

Osnovi mikroprocesorskih i mikrokontrolerskih sistema - Memorijski podsistem

prof. dr Ivan Mezei

28. mart 2022.

Glava 4

Memorijski podsistem

Memorijski podsistem je značajan deo svakog mikroračunarskog sistema i od njegovog kvaliteta zavisi i celokupni kvalitet sistema. Uzimajući u obzir proces izvršavanja svakog programa koji se sastoji od niza instrukcija koje je neophodno pre izvršavanja zahvatiti iz memorije, jasno je da pristup memoriji značajno utiče na performanse celog sistema (pogledati temu "Memorijski zid" u sekciji ??).

Idealna memorija bi bila ona koja je proizvoljnog kapaciteta, kod koje je moguć trenutni pristup podatku proizvoljne veličine i njegova pojava na izlazu memorije bez kašnjenja. Naravno, takva memorija ne postoji, pa su još davne 1946. godine *Burks, Goldstine, i von Neumann* uvideli potrebu i značaj memorijske hijerarhije koja govori o tome da su u mikroračunarskim sistemima potrebne memorije različite veličine i brzine [1].

U nastavku će biti obrađene sledeće teme u vezi memorijskog podsistema: principi lokalnosti, memorijska hijerarhija, svojstva memorija, tehnologija izrade memorija, klasifikacija memorija, skrivena memorija, operativna memorija, sprega procesora i memorije, memorijski adresni prostor i adresni dekodori.

4.1 Principi lokalnosti

U domenu memorijskih podsistema dva principa lokalnosti se koriste: princip prostorne i princip vremenske lokalnosti. Ovi principi su nastali na bazi činjenica da ako je neki podatak korišćen, velika je verovatnoća da će biti potreban neki njemu blizak (*princip prostorne lokalnosti*). Sa druge strane, ukoliko je neki podatak korišćen u nekom trenutku, velika je verovatnoća da će biti korišćen ponovo u

nekom bliskom narednom trenutku (*princip vremenske lokalnosti*).

4.1.1 Princip prostorne lokalnosti

Princip prostorne lokalnosti podrazumeva bliskost generisanih memorijskih adresa. Ako je mikroprocesor generisao adresu a u trenutku t onda je velika verovatnoća da će mikroprocesor generisati adresu iz opsega $(a - \Delta a, a + \Delta a)$ u intervalu vremena $(t + \Delta t)$, za malo Δt i Δa .

Ovaj princip kaže da mikroprocesor grupiše generisane adrese u nekom periodu vremena u relativno malom opsegu memorijskih adresa. Na primer, ako mikroprocesor izvršava petlju, instrukcije u telu petlje smeštene su u memorijskim lokacijama sa bliskim adresama. Ovaj princip je dodatno izraženiji jer mikroprocesor sekvencijalno izvršava instrukcije, pa ako je u periodu i pristupio adresi a , onda je najveća verovatnoća da će u periodu $i+1$ mikroprocesor pristupiti adresi $a+1$.

U skladu sa tim jasno je da mikroprocesor ne generiše adrese potpuno slučajno i da su generisane memorijske adrese lokalizovane. Lokalnost generisanih adresa potiče od činjenice da mikroprocesor izvršava instrukcije koje su sekvencijalno smeštene u memoriji, da često izvršava petlje, obrađuje podatke koji su u tabelama i slično. Generisane adrese su lokalizovane u nekom vremenskom intervalu u jednom opsegu adresa, posle toga, mikroprocesor može da izvrši npr. programski skok, pa će u narednom vremenskom intervalu adrese biti lokalizovane u nekom drugom opsegu adresa.

Princip lokalnosti generisanih adresa u memorijskoj hijerarhiji koristi se tako što kada mikroprocesor generiše neku adresu, onda se iz memorije na nižem hijerarhijskom nivou prenese u memoriju na višem hijerarhijskom nivou blok memorijskih lokacija koje su u okolini navedene memorijske adrese. Očekuje se da će mikroprocesor generisati adresu koja je iz datog bloka, koji je sada u višem hijerarhijskom nivou, pa će tako pristup biti mnogo brži, nego da je mikroprocesor pristupa istoj memorijskoj lokaciji koja je na nižem hijerarhijskom nivou.

4.1.2 Princip vremenske lokalnosti

Princip vremenske lokalnosti povezan je sa verovatnoćom ponovnog korišćenja podatka u narednom bliskom vremenskom trenutku. To znači da ukoliko je mikroprocesor koristio podatak p u trenutku t onda je velika verovatnoća da će mikroprocesor ponovo koristiti podatak p u intervalu vremena $(t + \Delta t)$, za malo Δt .

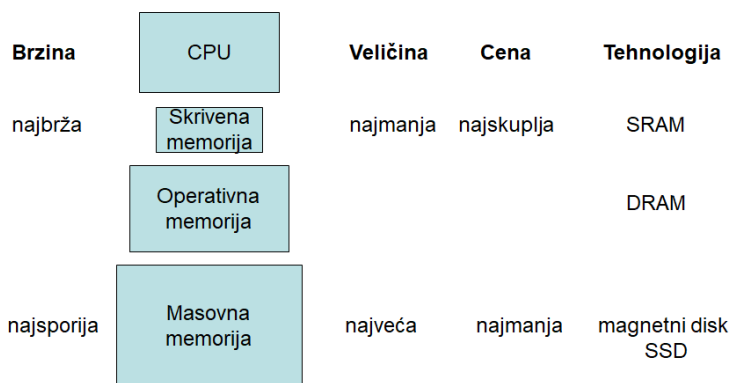
Drugim rečima, mikroprocesor veoma često ponovo koristi neke podatke ili promenljive. Na primer, ako mikroprocesor izvršava petlju, onda u svakom prolazu

petlje koristi brojač koji kontrolira broj izvršavanja petlje.

Princip ponovnog korišćenja podataka u realizaciji memorijske hijerarhije koristi se tako što kada mikroprocesor koristi neki podatak, onda se taj podatak privremeno još neko vreme čuva u memoriji na višem hijerarhijskom nivou. Ako mikroprocesor bude koristio isti podatak koji je u bržoj memoriji, onda će mikroprocesor brže pristupiti tom podatku pa će tako sistem brže da radi.

4.2 Memorijska hijerarhija

Značaj postojanja memorijske hijerarhije, kao što je u uvodu rečeno, poznat je od ranih dana računarstva. Da bi bolje razumeli koncept i potrebu memorijske hijerarhije ovde će biti opisan primer i analogija sa bibliotekom. Ukoliko student dolazi u biblioteku i želi da prouči neku temu iz različitih knjiga, jasno je da će odabrati nekoliko njih i poneti sa sobom do radnog stola. Kada počne sa radom čitaće neka poglavlja iz jedne knjige, neka iz druge pa ponovo iz prve itd. i vodiće beleške u svesci.

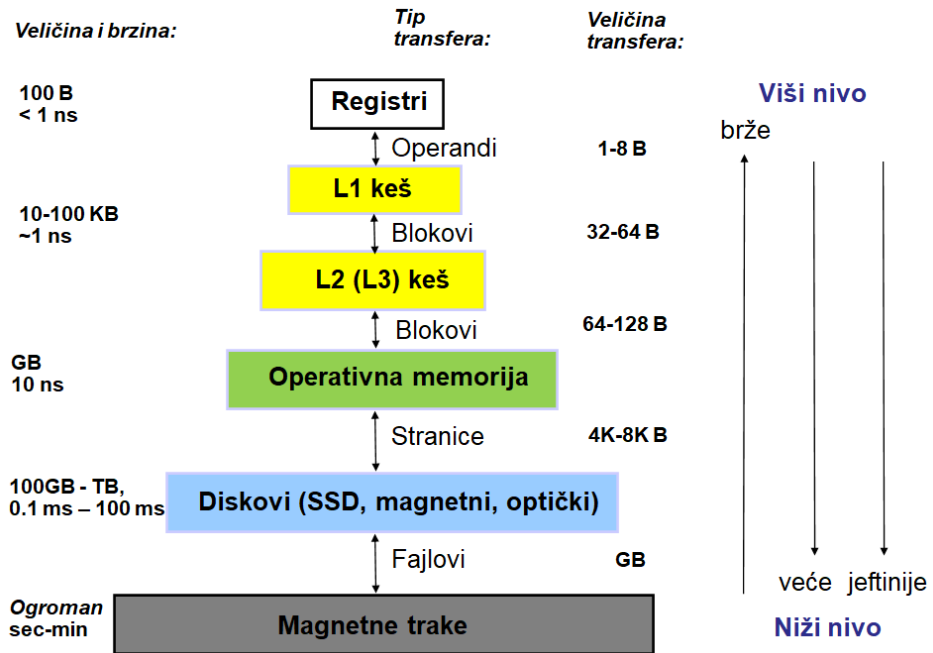


Slika 4.1: Memorijska hijerarhija

Iz prethodnog primera se može zaključiti da je potrebno postojanje velike (masovne) spore memorije (u primeru to je biblioteka), operativne (radne) memorije (u primeru to je sveska) kao i male lokalne brze memorije (u primeru to su knjige). Ovakva hijerarhija memorija koja se sastoji od različitih memorija određenih prema veličini i brzini (a i ceni) se naziva **memorijska hijerarhija**. Osnovna zakonitost koja važi je da što je memorija veća to je sporija i jeftinija, a što je manja

to je brža i skuplja po jedinici kapaciteta. Na slici 4.1 je dat jedan način prikaza memorijske hijerarhije.

Na slici 4.2 je detaljnije prikazana savremena memorijska hijerarhija. Može se uočiti veličina pojedinih tipova memorija, njihova brzina, kao i tipovi i veličine transfera među susednim memorijama u hijerarhiji. Pored toga vidi se i trend da sa prelaskom na više nivoe hijerarhije su brzine sve veće, ali zato sve skuplje i sve manje po jedinici kapaciteta.



Slika 4.2: Savremena memorijska hijerarhija

4.3 Svojstva memorija

Najznačajniji tehnološki parametri memorije su brzina, način pristupa, trajnost upisanih informacija, mogućnost izvršavanja operacije upisa i realizacija memorijskih ćelija. Na osnovu ovih svojstava, moguće su različite klasifikacije prema:

brzini, načinu pristupa, trajnosti upisanih informacija, mogućnosti upisa i načinu realizacije memorijskih ćelija.

Brzina memorije

Određuje se na osnovu vremena koje protekne od trenutka kada su aktivirani upravljački i adresni signali do trenutka kada se, kod operacije upisa, ulazna memorijska reč smesti u adresiranu memorijsku lokaciju, odnosno kod operacije čitanja, kada se memorijska reč iz adresirane memorijske lokacije prenese na spoljne linije za podatke. Ova definicija je neformalna i odnosi se na najnepovoljniji slučaj – drugim rečima navedena vremena su takva da garantuju da će memorija obaviti predviđene operacije.

Način pristupa

Može biti sekvencijalan, slučajan ili kombinovan. Memorijske sekvencijalne pristupom obično imaju upisno/učitnu glavu koja obavlja operaciju upisa i čitanja nad elementom koji je u kontaktu sa glavom. Kod sekvencijalnog pristupa, upisno/učitna glava mora preći preko svih elemenata (lokacija) koji se nalaze između elementa na kome se trenutno nalazi glava do zahtevanog elementa. Tipičan primer memorije sa sekvencijalnim pristupom je magnetna traka kod koje upisno/učitna glava mora da pređe preko dela trake između trenutnog položaja glave do adresiranog elementa na traci. Očigledno je da vreme pristupa kod sekvencijalnih memorija zavisi od trenutne pozicije upisno/učitne glave u odnosu na element na kome se pristupa.

Kod memorija sa slučajnim pristupom vreme pristupa za sve lokacije je isto. Primeri ovakvih memorija su poluprovodničke memorije tipa RAM (eng. *Random Access Memory*) i ROM (eng. *Read Only Memory*).

Neke vrste masovnih memorija, kao što su optički i feromagnetni diskovi, imaju način pristupa koji je poboljšan u odnosu na sekvencijalni pristup. Pristup adresiranom elementu obavlja se u dva koraka: u prvom koraku se upisno/učitna glava direktno pozicionira iznad kružne staze na kojoj se nalazi traženi element, a zatim se sekvencijalnim pristupom dolazi do tog elementa.

Trajnost upisanih informacija

Razlikujemo memorije kod kojih nestanak napona napajanja dovodi do gubitka svih zapisanih informacija i memorije koje zadržavaju sadržaj posle prestanka napona napajanja. Praktično sve optičke memorije i memorije sa feromagnetnim ćelijama, na primer magnetni diskovi i magnetne trake, zadržavaju sadržaj posle prestanka napona napajanja. Poluprovodničke memorije tipa ROM takođe zadržavaju sadržaj posle prestanka napona napajanja.

Poluprovodničke memorije tipa RAM gube sadržaj kada se isključi napon napajanja. Međutim, klasi memorija koje se nazivaju dinamički RAM, nije dovoljan neprekidan napon napajanja da bi se upisani sadržaj održao duže vreme. Memorijske ćelije dinamičkog RAM-a napravljene su u obliku kondenzatora minijaturnih dimenzija. Stanje memorijskih ćelija razlikuje se po količini naelektrisanja u kondenzatoru: ako je kondenzator napunjen, ćelija je u stanju logičke 1, a ako je prazan, ćelija je u stanju logičke 0. Napunjen kondenzator se vremenom prazni i time se gubi sadržaj dinamičkog RAM-a. Stoga je neophodno periodično osvežavanje, koje se ostvaruje tako što se redom čitaju sve memorijske lokacije. Kod operacije čitanja, elektronska kola za čitanje detektuju stanje kondenzatora, proslede signale na linije za podatke, a zatim pročitani sadržaj ponovo zapišu u istu memorijsku lokaciju i na taj način regenerišu napunjenost kondenzatora. Osvežavanje dinamičkih RAM-ova standardno se radi u intervalima reda veličine 2 ms, što znači da se u tom vremenskom intervalu moraju osvežiti sve memorijske ćelije.

Mogućnost upisa

U nekim praktičnim primenama operacija upisa u memoriju nije neophodna. Na primer, programi za obavljanje operacija u kalkulatoru trajno su zapisani u memoriji kalkulatora i nema potrebe da se menjaju. U ovakvim primenama koristi se memorija tipa ROM kod kojih se sadržaj jednom upiše i kasnije se ne menja. Kod poluprovodničkih memorija tipa ROM sadržaj se formira u toku izrade integrisanog kola. Postoji varijanta programabilnog poluprovodničkog ROM-a (PROM), kod koga su sve ćelije u jednom stanju, a korisnik može proizvoljne ćelije posebnim postupkom prevesti u drugo stanje. Postupak prevođenja ćelija u drugo stanje nije inverzan, tj. memorijske ćelije ne mogu se vratiti u početno stanje. Postupak upisa sadržaja u PROM naziva se programiranje i obavlja se primenom posebnog uređaja koji se naziva PROM programator.

Poluprovodničke memorije tipa EPROM (eng. *Erasable PROM*) mogu se programirati (upisati novi sadržaj primenom EPROM programatora) i poseduju mogućnost brisanja sadržaja, odnosno prevođenja memorijskih ćelija u početno stanje.

Programiranje ROM-a obavlja se tako što se integrisano kolo stavi u podnožje programatora i zatim aktivira postupak kojim se izabrane memorijske ćelije prevode u traženo stanje. Prevođenje ćelija najčešće se svodi na primenu električnih struja koje spaljuju veze između vrsta i kolona kod PROM-a ili na primenu električnog polja koje prevodi nosioce naelektrisanja na provodna ostrvca u izolatoru kod EPROM-a. U oba slučaja postupak je relativno spor. Važno je zapaziti da programiranje ROM-a ne može da obavi mikroprocesor koji je neposredno spregnut sa PROM-om ili EPROM-om u mikroračunarskom sistemu.

Razlikuju se dva načina brisanja sadržaja, jedan koji koristi ultravioletnu sve-

tlost (UV EPROM) i drugi koji koristi efekat električnog polja (EEPROM). Postupak brisanja UV EPROM-a sastoji se u izlaganju integrisanog kola ultraljubičastoj svetlosti koja ima dovoljnu energiju da neutrališe naelektrisanje provodnih ostrvaca u izolatoru. Kod EEPROM-a koristi se električno polje za neutralisanje naelektrisanja provodnih ostrvaca.

Realizacija memorijskih ćelija

Memorija se može realizovati na različite načine u zavisnosti od tehnologije izrade memorijskih ćelija. Realizacija memorijskih ćelija mora biti takva da omogućujući pouzdano raspoznavanje dva različita stanja. Memorijske ćelije mogu da se realizuju na sledeće načine kao:

- poluprovodnički flip-flopovi, koji mogu biti u jednom od dva stabilna stanja.
- delići feromagnetnog materijala, koji mogu biti namagnetisani u jednom od dva smera.
- tačke na optičkom disku mogu biti svetle ili tamne, odnosno odbijati ili upijati laserski snop svetlosti.
- kondenzatori malih dimenzija mogu biti napunjeni ili prazni.
- veze između vrsta i kolona u matrici mogu biti uspostavljene ili prekinute.

Od tehnologije izrade zavise svi ostali parametri memorije. Savremeni mikroprocesori mogu da se neposredno spregnu samo sa poluprovodničkim memorijama. Memorije napravljenije primenom drugih tehnologija zahtevaju dodatna elektronska kola ili elektromehaničke podsisteme koji prilagođavaju memoriju uslovima koje zahteva mikroprocesor.

4.4 Tehnologija izrade memorija

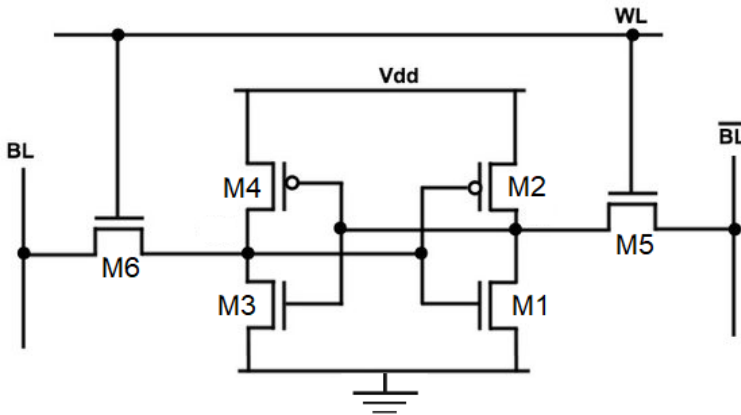
Sa stanovišta tehnologije izrade gde su četiri tehnologije dominantne, današnje memorije se mogu klasifikovati kao:

- SRAM
- DRAM
- FLASH ili
- na magnetnom medijumu.

U nastavku će biti svaka posebno objašnjena.

SRAM

SRAM predstavlja skraćenicu koja potiče od engleskih reči *Static Random Access Memory*. SRAM su memorije koje predstavljaju memorijski niz sa jednim pristupnim portom za čitanje ili upis. SRAM imaju fiksno vreme pristupa, pri tome vreme pristupa može biti različito za čitanje i upis. Ove memorije ne zahte-

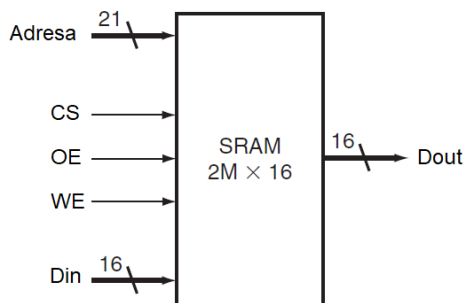


Slika 4.3: SRAM ćelija

vaju osvežavanje pa je vreme pristupa blisko periodu takta. U režimu mirovanja je potrebna minimalna potrošnja snage radi očuvanja sadržaja. Primer standardne SRAM ćelije realizovane pomoću šest mosfet tranzistora je prikazan na slici 4.3. Pristupni tranzistori M5 i M6 se uključuju preko WL (eng. *word line*) linije (ona se naziva i linija reda) služe za dozvolu upisa/čitanja preko bitske linije BL i njene invertovane linije koje se još nazivaju i linije kolone. Tranzistori M1 i M2 čine jedan CMOS inverter, a tranzistori M3 i M4 drugi. Ovi invertori su unakrsno spregnuti. Zavisno od dovedenog signala na bitskoj liniji uključuju se parovi tranzistora M1-M4 ili M2-M3 gde jedan slučaj predstavlja memorisanu 1-bitnu vrednost logičke jedinice, a druga logičke nule. Dobra strana je mala potrošnja jer u ovakvoj CMOS flip-flop konfiguraciji dva invertora potrošnja se javlja samo pri promeni stanja, a u ustaljenom stanju je vrlo mala.

Povezivanjem više ovakvih ćelija po principu matrične organizacije (redovi i kolone) moguće je napraviti SRAM memorijske module. U prošlosti su se posebni

SRAM čipovi koriste u većini personalnih računara i servera za primarne, sekundarne, pa i tercijalne skrivene memorije. Danas, zahvaljujući napretku poluprovodničke tehnologije, u skladu sa Murovim zakonom, te memorije su integrisane na čipu zajedno sa procesorom tako da je proizvodnja nezavisnih SRAM integrisanih kola takoreći prestala da postoji. Na slici 4.4 je prikazan SRAM 2M x 16 modul sa odgovarajućim priključcima. Pored priključaka za adresu, ulazne (Din) i izlazne podatke (Dout), tu su još i signali za: selekciju čipa (CS), dozvolu izlaza (OE) i dozvolu upisa (WE).



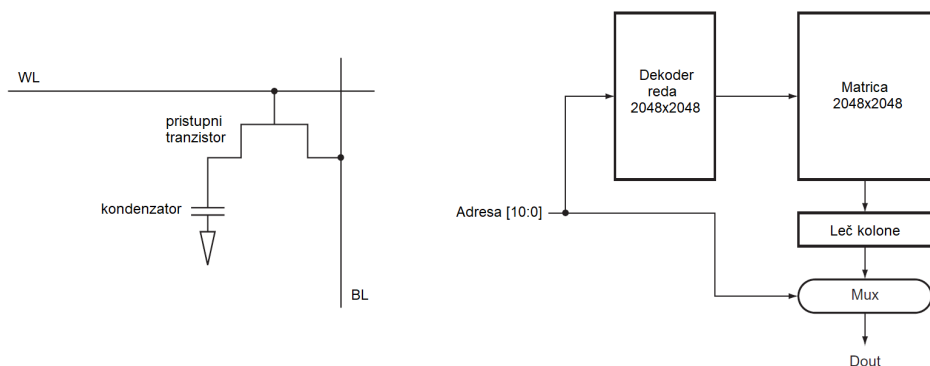
Slika 4.4: SRAM modul

DRAM

DRAM je skraćenica od engleskih reči *Dynamic Random Access Memory*. Kod SRAM dok je napajanje uključeno sadržaj memorijske ćelije je trajno očuvan jer je to rešeno poluprovodnički. Kod DRAM ta vrednost predstavlja naelektrisanje malog kondenzatora koje se postepeno gubi i zato je potrebno periodično vršiti osvežavanje sadržaja DRAM ćelija zbog čega se i zovu dinamička RAM. Za pristup radi čitanja, upisa ili osvežavanja DRAM ćelija se koristi tranzistor što je prikazano na levoj strani slike 4.5, a na desnoj je data logička struktura jedne DRAM memorije.

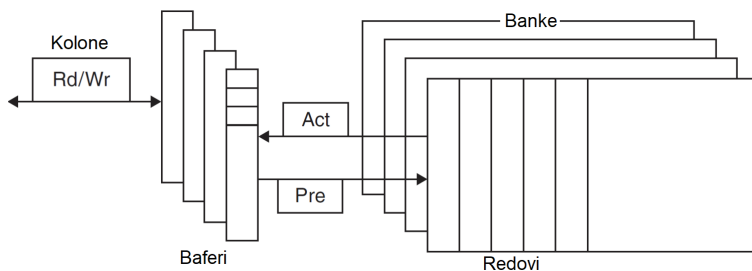
S obzirom da za DRAM treba samo jedan tranzistor (kod SRAM ćelija je potrebno 6-8 tranzistora tipično) ove memorije su gušće i jeftinije po bitu u odnosu na SRAM. Glavna mana je što je DRAM potrebno povremeno osvežavati zbog gubitka naelektrisanja. Srećom to se ne mora raditi često (red veličine svakih nekoliko milisekundi) tako da ostaje dovoljno vremena za pristup memoriji.

Savremene DRAM su organizovane po bankama (videti sliku 4.6), tipično četiri za npr. DDR3 [20]. Svaka banka se sastoji od niza redova. Slanje PRE (eng. *pre-charge*) komande otvara ili zatvara banku. Adresa reda se šalje sa Act (eng. *activate*)



Slika 4.5: DRAM ćelija (levo) i struktura DRAM (desno)

komandom, što uzrokuje transfer reda u bafer. Kada je red u baferu, može se izvršiti njegov transfer sukcesivnim slanjem adresa kolona u širini DRAM-a (obično 4, 8, ili 16 bita kod DDR3) ili specificiranjem blok transfera i početne adrese. Svaka komanda, kao i blok transferi su sinhronizovani taktom.



Slika 4.6: Organizacija DRAM po bankama

Radi daljeg poboljšanja, DRAM se sinhronizuje taktom i takva memorija se zove sinhroni DRAM ili SDRAM. Glavna prednost SDRAM-a je što upotreba takta eliminiše vreme potrebno da se memorija i procesor sinhronizuju. Posebnu prednost SDRAM donosi zbog tzv. *burst* režima transfera kada nije potrebno specificirati dodatne adresne bite. Dalja ubrzanja su primenjena kod DDR (eng. *textitdouble data rate*) memorija gde se koriste i uzlazna i silazna ivica takta i time dobija duplo povećanje protoka podataka.

FLASH

Fleš memorija spada u memorije koje je moguće brisati električnim putem (EEPROM). Glavni problem kod ovih memorija je ograničen broj upisa nakon kojeg se počinju gubiti određeni biti. Da bi se ovaj problem donekle prevazišao većina fleš memorija ima u sebi ugrađen kontroler koji nastoji da upise "razbaca" na takav način da se ćelije ravnomerno koriste. Ova tehnika se na engleskom naziva *wear leveling*. Iako usporava rad sa fleš memorijom, ova tehnika produžava "život" ćelija fleš memorije.

Danas su u upotrebi dva tipa fleš memorije NOR i NAND, iako je NAND dominantniji. NOR tip memorije se koristi kada je neophodna veća brzina čitanja, a manji kapacitet nije problem u poređenju sa NAND. Sa druge strane NAND tip odlikuje znatno veći kapacitet i veća brzina upisa u odnosu na NOR tip.

Danas se FLASH tehnologija veoma koristi u raznim uređajima tipa masovnih memorija (USB flash memorije, SSD uređaji, eksterni diskovi i dr.)

Magnetni medijum

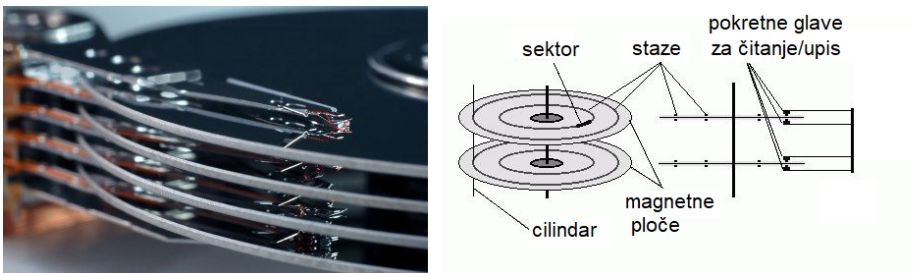
Kao magnetni medijumi u okviru masovnih ili sekundarnih memorija se danas standardno koriste magnetni diskovi. Pored njih se za potrebe čuvanja ogromnih količina podataka koriste i savremene magnetne trake zahvaljujući kompaniji IBM koja ih razvija¹.

Magnetni diskovi se sastoje od skupa metalnih magnetnih ploča koje se rotiraju brzinama od 5400 do 15000 obrtaja po minuti. Metalne ploče su presvučene magnetnim materijalom sa obe strane na koje je moguć upis podataka slično kao kod video i audio kasete. Da bi pročitati ili upisali podatak na hard disk postoje pokretne glave za upis ili čitanje i nalaze se odmah iznad površine magnetnih ploča što je prikazano na levoj strani slike 4.7.

Svaka površina diska je podeljena na koncentrične krugove nazvane staze kojih tipično ima na desetine hiljada po površini. Svaka staza je dalje podeljena na sektore kojih ima na 1000 po stazi. Sektori su tipične veličine 512 do 4096 bajtova. Glave za čitanje se kreću istovremeno na istim lokacijama iznad ploča. Termin cilindar se koristi za sve staze ispod glava na određenom mestu na svim pločama (slika 4.7 desno).

Da bi pristupio podacima, operativni sistem taj proces organizuje u tri faze. Prvi korak je pozicioniranje glave iznad odgovarajuće staze. Ova operacija se zove pretraga (eng. *seek*), a vreme potrebno da se glava pozicionira je vreme pretrage

¹<https://www.ibm.com/it-infrastructure/tape-drives>



Slika 4.7: Magnetne ploče sa glavama (levo) i organizacija diska (desno)

(eng. *seek time*). Uobičajeno srednje vreme koje daju proizvođači hard diskova je između 3 i 13 ms.

Sledeći korak je rotacija diska brzinom od 5400 do 15000 obrtaja po minuti da bi se željeni sektor pozicionirao ispod glave za čitanje/upis. Vreme koje je potrebno za ovo se naziva rotaciono kašnjenje (eng. *rotational latency* ili *rotational delay*). Srednje kašnjenje za brzinu 5400 se dobija kao: $0.5 \text{ rotacije} / 5400 \text{ RPM} = 5.6 \text{ ms}$.

Poslednji korak je vreme transfera (ili brzina prenosa) koje je funkcija od veličine sektora, brzine rotacije i gustine zapisa (do oko 100 MB/s).

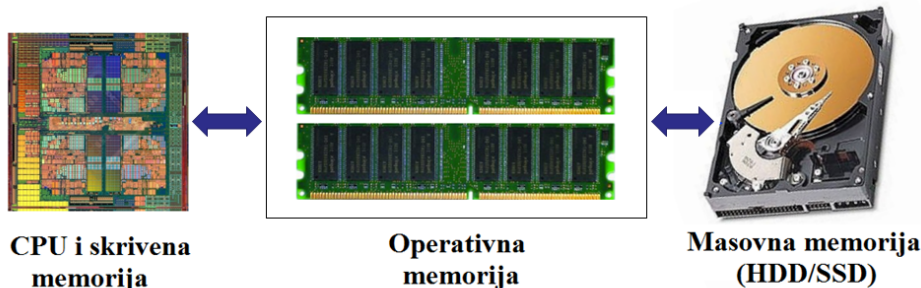
U današnje vreme poluprovodnički diskovi (SSD) polako preuzimaju ulogu koju su magnetni diskovi imali jer pružaju mnogo bolje performanse uz manji utrošak energije i nešto veću cenu. Vreme traženja je tipično oko 0.1 ms, a brzina prenosa ide i preko 2000 MB/s. Jedino na šta treba paziti je da dolazi do efekta *wear leveling* i potrebno je da kontroler pazi da se ravnomerno koriste svi blokovi.

4.5 Klasifikacija memorija

Memorije se mogu klasifikovati na osnovu različitih kriterijuma. Sa stanovišta prostorne lokalnosti, memorije se mogu klasifikovati u tri grupe: skrivena memorija (eng. *cache memory*), operativna (radna) i masovna memorija. Na slici 4.8 je prikazana ova klasifikacija kao i lokacije gde se koji tip nalazi.

Kod realizacije memorijskog podsistema mikroračunara često se pojavljaju suprotstavljeni zahtevi: veliki kapacitet, velika brzina i niska cena. Uzimajući u obzir da nije moguće direktno zadovoljiti ove suprotstavljene uslove, memorijski podsistem realizuje se iz tri dela: skrivene, operativne i masovne memorije.

Skrivena memorija je memorija najmanjeg kapaciteta u odnosu na operativnu i masovnu, ali je najbrža i najbliža procesoru i njen zadatak je da omogućiti da



Slika 4.8: Klasifikacija memorija

podatke ili instrukcije koji se često koriste procesor može brzo da dobije. Danas se ove memorije realizuju najčešće u tri nivoa (L1, L2 i L3) i nalaze se na čipu unutar mikroprocesora. Projektovanje skrivenih memorija, algoritmi upravljanja i hardverska podrška direktno su povezani sa principima lokalnosti obrađenim na početku poglavlja.

Operativna memorija realizuje se od poluprovodničkih memorijskih komponentata, koje mogu da se direktno spregnu sa mikroprocesorom. Poluprovodnička memorija mora da bude dovoljno brza da ne usporava rad mikroprocesora koji često pristupa operativnoj memoriji. Naravno, nepovoljne karakteristike poluprovodničke memorije su mali kapacitet, visoka cena i, u slučaju RAM-a, gubitak sadržaja kod prestanka napona napajanja.

Masovna memorija realizuje se od komponenata koje imaju veliki kapacitet i nisku cenu, kao što su na primer feromagnetni diskovi. Naravno, masovne memorije su po pravilu sporije i ne mogu da se neposredno spregnu sa mikroprocesorom. Da bi mikroračunar izvršio program koji je smešten na masovnoj memoriji, kao što je disk, mikroračunarski sistem prvo mora da taj program prebaci sa diska u operativnu memoriju i tek onda da ga izvrši.

4.5.1 Skrivena memorija

Podsistem skrivene memorije ima za cilj da ubrzava rad mikroračunarskog sistema. Upravljanje skrivenom memorijom obavlja se u toku mašinskog ciklusa pristupa memoriji, tako što se određuje da li se tražena memorijska lokacija nalazi u skrivenoj memoriji. Ako se nalazi, kažemo da se desio pogodak (eng. *Hit*), a ako

se ne nalazi onda se desio promašaj (eng. *Miss*). U slučaju pogotka mikroprocesor nastavlja započeti mašinski ciklus. U slučaju promašaja, potrebno je preduzeti mere koje će rešiti nastali promašaj tako što će traženi blok preuzeti iz operativne memorije (ili skrivene memorije nižeg nivoa). Iz ovoga zaključujemo da upravljanje skrivenom memorijom mora da bude veoma brzo i efikasno, a to pre svega može da se postigne ukoliko se primenjuju jednostavni i efikasni upravljački algoritmi. Ukoliko se ne postignu ovi ciljevi, može se desiti da mikroracunar sa skrivenom memorijom radi sporije od sistema bez skrivene memorije.

Današnji podsistemi skrivene memorije zasnivaju se na podeli (particiji) operativne memorije na blokove od r reči (na primer bajtova) i podeli skrivene memorije na pregratke kapaciteta takođe r reči, gde je r relativno malo, obično 4, 8, 16 ili 32. U slučaju promašaja kopije blokova prenose se iz operativne memorije u pregratke skrivene memorije. Ako to dovodi do zabune, umesto da kažemo da se u pregratku i nalazi kopija bloka j iz operativne memorije, često kažemo da se u pregratku i nalazi blok j .

Pretpostavimo da je kapacitet operativne memorije k_o reči (na primer bajtova), kapacitet skrivene memorije k_s reči i veličina (kapacitet) bloka (a time i pregratka) r reči. U tom slučaju su ukupan broj blokova b i broj pregradaka p :

$$b = k_o / r \quad (4.1)$$

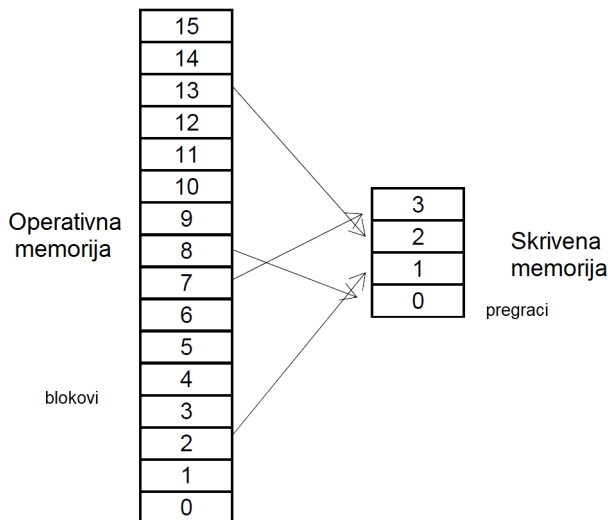
$$p = k_s / r \quad (4.2)$$

Blokovima i pregracima dodeljeni su redni brojevi, tako da su brojevi blokova (0, 1, 2, ..., $b - 1$) i redni brojevi pregradaka (0, 1, 2, ..., $p - 1$). Razlika između vrednosti rednih brojeva dva bloka naziva se udaljenost blokova. Kada se u svim pregracima nalaze kopije blokova iz operativne memorije, kažemo da je skrivena memorija popunjena.

Na slici 4.9 prikazan je pojednostavljen slučaj operativne memorije sa 16 blokova i skrivene memorije sa 4 pregratka. Vidi se da je blok 2 smešten u pregradak 1, blok 7 u pregradak 3, blok 8 u pregradak 0 i blok 13 u pregradak 2.

Sistem za upravljanje skrivenom memorijom treba da obavlja sledeće operacije:

1. Pretražiti skrivenu memoriju radi provere da li je u skrivenoj memoriji blok koji sadrži adresu koju je generisao mikroprocesor (provera da li je pogodak ili promašaj).
2. Odrediti koji pregradak u skrivenoj memoriji treba zameniti novim blokom (kada se desi promašaj i/ili je skrivena memorija popunjena).



Slika 4.9: Primer sprege operativne i skrivene memorije

3. Odrediti u koji pregradak treba smestiti blok kod prenosa bloka iz operativne u skrivenu memoriju (u slučaju promašaja). Algoritam koji određuje u koji pregradak se prenosi blok iz operativne memorije naziva se *funkcija preslikavanja*.
4. Obezbediti da su sadržaji istih blokova u operativnoj i skrivenoj memoriji jednaki. Ovaj uslov, koji se naziva *koherentnost* operativne i skrivene memorije, može biti narušen kod operacije upisa u skrivenu memoriju.

Složenost, performansa i cena podsistema za upravljanje skrivenom memorijom u najvećoj meri zavise od funkcije preslikavanja. Koriste se tri osnovne funkcije preslikavanja: direktno, asocijativno i kombinovano preslikavanje.

Direktno preslikavanje

Kod direktnog preslikavanja su relacije blok-pregradak unapred određene i fiksne (npr. Blokovi 0,4,8,12 u pregradak 0; blokovi 1,5,8,13 u pregradak 1 itd.). Prednost ovog načina je u tome što je vrlo jednostavan za realizaciju, ali su performanse slabe jer se može desiti da određeni pregraci budu dugo vremena neiskorišćeni, a da je zbog fiksnih relacija blok-pregradak potrebno često kopiranje blokova iz

operativne memorije.

Asocijativno preslikavanje

Kod asocijativnog je rešena glavna mana direktnog preslikavanja, a to je da se blokovi ne mogu kopirati u bilo koji pregradak. Drugim rečima, kod asocijativnog preslikavanja kopije bilo kojih p blokova iz operativne memorije mogu u istom trenutku da budu u skrivenoj memoriji. To znači da ako mikroprocesor pristupa bilo kojim od p blokova, po bilo kom redosledu, svi blokovi mogu u istom trenutku biti u skrivenoj memoriji, pa je procenat pogodaka najveći. Samim tim je performansa sistema najbolja, ali je i složenost implementacije najveća.

Kombinovano preslikavanje

Kombinovano preslikavanje (eng. *Set Associative*) predstavlja kompromis između direktnog i asocijativnog preslikavanja i postiže bolje performanse od direktnog preslikavanja uz manju složenost u odnosu na asocijativno preslikavanje. Kombinovano preslikavanje sastoji se u tome da se disjunktni podskupovi blokova koji su određeni prema direktnom preslikavanju, mogu smestiti u bilo koji od s pregradaka, $1 < s << p$.

Poboljšanje performanse postiže se time što čak i ako mikroprocesor naizmenično pristupa s blokovima iz istog podskupa, ovi blokovi mogu biti istovremeno u skrivenoj memoriji, što dovodi do većeg procenta pogodaka. Sa druge strane, složenost implementacije je manja nego kod asocijativnog preslikavanja, jer kod provere da li je blok u skrivenoj memoriji, umesto p pregradaka, potrebno je pretražiti s pregradaka, pri čemu je $r << p$.

Treba primetiti da se za $s = 1$ kombinovano preslikavanje svodi na direktno preslikavanje, a za $r = p$ svodi se na asocijativno preslikavanje. Izborom s između vrednosti 1 i p može se podešavati performansa i složenost implementacije skrivene memorije. Ako je s malo, dobija se slabija performansa i jednostavnija složenost, a veliko s daje bolje performanse ali veću složenost implementacije. Najveće povećanje performanse dobija se ako se umesto $s = 1$ uzme $s = 2$. Nešto manje povećanje performanse dobija se kada se s poveća sa 2 na 4, međutim, dalje povećanje s daje sve manje poboljšanje performansi.

Zbog dobrog odnosa između performansi i složenosti implementacije, kombinovano preslikavanje se koristi kod većine komercijalnih mikroprocesora sa relativno malim vrednostima za s (npr. $s = 2, 4, 8$ ili 16).

4.5.2 Operativna memorija

Sve informacije u mikroračunarskom sistemu predstavljaju se nizovima binarnih cifara, 0 i 1. Najmanja količina informacije predstavljena je jednom binarnom cifrom i naziva se bit (akronim od engleskog naziva *Binary Digit*). Zadatak memorije je da pamti binarne informacije i da ih, na zahtev, stavi na raspolaganje nekoj drugoj jedinici računara.

Osnovni sastavni deo memorije je memorijska ćelija, koja može biti u jednom od dva različita stanja kodirana nulom (0) i jedinicom (1). Dakle, memorijska ćelija može da pamti informaciju od jednog bita. Realizacija memorijskih ćelija zavisi od tehnologije izrade, a osnovne osobine ćelija su mogućnost prevođenja ćelija u stanje 0 ili 1, održavanje i kasnijem čitanju stanja memorijskih ćelija.

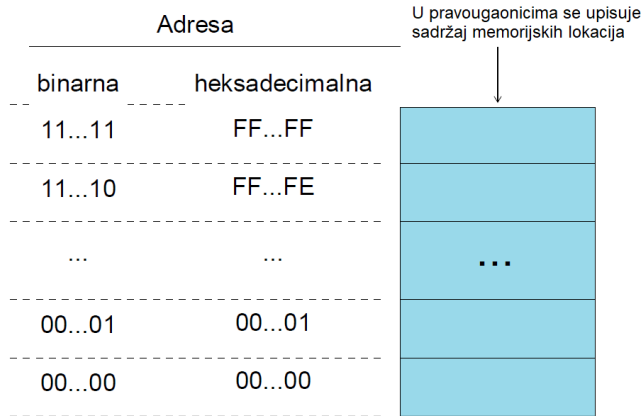
Memorija ima definisane dve operacije: upis i čitanje. Operacijom upisa binarna informacija smešta se u memoriju tako što se memorijske ćelije prevode u stanja koja predstavljaju binarne cifre ulazne informacije. Podrazumeva se da memorijske ćelije posle operacije upisa ostaju u stanjima u koja su prevedena operacijom upisa, što znači da 'pamte' upisanu informaciju.

Operacijom čitanja detektuju se stanja memorijskih ćelija i time 'čita' informacija koja je prethodno upisana u memoriju. Termin 'pristup' korisi se za operaciju čitanja ili upisa.

Radi veće brzine prenosa podataka memorijske ćelije su grupisane tako da mikroprocesor pristupa paralelno svim ćelijama u jednoj grupi. Grupa memorijskih ćelija kojima se paralelno pristupa naziva se **memorijska lokacija** ili memorijski registar. U memorijske lokacije smeštaju se informacije od više bita, koje se nazivaju memorijske reči. Dužina memorijske lokacije je broj memorijskih ćelija u lokaciji i jednak je broju bita memorijske reči koja može da se smesti u lokaciju. Reč dužine 8 bita naziva se bajt.

Memorija je organizovana u obliku linearnog niza memorijskih lokacija. Svakoj memorijskoj lokaciji pridružen je jedinstven binarni broj koji se naziva **adresa**. Organizacija memorije određena je brojem memorijskih lokacija, pridruženim adresama i brojem memorijskih ćelija u svakoj lokaciji. Slika 4.10 prikazuje grafički simbol za memoriju. Svaka memorijska lokacija predstavljena je pravougaonikom, pored koga se piše adresa lokacije, a u pravougaonik upisuje *sadržaj* memorijske lokacije. Veoma je važno u svakom trenutku jasno razlikovati šta predstavlja adresu neke lokacije, a šta njen sadržaj. Ovo je posebno značajno kod složenijih tipova adresiranja radi pravilnog razumevanja adresiranja podataka.

Organizacija memorije iskazuje se uređenim parom (l, n) , gde je l broj memorijskih lokacija, a n broj memorijskih ćelija u svakoj lokaciji. Broj l je iz skupa brojeva 2^m gde je m broj adresnih signala memorije, pa je uobičajeno da se or-



Slika 4.10: Simbol memorije sa adresama

organizacija memorije predstavlja u obliku ($2m \times n$). Radi jednostavnijeg pisanja uvedene su sledeće jedinice:

$$\begin{aligned}
 1 \text{ k (kilo)} &= 2^{10} \\
 1 \text{ M (mega)} &= 2^{20} = 2^{10} \text{ k} \\
 1 \text{ G (giga)} &= 2^{30} = 2^{10} \text{ M} \\
 1 \text{ T (tera)} &= 2^{40} = 2^{10} \text{ G}
 \end{aligned}$$

Primeri organizacije memorije su ($1k \times 4$), ($4k \times 8$), ($16k \times 8$), ($256k \times 1$), ($1M \times 1$) itd.

Kapacitet memorije je količina informacije izražene u bitima koja može da se upiše u memoriju. Kapacitet se jednostavno izračunava tako što se broj memorijskih lokacija pomnoži brojem ćelija u jednoj lokaciji. U tabeli 4.1 dati su neki primeri organizacije memorije i kapaciteti izraženi u bitima i bajtovima.

Prilikom pristupa neophodno je memoriji saopštiti adresu lokacije kojoj se pristupa i vrstu operacije koju memorija treba da izvrši. Zato memorija poseduje linije za prenos adresnih signala, za prenos memorijskih reči i za upravljačke signale. Dva jednostavna primera modela memorije prikazani su na slici 4.11.

Adresne i upravljačke linije imaju smer prenosa signala prema memoriji. Linije za prenos signala podataka u opštem slučaju su dvosmerne tako da kod operacije upisa signali se prenose prema memoriji, a kod operacije čitanja imaju suprotan

Tabela 4.1: Primeri organizacije memorije

Organizacija	Kapacitet u bitima	Kapacitet u bajtovima
1k x 4	4k	512
4k x 8	32k	4k
16k x 8	128k	16k
256k x 1	256k	32k
1M x 1	1M	128k



Slika 4.11: Model memorije sa signalima R i W (levo) i signalima CS i R/W (desno)

smer.

Treba primetiti da upravljački signali mogu biti aktivni kada su na logičkoj 0 ili na logičkoj 1. Aktivna logička 0 obeležava se tako što se iznad naziva upravljačkog signala stavi crtica.

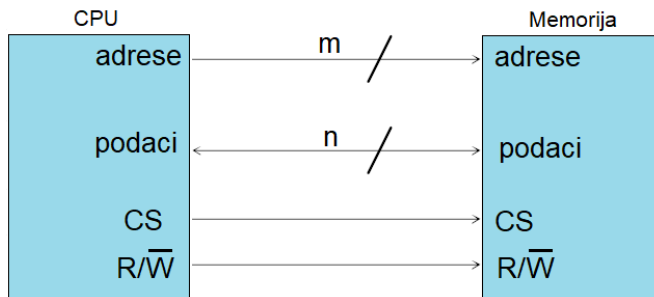
Tabela 4.2: Upravljački signali memorije

R	W	CS	R/\overline{W}	Operacija
0	0	0	X	nema operacije
0	1	1	0	upis
1	0	1	1	čitanje
1	1			nije dozvoljeno

U praksi se često sreću memorije koje umesto upravljačkih signala R i W imaju upravljačke signale CS i R/\overline{W} . Upravljački signal CS aktivira memoriju, a signal R/\overline{W} određuje operaciju čitanja (ako je na logičkoj 1) ili upisa (ako je na logičkoj 0). Iz tabele 4.2 se vidi objašnjenje operacija upravljačkih signala i njihovo značenje. Pored toga može se primetiti da signali R i W imaju zabranjenu kombinaciju ($R=W=1$) dok signali CS i R/\overline{W} nemaju zabranjene kombinacije.

4.6 Sprega memorije i mikroprocesora

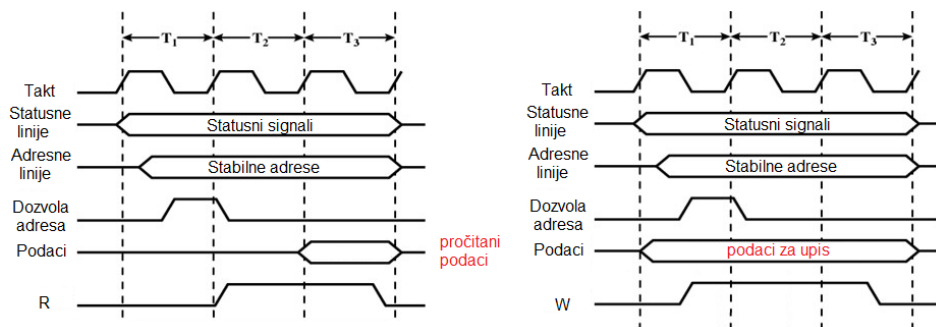
Kao što je već ranije rečeno, mikroprocesor i poluprovodnička memorija su međusobno kompatibilni u smislu da se mogu neposredno međusobno povezati. Neposredno veza znači da se adresni signali mikroprocesora povezuju sa adresnim signalima memorije, signali podataka mikroprocesora sa signalima podataka memorije i upravljački signali mikroprocesora sa odgovarajućim signalima memorije. Na slici 4.12 prikazana je neposredna veza mikroprocesora i memorije u slučaju da imaju isti broj adresnih signala (m), signala za podatke (n) i da su upravljački signali mikroprocesora i memorije CS i R/\overline{W} .



Slika 4.12: Neposredna veza mikroprocesora i memorije

Sprega mikroprocesora i memorije može biti sinhrona ili asinhrona. Kod sinhronne sprege podrazumeva se da je memorija dovoljno brza da operacije čitanja i upisa obavi u predviđenom vremenskom intervalu. Na slici 4.13 prikazuje primer vremenskih dijagrama ciklusa čitanja (levi dijagrami) i upisa (desni dijagrami) kod sinhronne sprege. Ciklus traje ukupno tri periode sinhronizacionog signala (signala takta). Na početku periode T1 mikroprocesor stavi na adresne linije signale adrese memorijske lokacije iz koje treba pročitati podatak, a zatim, kad su adresni signali dovoljno stabilni (što signalizira 'Dozvola adresa') aktivira signal čitanja Read.

U trenutku kada primi aktivan signal čitanja Read, memorija dekoduje adresne signale, pristupi adresiranoj memorijskog lokaciji i sadržaj te lokacije prosledi na magistralu podataka. Svakako da memorijskim kolima treba izvesno vreme da obavi sve ove operacije, tako da se sadržaj adresirane memorijske lokacije pojavljuje na linijama za podatke sa kašnjenjem. Mikroprocesor ostavlja memoriji tačno određeno vreme, u navedenom primeru to je do opadajuće ivice impulsa T3, kada mikroprocesor signale sa magistrale podataka učitava u interni registar. Posle toga adresni i upravljački signali (u ovom slučaju signal Read) prevode se u neaktivno



Slika 4.13: Primer vremenskih dijagrama kod sinhronne sprege mikroprocesora i memorije

stanje, čime se završava ciklus čitanja.

Ciklus upisa počinje tako što mikroprocesor stavlja na adresne linije adresne signale, a nakon toga na linije za podatke signale podatka koji treba upisati u adresiranu memorijsku lokaciju. Posle toga aktivira se signal Write za upis. Po isteku dovoljno dugog vremenskog intervala, u ovom primeru do opadajuće ivice impulsa T_3 memorija treba da podatak sa magistrale upiše u adresiranu memorijsku lokaciju. Posle tog trenutka, svi signali se vraćaju u početno stanje i sve je spremno za sledeći ciklus.

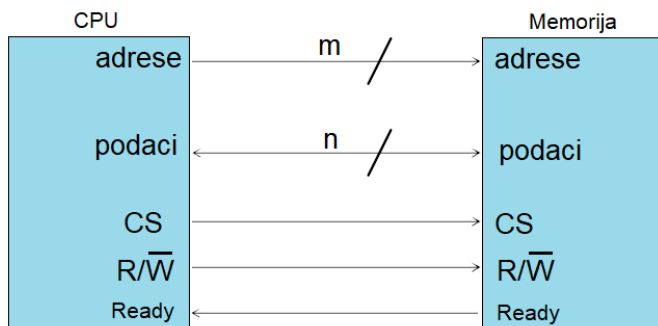
Zavisno od konkretnog mikroprocesora i memorije ovi vremenski dijagrami se mogu manje ili više razlikovati. Stoga je ovo samo jedan od primera vremenskih dijagrama sinhronne sprege mikroprocesora i memorije.

Sinhrona sprega mikroprocesora i memorije, ima sledeće karakteristike:

- Mikroprocesor nema povratnu informaciju da je memorija završila zahtevanu operaciju. Prema tome, mikroprocesor mora da memoriji ostavi dovoljno vremena da obavi operaciju čitanja ili upisa i to u najnepovoljnijem slučaju.
- Mikroprocesor može da radi onoliko brzo koliko dozvoljava najsporija memorijska jedinica u sistemu.
- Sinhrona sprega je jednostavna.
- Ako memorijski modul nije dovoljno brz, onda treba smanjiti frekvenciju takta mikroprocesora ili preći na korišćenje brže memorije.

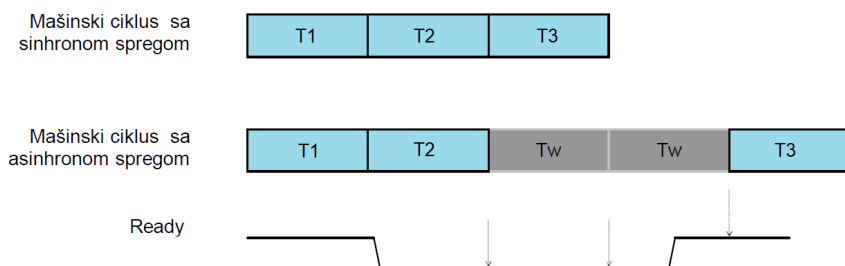
Kod asinhronne sprege memorija posebnim signalom (eng. *Ready*) saopštava mikroprocesoru da je završila operaciju čitanja odnosno upisa što je prikazano na

slici 4.14. Mikroprocesor nadgleda signal *Ready* i produžava mašinski ciklus sve do trenutka kada memorija završi zahtevanu operaciju.



Slika 4.14: Asinhrona sprega mikroprocesora i memorije

Na slici 4.15 se vidi mašinski ciklus sa asinhronom spregom u poređenju sa istim ciklusom kod sinhronne sprege. Radi jednostavnosti, ovde su pravougaonici sa upisanim oznakom perioda signala takta predstavljeni svi signali koji su navedeni u prethodnim vremenskim dijagramima. Tako na primer mašinski ciklus kod sinhronne sprege traje tri perioda takta označena sa T2, T2 i T3.



Slika 4.15: Primer asinhronog ciklusa sa umetanjem perioda čekanja T_w

Kod asinhronne sprege, na početku mašinskog ciklusa, na primer na kraju periode takta T1, memorija prevede signal *Ready* na logičku nulu i time obaveštava mikroprocesor da je započela zahtevanu operaciju, ali da je još nije završila. Na kraju periode T2 mikroprocesor proverava signal *Ready* i ako je na logičkoj nuli, produžava mašinski ciklus umetanjem periode čekanja T_w . Na kraju periode T_w ,

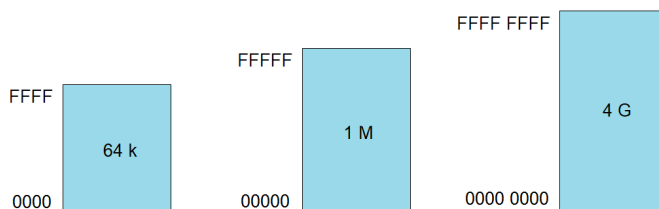
mikroprocesor ponovo proverava *Ready*, ako je i dalje na logičkoj 0 onda umetne još jednu periodu čekanja *T_w*. Ovaj postupak se ponavlja dok memorija ne završi zahtevanu operaciju i obavesti mikroprocesor tako što vrati signal *Ready* na logičku 1. Kada detektuje visoki nivo signala *Ready*, mikroprocesor završi mašinski ciklus tako što obavi mikrooperacije predviđene periodom T3.

Karakteristike asinhronne sprege su:

- Mikroprocesor dobija obaveštenje da je memorija završila predviđene operacije, pa je sprega mikroprocesora i memorije pouzdanija.
- Mikroprocesor može da radi sa memorijama različite brzine i prilagođava trajanje mašinskog ciklusa brzini svake memorijske jedinice.
- Asinhrona sprega je složenija u odnosu na sinhronu spregu.

4.7 Memorijski adresni prostor

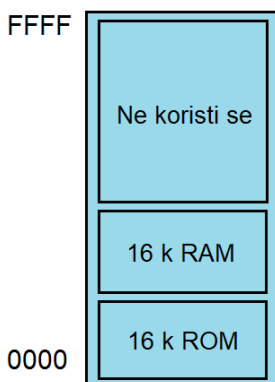
Skup svih memorijskih adresa koje mikroprocesor može da generiše naziva se memorijski adresni prostor. Ovaj prostor grafički predstavljamo pravougaonikom, slično kao i memoriju, a sa strane pravougaonika pišemo adrese (obično koristeći heksadecimalni brojevni sistem).



Slika 4.16: Primeri memorijskog adresnog prostora mikroprocesora

Na slici 4.16 prikazuje memorijske adresne prostore mikroprocesora sa 16, 20 i 32 adresna signala. Običaj je da se memorijski adresni prostor većeg kapaciteta predstavlja pravougaonikom većih dimenzija (ali naravno dimenzije pravougaonika i kapacitet memorije nisu međusobno proporcionalni!). U grafičkom predstavljanju često se izostavlja broj signala magistrale podataka jer se obično podrazumeva da je u pitanju 8 signala. Ukoliko je broj podataka veći od 8 (npr. 16, 32, 64 ili 128) tada je neophodno odrediti redosled upisivanja u bajtovima. Ova tema će biti obrađena u narednom poglavlju.

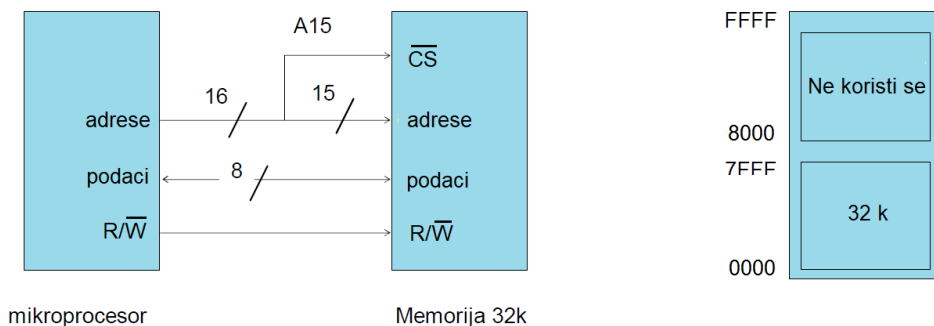
Kod grafičkog predstavljanja memorijskog adresnog prostora korisno je označiti delove adresnog prostora koji zauzimaju fizičke memorijske jedinice i delove memorijskog prostora koji se ne koristi. Slika 4.17 prikazuje memorijski adresni prostor mikroprocesora sa 16 adresnih signala (64 k) u mikroračunarskom sistemu sa 16 k memorije tipa ROM, koji se nalazi na dnu memorijskog adresnog prostora (adrese od 0000 do 3FFF) i 16 k memorije tipa RAM, koja se nalazi u delu memorijskog prostora koji je 'iznad' ROM-a (adrese od 4000 do 7FFF). Preostali deo memorije, od adrese 8000 (heksadecimalno) do FFFF (heksadecimalno) ne koristi se.



Slika 4.17: Primer memorijskog adresnog prostora sa fizičkom memorijom i delom koji se ne koristi

Uzmimo, na primer, mikroprocesor sa 16 adresnih signala i 8 signala za podatke koji je spregnut sa memorijom 32k x 8, slika 4.18, levo. Memorija organizacije 32k x 8, ima 15 adresnih linija i 8 linija za podatke i upravljačke signale \overline{CS} i R/\overline{W} . Memorija se aktivira kad je signal \overline{CS} na logičkoj 0. Linije za podatke mikroprocesora direktno su vezane na linije za podatke memorije, a signal R/\overline{W} (pretpostavimo da mikroprocesor ima takav upravljački signal) vezan je na signal iste namene na memoriji.

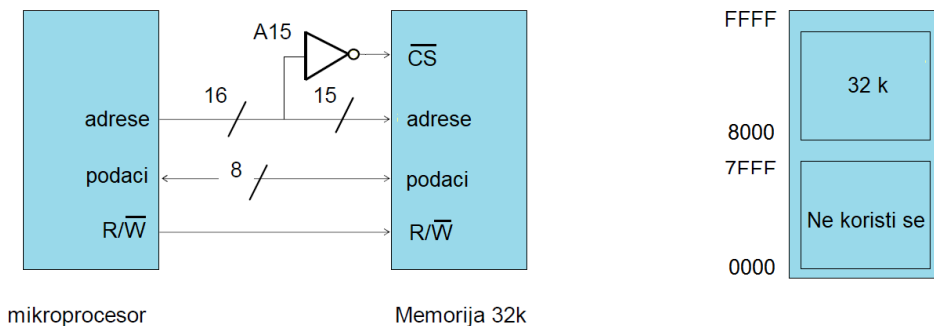
Vezivanje linija adresne magistrale nije jednoznačno, zato što mikroprocesor ima 16, a memorija 15 adresnih linija. U ovakvim slučajevima manje značajne linije adresne magistrale mikroprocesora vezuju se na odgovarajuće linije memorije, pri čemu više značajne adresne linije mikroprocesora nemaju linije na memoriji na koje mogu da se vežu. U datom primeru, 15 linija mikroprocesora, od A0 do A14 vežu se na linije memorije od A0 do A14. Linija A15 mikroprocesora ostaje slobodna i može se koristiti na tri načina. Na slici 4.18 prikazan je slučaj kada je linija A15



Slika 4.18: Primer sprege mikroprocesora i memorije (levo) i odgovarajući memorijski adresni prostor (desno)

mikroprocesora dovedena na signal \overline{CS} memorije. U ovom slučaju, kad je $A15=0$ memorija je aktivirana, a kada je $A15=1$, memorija nije aktivirana. Očigledno je u ovom slučaju memorijski adresni prostor kao na slici 4.18 desno: donjih 32 k adresnog prostora zauzima fizička memorija, a ne koristi se gornjih 32 k adresnog prostora.

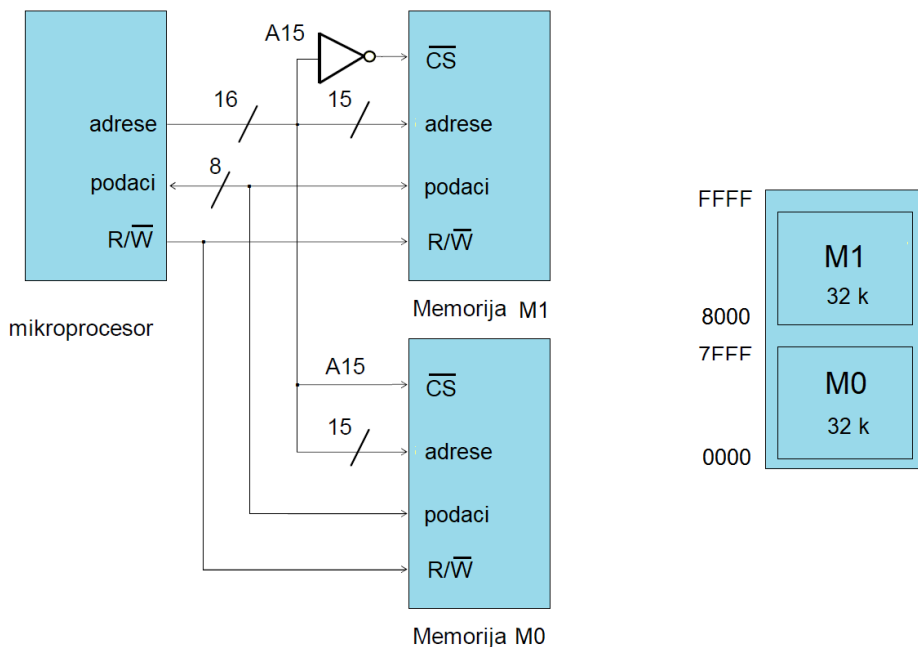
Ako se umesto signala A15 na ulaz \overline{CS} od memorije dovede invertovani signal A15, onda će 32 kB fizičke memorije biti u gornjoj polovini adresnog prostora, a donja polovina se neće koristiti što je prikazano na slici 4.19.



Slika 4.19: Modifikovana sprega mikroprocesora i memorije (levo) i odgovarajući memorijski adresni prostor (desno)

Kombinacijom prethodnih slučajeva mogu se vezati dva memorijska modula od

po 32 k tako da jedan bude u donjoj, a drugi u gornjoj polovini adresnog prostora (videti sliku 4.20).



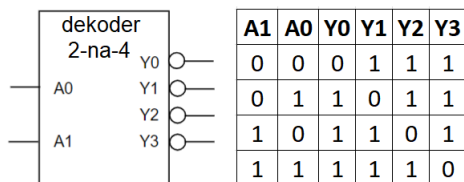
Slika 4.20: Sprega mikroprocesora sa dva memorijska modula

4.8 Adresni dekoderi

Dekoder spada u komponente kombinacionih digitalnih mreže. Primer dekodera 2-na-4 prikazan je na slici 4.21. Ova komponenta se može koristiti u slučajevima kada je potrebno da se vrši izbor memorija upotrebom selekcionih signala CS (*chip select*).

Neka mikroprocesor ima m adresnih linija i neka se spreže sa memorijskim modulom (ili modulima) koji imaju k adresnih linija. Iz prethodnih primera vidi se da se adresne linije mikroprocesora dele na dve grupe:

- Prva grupa od k adresnih linija neposredno se spaja na k ulaznih adresnih linija memorijskih modula tako što se signali $A0, A1, \dots, Ak-1$ mikroprocesora



Slika 4.21: Dekoder 2-na-4

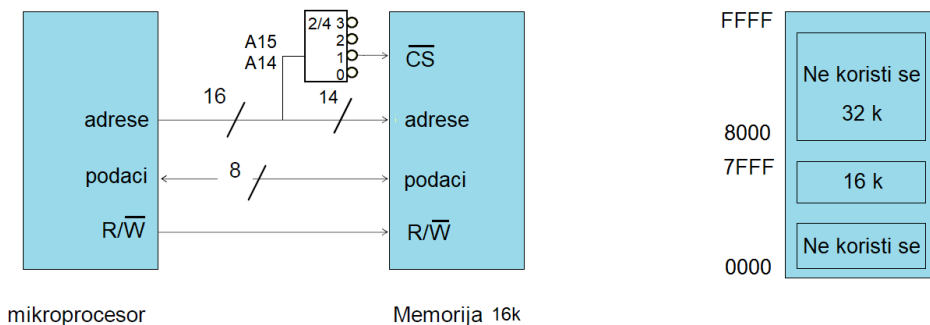
spajaju na signale $A0, A1, \dots, Ak-1$ memorije.

- Preostalih $(m-k)$ adresnih linija koriste se za adresne dekodere.

Adresni dekoderi su kombinacione logičke mreže koje na osnovu ulaznih adresnih signala generišu CS signale za memorijske module. Ako je p broj ulaznih signala adresnog dekodera, onda razlikujemo sledeća tri slučaja:

1. Ako je $p = m - k$, onda se radi potpuno adresno dekodiranje.
2. Ako je $0 < p < m - k$, onda je dekodiranje adresa delimično.
3. Ako je $p = 0$, nema dekodiranja adresa.

Uzmimo na primer mikroprocesor sa 16 adresnih signala i 8 signala za podatke i memorijski modul od 16 k. Pošto memorijski modul ima 14 adresnih signala, onda preostaju dva signala, A14 i A15, koji se dovode na ulaz adresnog dekodera (videti sliku 4.22).



Slika 4.22: Primena adresnog dekodera

Neka su izlazi dekodera $\overline{CS3}$, $\overline{CS2}$, $\overline{CS1}$, $\overline{CS0}$ i neka je funkcija dekodera definisana sledećom tablicom.

A15	A14	$\overline{CS3}$	$\overline{CS2}$	$\overline{CS1}$	$\overline{CS0}$
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	1

Mesto memorijskog modula u adresnom prostoru zavisi od izlaza dekodera koji se poveže na ulaz \overline{CS} za selekciju memorijskog modula. U primeru sa slike 4.22 izlaz $\overline{CS1}$ koristi se za aktiviranje memorije, pa je zato memorijski modul smešten u adresnom prostoru koji je prikazan na slici 4.22 desno. U sledećoj tabeli date su memorijske adrese memorijskog modula sa slike 4.22 u zavisnosti od izlaza adresnog dekodera koji se koristi kao signal selekcije \overline{CS} . U sistem mogu da se po potrebi stave dva, tri ili četiri memorijska modula sa proizvoljnim korišćenjem izlaza adresnog dekodera za selekciju memorijskih modula.

Signal adresnog dekodera	Opseg adresa memorijskog modula
$\overline{CS3}$	C000 do FFFF
$\overline{CS2}$	8000 do BFFF
$\overline{CS1}$	4000 do 7FFF
$\overline{CS0}$	0000 do 3FFF

4.9 Zadaci

Klasičan problem kod projektovanja mikroprocesorskog sistema sa memorijom nastaje kada projektant nema poluprovodničke memorijske module koji odgovaraju specifikaciji memorijske jedinice. Razlikujemo sledeća tri slučaja:

1. Memorijski moduli imaju dovoljan broj bita u memorijskim lokacijama, ali nemaju dovoljan broj memorijskih lokacija. Na primer, zahteva se memorijska jedinica od 128k x 8, a na raspolaganju su memorijski moduli sa organizacijom 32k x 8.
2. Memorijski moduli imaju dovoljan broj lokacija, ali nemaju dovoljan broj bita u memorijskim lokacijama. Na primer, zahteva se memorijska jedinica od 32k x 16, a na raspolaganju su memorijski moduli 32k x 8.

3. Kombinacija prethodna dva slučaja, odnosno raspoloživi memorijski moduli nemaju ni dovoljan broj lokacija, ni dovoljan broj bita u lokacijama.

Projektovanje memorijskog podsistema svodi se na projektovanje adresnog dekodera i povezivanje zadatih memorijskih modula u memorijsku jedinicu sa zadatom specifikacijom.

Zadatak 1

Projektovati mikroprocesorski sistem koji ima sledeće karakteristike:

- CPU ima 16 adresnih linija i 8 linija za podatke
- na raspolaganju su dve memorije: 16k x 8 i 32k x 8 i druge komponente (npr. dekoderi)

Veća memorija treba da bude u gornjoj polovini adresnog prostora, a manja u gornjoj polovini donje polovine adresnog prostora.

Pored nacrtane šeme sistema označiti i opsege adresa koje memorije zauzimaju u adresnom prostoru.

Rešenje:

Za selekciju veće memorije korišćemo način aktiviranja \overline{CS} signala na memoriji kao na slici 4.19, a za selekciju manje memorije kao na slici 4.22.

Opseg adresa veće memorije je 8000-FFFF, a manje je 4000-7FFF.

Zadatak 2

Projektovati memorijsku jedinicu organizacije 128k x 8 na dnu adresnog prostora mikroprocesora sa 20 adresnih signala, ako su na raspolaganju memorijski moduli organizacije 32k x 8.

Rešenje:

Očigledno je da nam za rešenje ovog problema trebaju 4 memorijska modula, zato što će 4 x (32k x 8) dati memorijsku jedinicu 128k x 8. Iz organizacije memorijskih modula koji su na raspolaganju, vidi se da oni imaju 15 adresnih linija i 8 linija za podatke. Prema tome, od ukupno 20 adresnih signala mikroprocesora, donjih 15 signala neposredno se vezuju na odgovarajućih 15 adresnih signala memorijskih modula.

Preostalih 5 adresnih signala koristi se za adresni dekodер. Treba primetiti da pošto se koriste 4 memorijska modula, dovoljna su 2 adresna signala, A16 i A15,

za generisanje signala za selekciju ($\overline{CS3}$, $\overline{CS2}$, $\overline{CS1}$ i $\overline{CS0}$). Preostala tri signala A19, A18 i A17 koriste se za generisanje signala dozvole rada adresnog dekodera. Prema tome, može se koristiti adresni dekodер 2-na-4, ali sa signalom dozvole koji se aktivira kada je $A19 = A18 = A17 = 0$.

4.10 Pitanja

1. Objasniti princip prostorne lokalnosti kod memorija.
2. Objasniti princip vremenske lokalnosti kod memorija.
3. Objasniti šta znači memorijska hijerarhija i navesti primer.
4. Objasniti podelu memorija sa stanovišta način pristupa memorijskim lokacijama.
5. Objasniti podelu memorija sa stanovišta trajnosti upisanih informacija.
6. Šta je to $_ROM$?
7. Objasniti kako se mogu realizovati memorijske ćelije.
8. Koje tehnologije su dominantne pri izradi memorija?
9. Objasniti razlike između SRAM i DRAM memorija.
10. Zbog čega danas SSD sve više zamenjuju magnetne diskove i na šta treba paziti?
11. Kako se memorije klasifikuju sa stanovišta prostorne lokalnosti? Objasniti ulogu svake od njih.
12. Objasniti svrhu skrivene memorije i pojmove *miss* i *hit*.
13. Objasniti pojmove blok, pregradak, kapacitet skrivene memorije, kapacitet operativne memorije i objasniti funkcionisanje skrivene memorije na primeru $b = 16$ i $p = 4$.
14. Šta sve treba da obezbedi podsistem za upravljanje skrivenom memorijom?
15. Objasniti sva tri tipa preslikavanja kod realizacija skrivenih memorija.
16. Sinhrona veza procesora i memorije.

17. Nacrtati i objasniti vremenske dijagrame čitanja memorije.
18. Nacrtati i objasniti vremenske dijagrame upisa u memoriju.
19. Asinhrona sprega procesora i memorije.
20. Objasniti različite mogućnosti adresnog dekodovanja na primeru CPU sa m adresnih linija, memorije sa k adresnih ulaza i dekodera sa p adresnih ulaza.
21. Projektovati mikroprocesorski sistem sa memorijom čije željene karakteristike (broj linija za podatke i adresnih linija, veličine memorija, adresni prostor i sl.) su zadate daljim tekstom zadatka.

Bibliografija

- [1] Arthur W. Burks, Herman H. Goldstine, and John von Neumann, "Preliminary discussion of the logical design of an electronic computing instrument", The Institute of Advanced Study, Princeton, USA, 1946-47.
- [2] J. Biggs, J. Myers, J. Kufel et al. A natively flexible 32-bit Arm microprocessor, *Nature* 595, pp. 532–536, 2021.
- [3] F. Arute, K. Arya, R. Babbush et al. Quantum supremacy using a programmable superconducting processor. *Nature* 574, pp. 505–510, 2019.
- [4] J. Hochstetter, R. Zhu, A. Loeffler et al. Avalanches and edge-of-chaos learning in neuromorphic nanowire networks. *Nat Commun* 12, 4008, 2021.
- [5] D. A. Patterson, J. L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, Fourth Edition, Morgan Kaufmann, 2011.
- [6] M. Abd-El-Barr, H. El-Rewini, *Fundamentals Of Computer Organization And Architecture*, John Wiley and Sons, Inc., 2005.
- [7] R. S. Sandige, M. L. Sandige, *Fundamentals of Digital and Computer Design with VHDL*, McGraw Hill, 2012.
- [8] M. M. Mano, C. R. Kime, T. Martin, *Logic and Computer Design Fundamentals*, Fifth edition, Pearson, 2015.
- [9] J.L. Gustafson, Moore's Law. In: D. Padua D. (eds) *Encyclopedia of Parallel Computing*. Springer, 2011.
- [10] S.A. McKee, R.W. Wisniewski, Memory Wall. In: D. Padua (eds) *Encyclopedia of Parallel Computing*. Springer, 2011.

- [11] P. Bose, Power Wall. In: D. Padua (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [12] A. Tsakyridis, T. Alexoudi, A. Miliou, N. Pleros, and C. Vagionas, "10 Gb/s optical random access memory (RAM) cell," Opt. Lett. 44, pp. 1821-1824, 2019.
- [13] W.W. Royce, "Managing the Development of Large Software Systems: Concepts and Techniques", IEEE Wescon Proc., Aug. 1970.
- [14] C.G. Bell, A. Newell, Computer structures: Readings and Examples, McGraw-Hill, 1971.
- [15] E.O. Hwang, Digital Logic and Microprocessor Design With VHDL with Interfacing, Cengage Learning, Second Edition, 2017.
- [16] S. Brown, Z. Vranesic, Fundamentals of Digital Logic with VHDL Design, McGraw Hill, 2009.
- [17] D. L. Perry, VHDL: Programming by Example, Fourth edition, McGraw Hill, 2002.
- [18] L. Null, J. Lobur, Essentials of Computer Organization and Architecture, Jones and Bartlet Learning, 3rd Ed., 2010.
- [19] I. Mezei "Evolution of an educational microprocessor", Computer Applications in Engineering Education, 28(5), Wiley, pp. 1265-1277, 2020.
- [20] D. Patterson, J.L. Hennessy, Computer Organization and Design RISC-V Edition: The Hardware Software Interface, The Morgan Kaufmann Series in Computer Architecture and Design, 1st Edition, 2017.
- [21] D. Patterson, J.L. Hennessy, Computer Organization and Design RISC-V Edition: The Hardware Software Interface, The Morgan Kaufmann Series in Computer Architecture and Design, 2nd Edition, 2021.
- [22] S. L. Harris, D. Harris, Digital Design and Computer Architecture, RISC-V Edition, Morgan Kaufman, 1st edition, 2021.
- [23] D. Patterson, "Reduced Instruction Set Computers Then and Now" in Computer, vol. 50, no. 12, pp. 10-12, 2017. <https://doi.ieeecomputersociety.org/10.1109/MC.2017.4451206>
- [24] S. Harris, D. Hariss, Digital Design and Computer Architecture, Arm edition, Morgan Kaufmann, 2015.

- [25] D. A. Patterson, J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, ARM edition, The Morgan Kaufmann Series in Computer Architecture and Design, 2016.
- [26] D. Kushner, "The making of arduino", IEEE Spectrum 26, 2011.
- [27] M. Banzi, "How Arduino is open-sourcing imagination", TEDtalk, Scotland, 2012. <https://www.youtube.com/watch?v=UoBUXOOdLXY> (datum pristupa: 31.01.2022.)
- [28] Sparkfun, "Arduino shields v2", <https://learn.sparkfun.com/tutorials/arduino-shields-v2>, (datum pristupa: 01.02.2022.)
- [29] Arduino Uno Rev3 schematic, https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf (datum pristupa: 02.02.2022.)
- [30] Arduino software, <https://www.arduino.cc/en/Main/Software> (datum pristupa: 02.02.2022.)
- [31] 74HC595 datasheet, Philips, 1998, <https://www.arduino.cc/en/uploads/Tutorial/595datasheet.pdf> (datum pristupa: 03.03.2022.)
- [32] <https://www.ti.com/lit/ds/symlink/sn74hc165.pdf> (datum pristupa: 03.03.2022.)
- [33] ATmega328 datasheet, Atmel, 2015, https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf (datum pristupa: 15.03.2022.)

