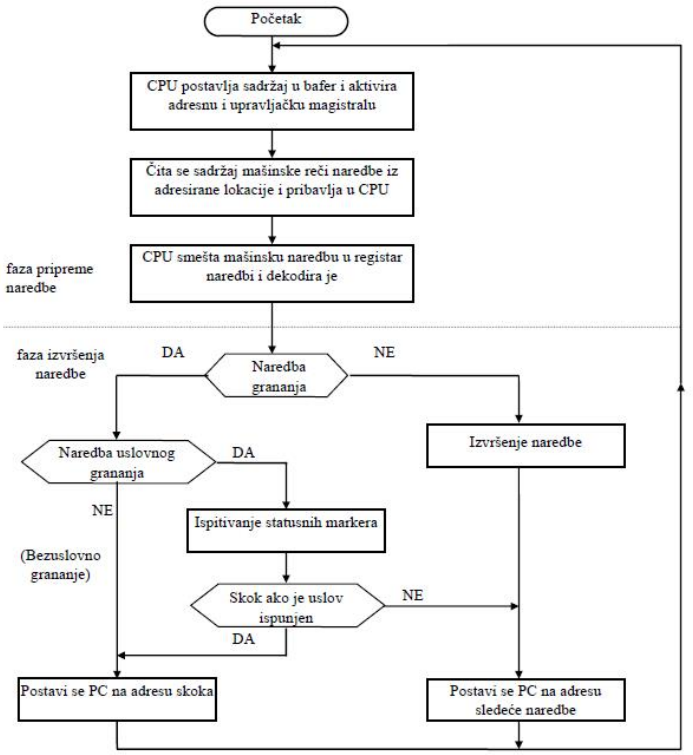
• **Dekodiranje instrukcije (*id*)** - vrši se analiza instrukcije sa ciljem da se odredi tip operacije koja će se izvesti nad operandima koji se koriste od strane te instrukcije.

• **Izračunavanje adrese operanada (*aoc*)** - ako se operacijom specificira obraćanje operandu koji je smešten u memoriji ili je dostupan preko U/I, tada se određuje adresa operanda.

• **Priprema (pribavljanje) operanada (*of*)** - pribavlja se operand iz memorije ili se čita iz U/I uređaja.

• **Operacija nad podatkom (*do*)** - obavlja se specificirana operacija.

• **Smeštanje operanada (*os*)** - upisuje se operand u memoriju ili u U/I uređaj.



Slika 3.3 Tipični dijagram toka aktivnosti CPU-a u toku izvršenja jedne naredbe. [Izvor: Autor]

Poglavlje 4

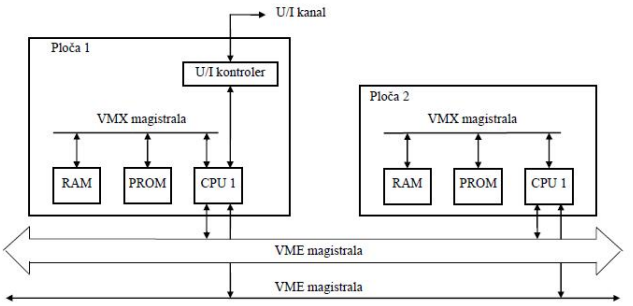
Organizacija računarskog sistema sa aspekta magistrala

SISTEM MAGISTRALA

Magistrala se koristi za povezivanje dva ili većeg broja sistemskih elemenata.

Magistrala se koristi za povezivanje dva ili većeg broja sistemskih elemenata. Ona predstavlja skup linija (veza). Skup veza se deli od strane sistemskih elemenata, ali se takođe i koristi od strane sistemskih elemenata za međusobnu komunikaciju. Pojam magistrala obično ukazuje da su veze paralelne, pri čemu se duž istog puta prenosi po nekoliko signala.

Globalno posmatrano, ***sistem magistrala*** predstavlja skup većeg broja magistrala koje se koriste za povezivanje različitih sistemskih elemenata elemenata u okviru računarskog sistema. Kao što je prikazano na slici 1 , sistem magistrala čine magistrale koje su hijerarhijski organizovane na sledećim, različitim nivoima:



Slika 4.1 Sistem magistrala. [Izvor: Autor]

Nivo ploča - na najnižem nivou je komponentno-orijentisana magistrala definisana od strane CPU-a i perifernih čipova. Obično se ova magistrala zove *lokalna magistrala*.

• ***Zadnja ploča*** - obezbeđuje komunikaciju između elemenata sistema, na nivou ploča. Zadnja ploča je obično realizovana kao štampana ploča sa većim brojem identičnih konektora koji su paralelno povezani. Svaka ploča se postavlja u svoj konektor i na taj način povezuje sa ostalim pločama. Zadnja ploča se često zove i ***majka ploča*** (motherboard). Definicija signala na zadnjoj ploči obično se ne vezuje za određeni tip CPU-a kao što je to slučaj sa lokalnom magistralom. Na tržištu danas postoji veći broj standardizovanih magistrala zadnje ploče, od kojih su poznatije VME, Multibus I i II, Nubus, Fastbus, Futurebus i dr.

• ***Nivo interfejsa*** - ovim tipovima magistrala ostvaruje se komunikacioni put između U/I uređaja (diskovi, štampači, i dr.) i ostatka sistema. Ove magistrale obično povezuju nezavisne

sisteme i predviđene su za rad na većim rastojanjima u odnosu na nivo zadnje ploče. Tipični predstavnici su *SCCI* (Small Computer System Interconnect) magistrala, *GPIB* (General Purpose Interface Bus) poznata kao IEEE 488, i dr.

Sa ciljem da se poboljšaju performanse sistema u sistem se ugrađuju dodatne magistrale.

PODELA MAGISTRALA U ZAVISNOSTI OD NAMENE

Magistrale se mogu podeliti na namenske i nenamenske.

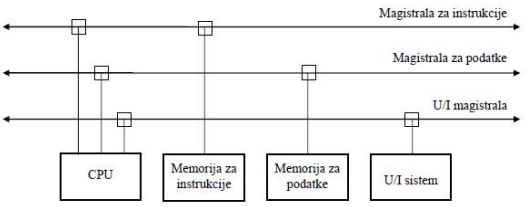
Najveći broj magistrala se može podeliti na sledeće tri sekcije (magistrale):

• adresna magistrala - specificira jednu memorijsku lokaciju ili U/I port koji komunicira sa CPU-om,

• magistrala za podatke - koristi se za prenos podataka,
• upravljačka magistrala - upravljački signali pomoću kojih se sinhronizuje prenos podataka. Kod ove magistrale postoje posebne linije pomoću kojih se reguliše prioritet prekida i dodela deljivih resursa na korišćenje, kao što je magistrala.

Podela magistrala u zavisnosti od namene

Magistrale se mogu podeliti na *namenske* i *nenamenske*. Namenske magistrale se (slika 2), kao što i samo ime ukazuje, koriste za obavljanje specifičnih funkcija. Osnovna njihova prednost je izrazito visoka propusnost, a nedostatak je njihov veliki broj. Kada je u sistemu instaliran veći broj namenskih magistrala, prenos se može istovremeno ostvariti po svim magistralama. Zbog zahteva za ugradnju velikog broja konektora namenske magistrale se ne koriste kod realizacije manjih računarskih sistema. Nenamenske magistrale, zovu se takođe i deljive magistrale, se koriste da obave veći broj funkcija.



Slika 4.2 Namenske magistrale. [Izvor: Autor]

KLASIFIKACIJA DELJIVIH MAGISTRALA

Deljive magistrale možemo svrstati u zavisnosti od toga kako je izvršena podela funkcija sistema po pločama. Kriterijum podele može biti tip resursa ili funkcija koja se obavlja.

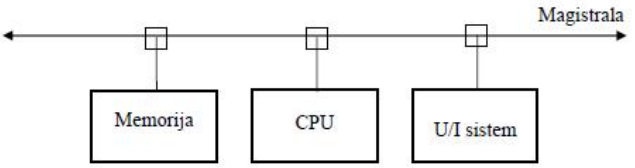
Na slici 3 CPU je povezan sa većim brojem jedinica preko jedinstvene deljive magistrale. Kod deljivih magistrala kapacitet brzine prenosa je manji, ali postoji i potreba za ugradnjom arbitražnog mehanizma pomoću koga se ostvaruje zaštita od konflikata u slučaju kada dva, ili veći broj potencijalnih korisnika iztovremeno zahteva dodelu magistrale.

Klasifikacija deljivih magistrala

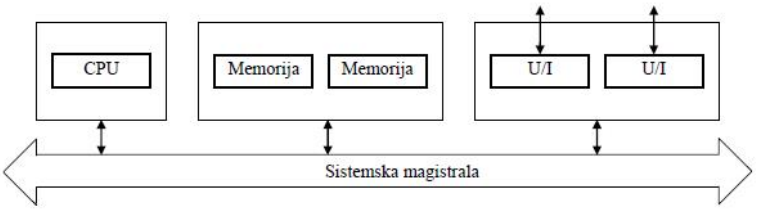
Deljive magistrale možemo svrstati u zavisnosti od toga kako je izvršena podela funkcija sistema po pločama. Kriterijum podele može biti tip resursa ili funkcija koja se obavlja.

Podela magistrala u zavisnosti od tipa resursa

Resursi istog tipa, recimo memorije, pakuju se u jedinstvenu celinu i povezuju na magistralu. Kompletan mali računarski sistem se formira povezivanjem svih tipova resursa, CPU-memorija-U/I magistrala, kao što je prikazano na slici 4 .



Slika 4.3 Jedinstvena deljiva magistrala. [Izvor: Autor]



Slika 4.4 CPU-memorija-U/I magistrala. [Izvor: Autor]

Podela magistrala u zavisnosti od funkcije

U ovom slučaju vrši se povezivanje ploča koje obavljaju polunezavisne funkcije. Za svaku ploču se pretpostavlja da ima dovoljnu lokalnu procesnu moć, memoriju i U/I, tj. da predstavlja jedinstveni računar na ploči.

TERMINOLOGIJA VEZANA ZA MAGISTRALE

Neki od uređaja mogu biti i gospodar i sluga, ali ne istovremeno.

Ploče koje se povezuju na magistralu mogu biti tipa:
a) *gospodar* (en. master) - može inicirati ciklus na magistrali
b) *sluga* (en. slave) - odaziva se gospodaru.

Neki od uređaja mogu biti i gospodar i sluga, ali ne istovremeno. Svi uređaji koji rade kao gospodari magistrale zovu se potencijalni gospodari magistrale. S obzirom da je u jednom trenutku magistrala dodeljena samo jednom gospodaru, koga zovemo tekući gospodar magistrale, potrebno je ugraditi arbitražni mehanizam kojim se odlučuje koji će gospodar magistrale u narednom trenutku dobiti pravo upravljanja nad magistralom. Kada prenos podataka počne, ploča koja predaje podatak se zove izvor, a ploča koja prima podatak odredište.

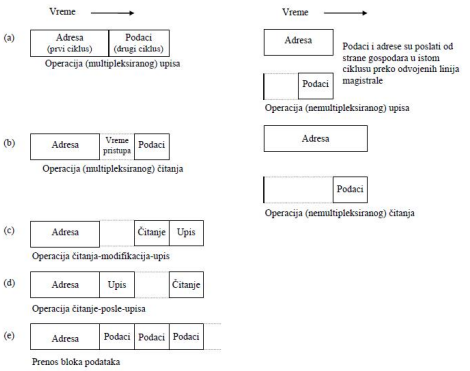
Kompletna sekvenca, od trenutka kada se magistrala zahteva do trenutka završetka prenosa podataka zove se transakcija na magistrali. Da bi se obavio prenos po magistrali, neophodno je da se obave sledeće operacije:

- ***zahtev***
- ***arbitraža***
- ***adresiranje***
- ***prenos podatka***
- ***detekcija***
- ***signalizacija greške.***

Prenos podataka

Kao što je prikazano na slici 5 , moguće je ostvariti sledeće tipove prenosa podataka:

• *Operacija upis* (slika 5a) - gospodar magistrale predaje adresu u prvom ciklusu, a zatim podatak u narednom ciklusu. Sluga mora da prepozna adresu pre nego što prihvati podatak. Na levoj strani slike 6a prikazan je vremenski multipleksirani rad, a na desnoj strani nemultipleksirani rad operacije upis. Kod multipleksiranog rada iste linije se u prvom ciklusu koriste za prenos adresa, a u drugom za prenos podataka. Kod nemultipleksiranog rada za prenos podataka i adresa se koriste posebne linije.



Slika 4.5 Tipovi prenosa podataka i odnosi na magistrali. [Izvor: Autor]

SPECIFIKACIJA ADRESE SLUGE

Adresa sluge specificira se adresnim linijama na magistrali.

• Operacija čitanja (slika 5b) - gospodar predaje adresu i čeka da podatak bude dostupan od strane sluge. Vreme čekanja se često zove vreme pristupa.

• Operacija čitanje-modifikacija-upis (slika 5c) - čine je tri ciklusa: adresni, čitanje podataka i upis podataka. Celokupna operacija je nedeljiva, sa ciljem da se u toku modifikacije elemenata podataka zabrani pristup tom podatku od strane drugog gospodara magistrale.

• Operacija čitanje-posle-upisa (slika 5d) - čini je nedeljiva sekvenca upis-čitanje. Operacija čitanja se obavlja radi provere.

• Operacija prenos bloka podataka (slika 5e) - nakon jednog adresnog ciklusa sledi "*n*" ciklusa podataka. Prvi podatak se prenosi na specificiranu adresu, a naredni na sukcesivne adrese.

• Razbijeni prenos podataka - u toku operacije čitanja vreme pristupa informaciji može biti dugo. Umesto da se magistrala zauzme celo vreme, ona se oslobađa kako bi se obavile druge transakcije. Kasnije se prenos završava. Ova tehnika se često zove paketna komutacija (*message* ili *packet switching*).

Adresiranje - Nakon što je gospodar stekao pravo upravljanja nad magistralom on treba da uspostavi kontakt sa jednim ili

većim brojem sluga sa ciljem da se ostvari prenos podataka. Ovo se zove *adresiranje*, a sam postupak čine dva koraka: adresiranje ploče i adresiranje elemenata na toj ploči.

Obično se isti mehanizam adresiranja koristi u oba slučaja. Svakoj ploči se dodeljuje blok adresa, tj. MS adresnim bitovima specificira se ploča, a LS adresnim bitovima iste sdrese specificira se elementa podataka na ploči.

U daljem tekstu ukažimo na način specifikacije adrese sluge.

Specifikacija adrese sluge

Adresa sluge specificira se adresnim linijama na magistrali. Adresa sluge obično odgovara adresi ploče, i ova je jedinstvena. U globalu razlikujemo dve metode za dodelu adrese pločama:

• **Logičko adresiranje** - zove se takođe i lokaciono-nezavisno. Svakoj ploči se dodeljuje jedinstvena adresa ili grupa adresa. Dodela adresa se ostvaruje ručnim postavljanjem prekidača ili kratkospojnika na samoj ploči. Adrese se postavljaju proizvoljno i ne zavise od tipa ploče i njene pozicije na zadnjoj ploči. Da bi se ostvarilo logičko adresiranje na nivou svake ploče, treba ugraditi dodatni hardver. Standardne magistrale kao što su VME i Unibus koriste ovaj način adresiranja.

BROJ SLUGA KOJI UČESTVUJU U TRANSAKCIJI NA MAGISTRALI

Često se u toku rada javlja potreba da jedan gospodar komunicira sa većim brojem sluga. U tom cilju koriste se broadcall i broadcast operacije.

Geografsko adresiranje - zove se takođe i lokaciono-zavisno. Svaka ploča se adresira na osnovu njene fizičke lokacije, tj. mesta gde je ploča postavljena na zadnjoj ploči. Ova lokacija se zove *broj slota* i do nje se direktno dovode signali za selekciju te ploče.

Broj sluga koji učestvuju u transakciji na magistrali

Često se u toku rada javlja potreba da jedan gospodar komunicira sa većim brojem sluga. U tom cilju koriste se *broadcall* i *broadcast* operacije.

Broadcall operacija (selektivna emisija) uslovljava da sve selektovane ploče tipa sluga postave svoje podatke na magistralu. Nakon toga vrši se zbirna AND ili OR operacija podataka selektovanih sluga. Ova operacija ima ograničeno delovanje, a tipično se koristi za identifikaciju izvora prekida. Na primer, svakom potencijalnom izvoru prekida dodeljuje se

jedna bit pozicija na magistrali. Ovom bit pozicijom na jedinstveni način se identifikuje sluga. Ako je magistrala podataka 32-bitna, moguće je identifikovati do 32 izvora prekida.

Broadcast operacija (emisija svima) se uglavnom koristi za održavanja konzistentnosti podataka. Njeno korišćenje ima puni smisao kod multiprocesorskih sistema kada je na nivou svakog

procesora instalirana keš memorija, tj. postoji distribucija keša a shodno tome i potreba za simultano ažuriranje. Reset operacija, slučaj kada se svaka ploča postavlja u svoje inicijalno stanje, je takođe oblik broadcast operacije. ***Broadcast*** i ***Broadcall*** operacije obično se iniciraju preko specijalnih adresa (adresa 0 ili FF), ili aktiviranjem specijalne upravljačke linije pri čemu se adresne linije koriste za selekciju podskupa, ili svih sluga.

Prenos bloka podataka - Elementi bloka podataka se obično smeštaju u uzastopnim memorijskim lokacijama i, kao što je prikazano na slici 6 , oni se mogu preneti na dva načina. U prvom slučaju (slika 6a), kažemo da se radi o prenosu na nivou jednog ciklusa, tj. posle svake generisane adrese vrši se prenos po jednog podatka. U drugom slučaju (slika 6b), podaci se prenose u paketima (en. burst). U ovom slučaju magistrala je osposobljena za ovakav tip prenosa. Na početku prenosa predaje se inicijalna adresa nakon čega sledi blok podataka. Dužina bloka može biti fiksna, 1, 2, 4, 8, ili 16 reči ili promenljiva.

Slika 4.6 Načini prenosa podataka. [Izvor: Autor]

Poglavlje 5

Prenos podatka kroz internu magistralu

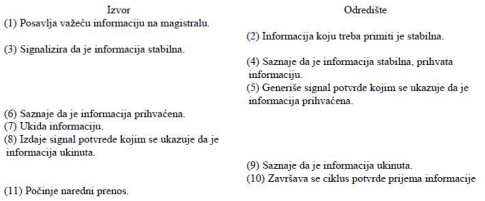
SINHRONIZACIJA KOD PRENOSA

Kada dve jedinice sistema, kao što su to procesor i memorija, razmenjuju podatke, sistemsko rešenje treba da osigura da će u toku prenosa biti ostvarena sinhronizacija u radu obe jedinice.

Ozbiljan nedostatak paketnog prenosa ogleda se u sledećem. Kada prenos počne, a poruka je dugačka, gospodar neće osloboditi magistralu sve dok se prenos ne završi. Sa stanovišta obrade može se desiti da se propusti analiza nekog bitnog ili prioritetnijeg događaja. Da bi se ovaj problem uspešno rešio, mora da postoji mehanizam *istiskivanja*. To znači da se tekućem gospodaru magistrale signalizira kada neki potencijalni gospodar višeg nivoa prioriteta zahteva dodelu magistrale. Nakon prihvatanja zahteva tekući gospodar magistrale prekida prenos, oslobađa magistralu, a kasnije završava sa započetim prenosom.

Sinhronizacija kod prenosa

Kada dve jedinice sistema, kao što su to procesor i memorija, razmenjuju podatke, sistemsko rešenje treba da osigura da će u toku prenosa biti ostvarena sinhronizacija u radu obe jedinice. Jedna tipična sekvenca događaja, kada izvor upisuje informaciju u odredište, ima sledeći oblik:



Slika 5.1 Sinhronizacija prenosa podataka. [Izvor: Autor]

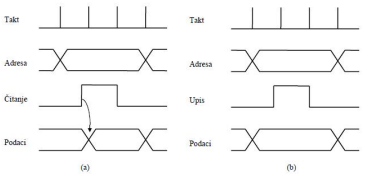
Analizirana sekvenca se može realizovati koristeći se jednim od sledeća tri rešenja vremenskog vođenja događaja:

- sinhron prenos- svi događaji se dešavaju u fiksnim vremenskim intervalima.
- asinhron prenos- signali u sekvenci se generišu u proizvoljnim vremenskim intervalima.
- polusinhron prenos- signali u sekvenci se generišu u fiksnim trenucima u odnosu na taktni signal.

Saglasno ovome magistrale delimo na sinhrone, asinhrone i polusinhrone.

Sinhrone magistrale

Kod sinhronih magistrala prenos podataka je kontrolisan globalnim taktom koji se generiše od strane zajedničkog oscilatora. Opšta sekvenca događaja kod sinhronih magistrala prikazana je na slici 9.



Slika 5.2 Sekvence prenosa podataka. [Izvor: Autor]

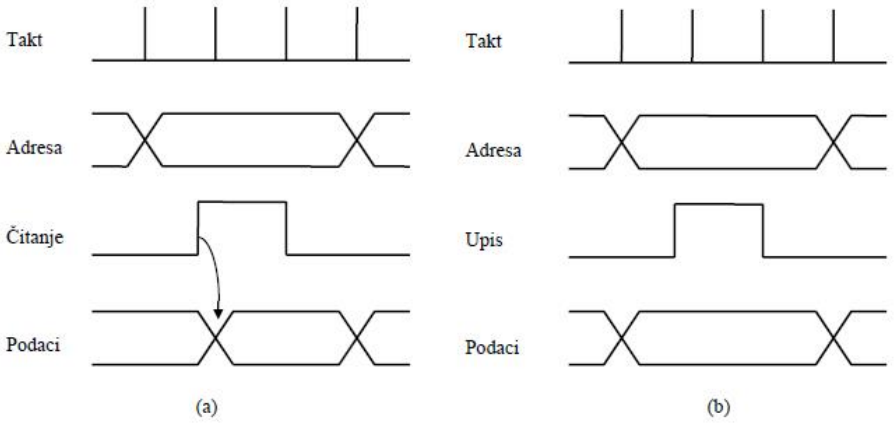
SINHRONE MAGISTRALE

Sve komponente, kod ispravnog prenosa, mora da rade sa brzinom koja je diktirana od strane glavnog takta.

Sinhrone magistrale

Kod sinhronih magistrala prenos podataka je kontrolisan globalnim taktom koji se generiše od strane zajedničkog oscilatora. Opšta sekvenca događaja kod sinhronih magistrala prikazana je na slici 3.

Na slici 3a opisana je sekvenca događaj kod operacije čitanje, a na slici 3b sekvenca događaja kod operacije upis.
I pored toga što se prenos podataka kod ovih magistrala obavlja veoma brzo, osnovni nedostatak je taj što sve komponente, kod ispravnog prenosa, mora da rade sa brzinom koja je diktirana od strane glavnog takta.



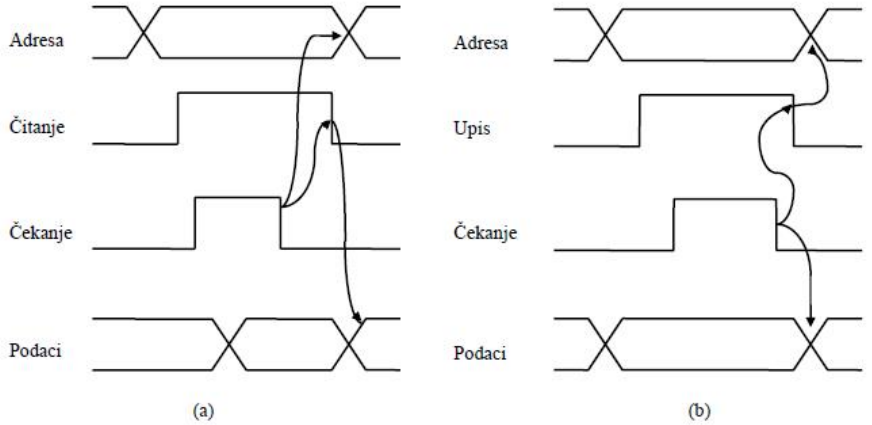
Slika 5.3 Sekvence prenosa podataka. [Izvor: Autor]

ASINHRONE MAGISTRALE

Kod asinhronih magistrala sinhronizacija prenosa nije diktirana glavnim taktom. Metod asinhronog prenosa podataka se više primenjuje kod magistrala mikroračunarskih sistema.

Asinhrone magistrale

Kod ovih magistrala sinhronizacija prenosa nije diktirana glavnim taktom. Metod asinhronog prenosa podataka se više primenjuje kod magistrala mikroračunarskih sistema. Opšta sekvenca prenosa prikazana je na slici 4 . Slika 4a se odnosi na operaciju čitanje, a slika 4b na operaciju upis.



Slika 5.4 Operacija čitanja i upisa. [Izvor: Autor]

Obično asinhrone magistrale zahtevaju "signal priznavanja" (potvrde) koga, da bi se rad magistralnog ciklusa okončao generiše adresirani uređaj. Brzi uređaji potvrđuju priznavanje odmah nakon dekodiranja adrese, dok spori uređaji unose čekanje sve dok ne budu apremni da završe ciklus. U oba slučaja efekat je isti i svodi se na generisanje WAIT ili READY signala.

Suština prenosa sastoji se u tome što u toku svakog ciklusa svaki adresirani uređaj mora generisati signal priznavanja. Drugim rečima. to ne treba da bude isključivi zadatak samo sporih uređaja. Asinhrone magistrale omogućavaju da signal priznavanja bude asinhron u odnosu na sistemski takt, tako da se problem sinhronizovanog prenosa podataka rešava od strane mikroporocesora ili dodatno ugrađenog hardvera.

Jedna od prednosti ovog pristupa je povećana sistemska sigurnost. Naime, ako se pristupa nepostojećoj memorijskoj lokaciji ili U/I portu, sistem se automatski zaustavlja, pošto će on zauvek čekati na priznavanje. Da bi se rešio problem beskonačnog zaustavljanja, u sistem se ugrađuje posebna jedinica nazvana pas-čuvar (watchdog) čiji je prvenstveni zadatak da uslovi prelaz na izvršenje rutine za obradu greške kada dođe do zastoja.

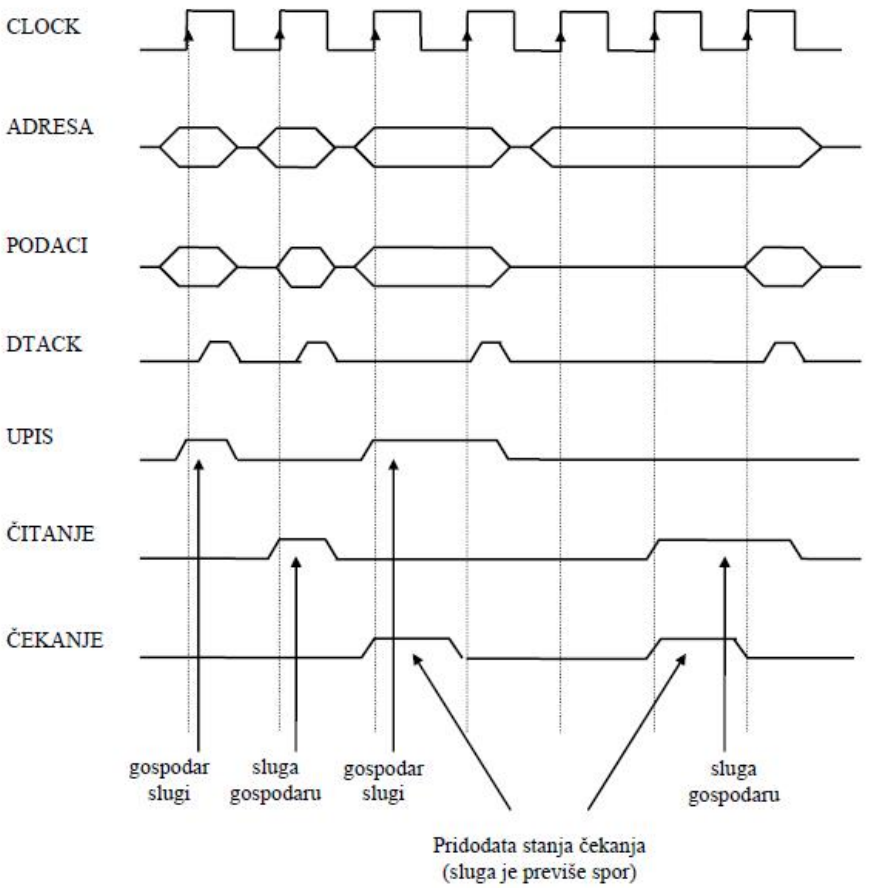
POTVRDA PODATAKA KOD ASINHRONIH MAGISTRALA

Asinhrona magistrala se može implementirati kod mikroprocesora uz pomoć signala READY ili WAIT i ugradnjom dodatne logike.

Asinhrona magistrala se može implementirati kod mikroprocesora uz pomoć signala READY ili WAIT i ugradnjom dodatne logike. READY ulaz mikroprocesora se uobičajeno nalazi u stanju "nisko", a postavlja se na "visoko" kada se od strane adresiranog uređaja primi signal priznavanja.

Ovaj princip rada se ponekad zove *normalno-nespreman*. Kod sistema tipa normalno-spreman, periferno ili memorijsko interfejs kolo mora da invertuje READY kako bi se ubacila stanja čekanja (Slika 5). Signal READY se mora invertovati u okviru fiksnog vremenskog intervala, a ako se ne invertuje magistralni ciklus produžava bez stanja čekanja. Shodno prethodnom, normalno-nespremna asinhrona magistrala može tolerisati rad sa sporijim uređajima.

Mikroprocesor *Motorola MC68000* direktno podržava rad asinhrone magistrale, kod koga DTACK (data transfer acknowledge) ulaz zamenjuje liniju READY/WAIT koja se koristi kod najvećeg broja mikroprocesora.



Slika 5.5 Operacija čitanja i upisa kod asinhronih magistrala. [Izvor: Autor]

POLUSINHRONE MAGISTRALE

Kod polusinhronih magistrala postoje upravljački signali čije se promene dešavaju u trenucima određenim od strane fiksne taktne pobude.

Polusinhrone magistrale

Kod polusinhronih magistrala postoje upravljački signali čije se promene dešavaju u trenucima određenim od strane fiksne taktne pobude. Vremenski interval između upravljačkih signala može biti promenljiv ali samo u umnošcima taktne pobude. Signal fiksne taktne pobude može biti generisan od strane glavnog sistemskog oscilatora ili od strane tekućeg gospodara magistrale.

Veliki broj današnjih mikroprocesora koristi polusinhroni pristup za upravljanje prenosom podataka na magistrali. Mikroprocesor obično ne očekuje odziv od memorije ili perifernog uređaja. Ipak "*wait*" ulaz CPU-a treba da bude potvrđen ako memorija ili periferni sklop treba da produže magistralni ciklus. Kako se takt mikroprocesora ne može zaustaviti, produžava se ciklus magistrale.

Mikroprocesori Z80, 8085, 8086 i drugi koriste ovakav princip kod sinhronizacije u prenosu podataka.

Prednost polusinhronog prenosa ogleda se u manjoj osteljivosti na smetnje, jer se samo signali smetnji koji su na upravljačkim linijama u toku usponske ili opadajuće ivice takta mogu pogrešno interpretirati. Nedostatak ovog prenosa je taj što se vreme odziva zaokružuje na umnožak taktnog perioda.

Poglavlje 6

Pokazne Vežbe

ARBITRAŽA NA MAGISTRALI

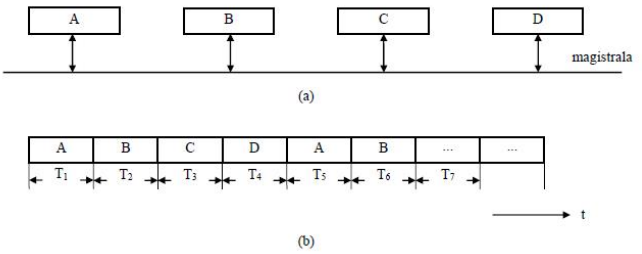
Obično se na magistralu priključuje veći broj potencijalnih gospodara, ali se ona u datom trenutku može dodeliti samo jednom.

Arbitraža na magistrali

Obično se na magistralu priključuje veći broj potencijalnih gospodara, ali se ona u datom trenutku može dodeliti samo jednom. Arbitražni mehanizam magistrale garantuje da će se dodela ostvariti bez konflikta. Naime, u datom trenutku samo će jedan gospodar upravljati magistralom, a ostali se takmiče da bi dobili pravo upravljanja nad njom. Arbitraža se može izvesti kao statička ili dinamička.

***Statička arbitraža na magistrali (20* minuta)**

Kod ove arbitraže raspored transakcija na magistrali između potencijalnih gospodara vrši se po unapred određenom redosledu. Obično, kao što je prikazano na slici 1 , redosled dodeljivanja je kružni.



Slika 6.1 Statička arbitraža na magistrali. [Izvor: Autor]

Bilo kakav prenos, sinhroni, asinhroni ili polusinhroni, može se koristiti za prenos podataka, ali se u najvećem broju slučajeva radi o sinhronom prenosu, jer on svakom potencijalnom gospodaru garantuje određeni broj transakcija po jedinici vremena. Prednost ovakvog pristupa je jednostavnost izvođenja, tj. ugradnja jednostavnog hardvera uz veliku propusnost u toku prenosa podataka. Nedostatak ove dodele je što ona ne uzima u obzir realne potrebe za prenosom. Drugim rečima, magistrala se dodeljuje onom gospodaru koji u tom trenutku nema potrebe za njom, tj. on okupira magistralu NOP operacijama.

Dinamička dodela magistrale

Ovaj tip dodele obezbeđuje da se vlasnik magistrale određuje dinamičkim putem, po zahtevu. Dodela i oslobađanje magistrale ostvaruje se određenom politikom. U daljem tekstu analiziraćemo detaljnije ove politike.

Politike dodele magistrale

Kada potencijalni gospodar želi da obavi transakciju na magistrali, on izdaje zahtev za korišćenjem. Slični zahtevi se mogu javiti i od strane drugih potencijalnih gospodara. Kada tekući gospodar oslobodi upravljanje nad magistralom, on mora da odluči kom će potencijalnom gospodaru predati upravljanje.

POLITIKE DODELE MAGISTRALE

Da bi se to odredilo mora da postoji politika dodele koja može biti: Zasnovana na prioritetu, Zasnovana na nepristrasnosti ili kombinovana.

Da bi se to odredilo mora da postoji politika dodele koja može biti:

• **Zasnovana na prioritetu** - svakom potencijalnom gospodaru je dodeljen fiksni prioritet. Magistrala se dodeljuje gospodaru sa najvišim prioritetom.

• **Zasnovana na nepristrasnosti** - u situacijama kada potencijalni gospodari imaju isti prioritet, svakom gospodaru koji je izdao zahtev za dodelu se mora garantovati dodela magistrale pre nego što se bilo kom drugom gospodaru po drugi put ona ponovo dodeli.

• **Kombinovana** - politike zasnovane na prioritetu i nepristrasnosti se mogi kombinovati. Obično se zahtevima najvišeg prioriteta dodeljuje politika zasnovana na prioritetu, a zahtevima najnižeg prioriteta politika zasnovana na nepristrasnosti.

Multiprocesorski sistemi standardno koriste kombinovanu politiku.

Politike oslobađanja magistrale

Za oslobađanje magistrale koriste se sledeće politike:

• **Oslobađanje po zahtevu** - tekući gospodar magistrale ima

pristup magistrali sve dok se ne generiše drugi zahtev, tj. on zadržava pravo upravljanja nad magistralom i pored toga što ne koristi magistralu.

• **Oslobađanje nakon obavljene transakcije** - nakon obavljene transakcije gospodar oslobađa magistralu.

• **Istiskivanje** - gospodar koji ima viši prioritet u odnosu na tekući, nakon izdavanja zahteva uslovljava da gospodar za nižim prioritetom preda upravljanje nad magistralom i pored toga što nije završio sa prenosom.

Hardverski mehanizmi za arbitražu na magistrali

Hardver za arbitražu može se realizovati kao centralizovani ili distribuirani.

Centralizovana arbitraža

Kod centralizovane arbitraže hardver je koncentrisan na jednom mestu, a može biti lociran u jednom od modula koji se povezuju na magistralu ili izveden kao poseban hardver koji se zove arbitar magistrale.

Gospodar koji zahteva dodelu magistrale predaje zahtev za dodelu centralnom arbitru.

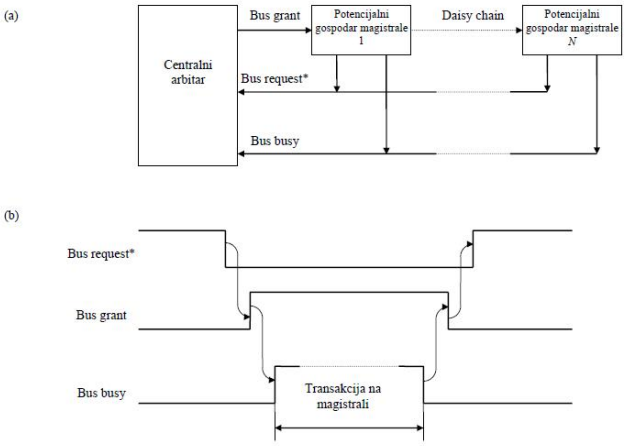
CENTRALIZOVANA ARBITRAŽA (15 MINUTA)

Kod centralizovane arbitraže hardver je koncentrisan na jednom mestu, a može biti lociran u jednom od modula koji se povezuju na magistralu ili izveden kao poseban hardver.

Arbitar odlučuje, ako postoji veći broj zahteva, kom će gospodaru dodeliti magistralu na osnovu politike dodele. Hardverski mehanizmi koji se koriste za dodelu i zahvatanje magistrale mogu se podeliti na sledeće grupe.

Deljivi zahtev i lančano zahvatanje (25 minuta)

Princip ovog metoda prikazan je na slici 2 .



Slika 6.2 Deljivi zahtev i lančano zahvatanje. [Izvor: Autor]

Svaki potencijalni gospodar magistrale izdaje zahtev za dodelu magistrale preko linije "BUS REQUEST*". Zahtev je aktivan kao "nisko", a svi zahtevi su povezani žičanom ILI logikom. Kada centralni arbitar primi zahtev, on predaje signal "BUS GRANT" potencijalnom gospodaru magistrale označenom sa 1. Linija "BUS GRANT" povezuje sve potencijalne gospodare u lanac (daisy chain), tako da gospodar 1 prosleđuje signal gospodaru 2, itd.

Politika dodele se zasniva na prioritetu. Ovaj gospodar koji je fizički bliži centralnom arbitru ima viši prioritet. Ako potencijalni gospodar magistrale ne zahteva dodelu on predaje signal dodele narednom gospodaru u lancu sve dok se ne naiđe na gospodara koji je izdao zahtev. Gospodar koji je izdao zahtev ne prenosi dalje signal dodele kroz lanac (prekida lanac) i aktivira liniju "BUS BUSY" na visoko čime ukazuje da je zahvatio magistralu. Kada tekući gospodar magistrale završi sa prenosom, postavlja liniju "BUS BUSY" na nisko, "BUS GRANT" je već postavljena na "nisko" od strane arbitra i naredni ciklus arbitraže može da počne.

Prednost ove šeme je jednostavnost izvođenja, a nedostatak što je prioritet dodele određen fizičkom pozicijom. Naime, može se desiti da gospodar najnižeg prioriteta ne dobije magistralu na dodelu ako se zahtevi za dodelu od strane gospodara sa višim prioritetom često izdaju, tj. kaže se da će on "umreti od gladi".

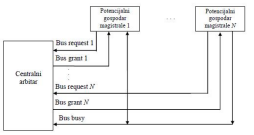
DELJIVI ZAHTEV I LANČANO ZAHVATANJE (15 MINUTA)

Principska šema realizacije kombinovanog mehanizma arbitraže se zasniva na korišćenju prethodna dva arbitražna mehanizma.

Nezavisni zahtevi i zahvatanja - Princip ovog metoda prikazan je na slici 3 . Svakom potencijalnom gospodaru magistrale se dodeljuje posebna linija za izdavanje zahteva i posebna linija za zahvatanje preko kojih se povezuje sa centralnim arbitrom. Kada potencijalni gospodar želi dodelu on aktivira svoju liniju "BUS REQUESTn". Arbitar odabira potencijalnog gospodara i aktivira odgovarajuću liniju "BUS GRANTn". Odabrani gospodar deaktivira liniju "BUS REQUEST" i aktivira linju "BUS BUSY" ukazujući drugim gospodarima da je on taj koji trenutno koristi magistralu. Kada tekući gospodar želi da oslobodi magistralu on deaktivira liniju "BUS BUSY", nakon čega arbitar može dodeliti magistralu drugom gospodaru ako je neki od njih izdao zahtev.

Politika dodele magistrale može biti različita (kružna, sa prioritetom i dr.) i zavisi od realizacije centralnog arbitra.

Prednost ove šeme je kratko vreme arbitraže, a nedostatak je veliki broj linija za povezivanje potencijalnih gospodara magistrale i centralnog arbitra.



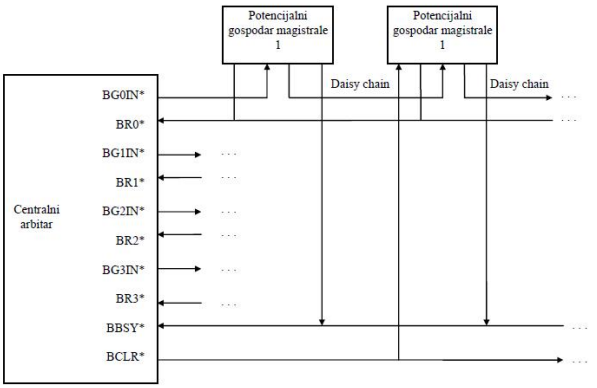
Slika 6.3 Nezavisni zahtevi i zahvatanja. [Izvor: Autor]

Kombinovani mehanizam

Principska šema realizacije kombinovanog mehanizma arbitraže prikazana je na slici 4 . Sistem se zasniva na korišćenju prethodna dva arbitražna mehanizma.

Distribuirana arbitraža

Kod ove šeme, hardver za arbitražu je raspodeljen po potencijalnima gospodarima magistrale. Ovakvo rešenje se koristi kod multiprocesorskih sistema koji rade u čvrstoj sprezi.



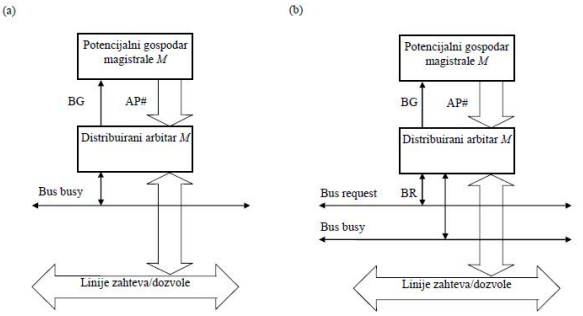
Slika 6.4 Kombinovani mehanizam. [Izvor: Autor]

PRINCIP RADA DISTRIBUIRANE ARBITRAŽE (15 MINUTA)

Princip rada distribuirane arbitraže je zasnovan na tome da potencijalni gospodar magistrale koji zahteva upravljanje izdaje zahtev aktiviranjem linija #AP prema distribuiranom arbitru.

Princip rada distribuirane arbitraže je prikazan na slici 5 . Potencijalni gospodar magistrale koji zahteva upravljanje izdaje zahtev aktiviranjem linija #AP prema distribuiranom arbitru. Arbitar predaje *#AP* zahtev na deljive linije "REQUEST/GRANT". Svi ostali gospodari koji izdaju zahtev za dodelu takođe predaju svoj zahtev preko svojih #AP linija.

Treba istaći da se svakom *#AP* zahtevu dodeljuje po jedna linija na zajedničkim linijama "*REQUEST/GRANT*". Na ovaj načina se formira zbirni #AP zahtev. Nakon ovoga, svaki distribuirani arbitar poredi svoj zahtev sa zbirnim. Ako je njegov #AP zahtev niži od zbirnog to znači da je njegov prioritet niži, tj. magistrala se dodeljuje onom gospodaru čiji je prioritet najviši.



Slika 6.5 Princip rada distribuirane arbitraže. [Izvor: Autor]

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 7

Zadaci za samostalni rad

ZADACI ZA SAMOSTALNI RAD #5

Zadaci za samostalni rad rade se okvirno 45 minuta

Zadatak #1 (15 minuta):
Sastaviti memorijski modul 32k x 48 od modula 8k x 8.

Zadatak #2 (15 minuta):

Sastaviti memorijski modul 64k x 48 od modula 8k x 8.

Zadatak #3 (15 minuta)

Sastaviti memorijski modul 128k x 64 od modula 32k x 8.

Poglavlje 8

Domaći zadatak

DOMAĆI ZADATAK #8

Domaći zadatak #8 odrađuje se okvirno 45 minuta

Napomena:

Svaki student dobija jedinstveni domaći zadatak jer zavisi od broja indeksa.

Predaja domaćeg zadatka:

Domaći zadatak slati odgovarajućem predmetnom asistentu, sa predmetnim profesorom u CC.

Predati domaći zadatak koristeći .doc/docx uputstvo dato u prvoj lekciji.

Domaći zadatak #5

Student bira zadatak prema broju indeksa po pravilu:

broj_indeksa % 10 + 1

Potrebno je od datih memorijskih modula sastaviti druge memorijski modul horizontalnim i vertikalnim proširenjem. Opisati sve korake proširenja.

1. Sastaviti memorijski modul 32k x 8 od modula 16k x 8 .
2. Sastaviti memorijski modul 32k x 16 od modula 16k x 8 .
3. Sastaviti memorijski modul 32k x 24 od modula 16k x 8 .
4. Sastaviti memorijski modul 64k x 16 od modula 16k x 8 .
5. Sastaviti memorijski modul 64k x 8 od modula 16k x 8 .
6. Sastaviti memorijski modul 32k x 8 od modula 8k x 8 .
7. Sastaviti memorijski modul 32k x 16 od modula 8k x 8 .
8. Sastaviti memorijski modul 32k x 24 od modula 8k x 8 .
9. Sastaviti memorijski modul 64k x 16 od modula 8k x 8 .
10. Sastaviti memorijski modul 64k x 8 od modula 8k x 8 .

Zaključak

MIKROPROGRAMIRANJE – NAJBOLJA (SISTEMSKA) METODA ZA REALIZACIJU UPRAVLJAČKE JEDINICE

Rezime lekcije #5

U ovoj lekciji prvo je pokazan najjednostavniji mogući procesor gde je moguće objasniti principe rada procesora iako se radi o veoma softificiranom delu računara (ako ne i najkomplikvaniji).

Zatim su na primeru jednostavnog programa prikazani vremenski dijagram stanja na spoljnim sabirnicama i promene u sadržajima registara modela u cilju objašnjenja rada mikroprocesora.

Program čini skup instrukcija koje su smeštene u memoriji. CPU tipično čita (pribavlja) instrukcije iz memorije po jednu u datom trenutku, izvršava svaku instrukciju, a zatim pribavlja narednu. Proces se ponavlja beskonačno dugo. Ova sekvenca je poznata kao ciklus pribavljanje-dekodiranje-izvršenje.

Magistrala se koristi za povezivanje dva ili većeg broja sistemskih elemenata. Ona predstavlja skup linija (veza). Skup veza se deli od strane sistemskih elemenata, ali se takođe i koristi od strane sistemskih elemenata za međusobnu komunikaciju. Pojam magistrala obično ukazuje da su veze paralelne, pri čemu se duž istog puta prenosi po nekoliko signala.

Literatura:

A. Tanenbaum, Structured Computer Organization, Chapter 03, pp. 185 - 196.