UNIVERZITET
METROPOLITAN
BEOGRAD

CS120 - ORGANIZACIJA RAČUNARA

Sloj digitalne logike - Sekvencijalna kola

Lekcija 04

PRIRUČNIK ZA STUDENTE

CS120 - ORGANIZACIJA RAČUNARA

Lekcija 04

SLOJ DIGITALNE LOGIKE - SEKVENCIJALNA KOLA

UVOD

Uvod u lekciju #4

U ovoj lekciji bilo je najpre biće predstavljena razlika između kombinatornih kola iz prethodne lekcije, i sekvencijalnih kola, čiji se trenutni izlaz računa ne samo u zavisnosti od trenutnih stanja na ulazu, već i od prethodnih izlaza.

Kao osnovno sekvencijalno kolo predstavljeno je SR-latch kolo, kao i D-flip-flip. Ova kola su pogodna za skladištenje informacija, pa se koriste za realizaciju memorijskih i registarskih elemenata.

Upravo zbog toga biće predstavljeni pomerački i brojački registri, kao i registri po modulu M.

Nakon toga, sledi objekat učenja o samoj RAM memoriji, i njene strukture i podele na SRAM i DRAM.
Konačno biće reči o stack i queue memoriji, koje se uveliko koriste u različitim realizacijama.

Na vežbama pokazana su kola koder/dekoder, multiplekser/demultiplekser, brojači, kao i memorijski moduli.

Poglavlje 1

Clock signal

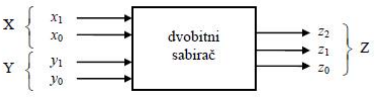
ODNOS IZMEĐU KOMBINACIONIH I SEKVENCIJALNIH KOLA

Ponašanje kombinacionog kola u potpunosti je specificirano istinitosnom tablicom ili skupom jednačina koje za svaku ulaznu kombinaciju daju odgovarajuću izlaznu kombinaciju.

Na logičkom nivou digitalna kola se dele na dve velike klase: ona koja ne poseduju memoriju nazivamo *kombinacionim kolima*, dok su ona koja poseduju memoriju poznata pod imenom *sekvencijalna kola*. Ponašanje kombinacionog kola u potpunosti je specificirano istinitosnom tablicom ili skupom jednačina koje za svaku ulaznu kombinaciju daju odgovarajuću izlaznu kombinaciju. Ova kola preslikavaju ulazne podatke u izlazne, tj. obavljaju izračunavanje u jednom koraku. U praksi postoji neznatno kašnjenje pre nego što izlazni signali promene svoje stanje kao odziv na promene vrednosti signala na ulazu. Vreme odziva je obično veoma kratko, najčešće reda nanosekunde ili kraće, tako da se sa tog aspekta za odziv kombinacionih kola kaže da je trenutan.

U idealnom slučaju, kombinaciono kolo se definiše kao kolo čije je vreme odziva nula. U opštem slučaju, ponašanje kombinacionog kola sa n ulaza koje koristi binarne signale se može opisati istinitosnom tablicom koja ima 2^n vrsta, po jednu za svaku moguću ulaznu kombinaciju. Ilustracije radi, dvobitni sabirač sa slike 1 a ima četiri ulazna signala, a shodno prethodnom zaključku, njegov rad se opisuje istinitosnom tablicom koja ima 16 vrsta (slika 1 b).

Da bi opisali rad sabirača dva 16-bitna broja, kakve obično srećemo u praksi, potrebna je istinitosna tablica koja ima astronomskih 2^{32} = 4,294,967,296 vrsta. Imajući ovo u vidu, logično se nameće potreba za efikasnijim opisom ponašanja kombinacionih kola. Dva moćna sredstva koja se koriste za ovu namenu su Bulova algebra i jezici za opis hardvera.



a)

Ulazi				Izlazi		
x_1	x_0	y_1	y_0	z_2	z_1	z_0
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	1	0
0	0	1	1	0	1	1
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	0	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	1	0	0
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

b)

Slika 1.1 (a) Dvobitni sabirač i (b) njegova istinitosna tablica. [Izvor: Autor]

POJAM SEKVENCIJALNIH KOLA

Sekvencijalna kola moraju da pamte parcijalne rezultate između koraka.

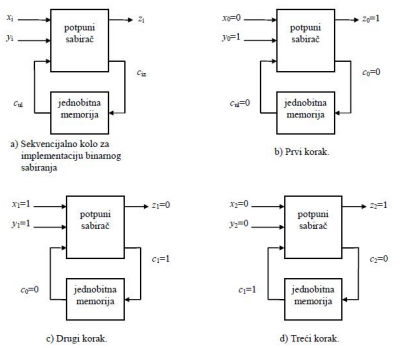
Naime, digitalna kola koja izvršavaju specificirani zadatak obavljaju aktivnost kao sekvencu koraka. Zbog toga za ova kola kažemo da su sekvencijalna. Ovakva kola sadrže memorijske elemente koji zadržavaju, tj. memorišu vrednosti logičkih signala. Kaže se da ukupni sadržaj svih memorijskih elemenata u jednom sekvencijalnom kolu predstavlja *stanje* tog kola. Kada se promene vrednosti ulaza, kolo ili ostane u istom (tj. tekućem) stanju, ili promeni svoje stanje. Vremenom, kao posledica promena ulaza, kolo prolazi kroz sekvencu stanja. Kolo koja se ponašaju na ovaj način zovu se sekvencijalna kola (en. sequential digital circuits).

Sekvencijalna kola moraju da pamte parcijalne rezultate između koraka. Pojmovi sekvencijalno kolo i kolo koje poseduje memoriju predstavljaju sinonime.

Prednost sekvencijalnog rada je ta što je za izvršenje svakog koraka potrebno ugraditi jednostavniji hardver, ali sa druge strane, treba da protekne znatno duži period dok se ne dobije konačan rezultat.

To znači da implementacija nekog zadataka pomoću kombinacione logike rezultira kraćem vremenu generisanja rezultata u odnosu na implementaciju zasnovanu na sekvencijalnoj logici, ali će zato obim hardvera biti veći.

Da bi ukazali na kompromis između obima ugrađenog hardvera i brzine izračunavanja, analiziraćemo primer sabiranja dva n-tobitna broja. Koristeći princip "papira i olovke" primenićemo višekoračni metod kod koga se u datom trenutku sabira odgovarajući par cifara, počev od cifre najmanje težine. Kada suma izračunata u tekućem koraku premaši vrednost najveće cifre "prenosimo jedinicu" ka narednom paru cifara. To znači da je za sabiranje n cifara potrebna sekvenca od n koraka, pri čemu se u svakom koraku cifra x_i prvog broja sabira sa po težini odgovarajućom cifrom y_i drugog broja. Takođe, sumi se dodaje cifra prenosa (0 ili 1) koja je generisana u toku prethodnog koraka. Ilustracije radi, na slici 2 je prikazan sekvencijalni način sabiranja binarnih brojeva. Kao što se vidi sa slike 2 a, u svakom koraku se sabiraju tri bita, x_i, y_i i c_{ul}, gde je c_{ul} bit prenosa iz prethodnog koraka, i određuje bit sume zi i novi bit prenosa c_{iz}. Na slici 2 b-d prikazani su koraci kod izračunavanja 2+3=5 (10+11=101).



Slika 1.2 Korišćenje sekvencijalnog sabirača za izračunavanje 2+3=5. [Izvor: Autor]

STRUKTURA I PODELA SEKVENCIJALNIH KOLA

Primarni izlazi se koriste za upravljanje radom okruženja kola dok se sekundarni izlazi koriste da specificiraju naredno stanje koje će se pamtiti u memoriji.

Odgovarajuća kombinacija sekundarnih ulaznih promenljivih u datom trenutku se naziva **trenutno stanje kola**, dok su sekundarne promenljive poznate kao **promenljive stanja**.

Ako postoji m sekundarnih ulaznih promenljivih, tada sekvencijalno kolo može da se nađe u jednom od 2^m različitih tekućih stanja. Izlazi kombinacionog dela kola se dele na dva skupa. Primarni izlazi se koriste za upravljanje radom okruženja kola dok se sekundarni izlazi koriste da specificiraju ***naredno stanje*** koje će se pamtiti u memoriji. Broj sekundarnih izlaznih promenljivih zavisi od tipa memorije i korišćenog memorijskog elementa.

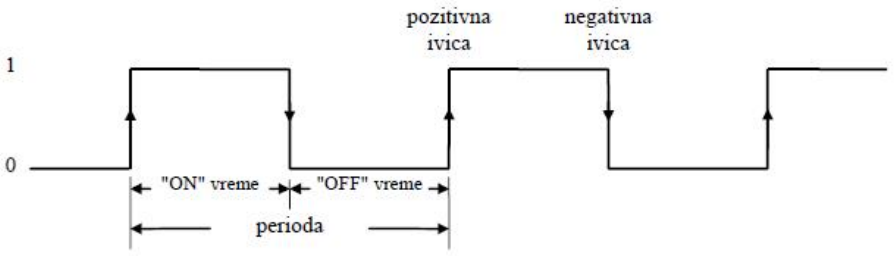
Podela sekvencijalnih kola

Sekvencijalna logička kola se mogu podeliti na ***sinhrona*** *(en.* synchronous*)* i *asinhrona* (en. asynchronous).

Kod sinhronih kola interna stanja se menjaju u diskretnim vremenskim trenucima pod kontrolom impulsa za sinhronizaciju koga nazivamo *taktni* ***signal*** ili *clock signal*.

Talasni oblik taktnog impulsa je obično pravougaoni (slika 3).

"ON" vreme se definiše kao period dok je signal u stanju 1, a*"OFF" vreme* kao period dok je signal u stanju 0.



Slika 1.3 Taktni signal. [Izvor: Autor]

TAKTNI PERIOD

Taktni period je vremenski interval između dve uzastopne promene taktnog signala u istom smeru, tj. između dve rastuće ili između dve opadajuće ivice taktnog signala.

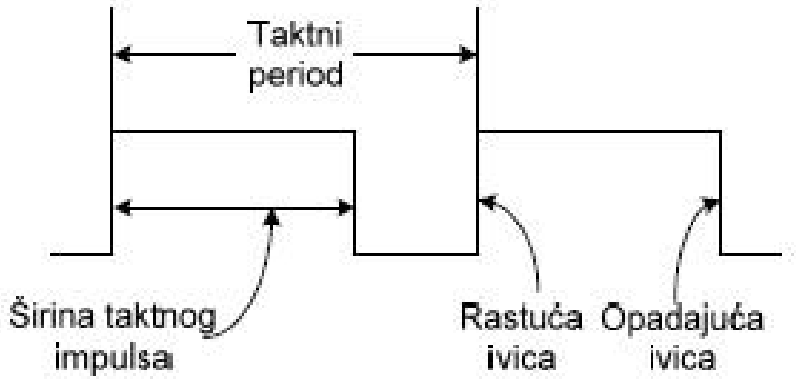
Prelazi stanja kod sinhronih sekvencijalnih kola se obično javljaju u trenutku kada postoje prelazi taktnih impulsa bilo sa 0 na 1 ili sa 1 na 0. Prelaz sa 0 na 1 se naziva ***pozitivna ivica***

ili *usponska ivica*, dok prelaz sa 1 na 0 odgovara ***negativnoj*** ili ***opadajućoj ivici*** taktnog signala. Između sukcesivnih taktnih impulsa ne dolazi do promene informacije koja se čuva u memoriji. Sinhrona sekvencijalna kola su takođe poznata i kao *taktovana sekvencijalna kola*.

Taktni period je vremenski interval između dve uzastopne promene taktnog signala u istom smeru, tj. između dve rastuće ili između dve opadajuće ivice taktnog signala (Slika 4). Recipročna vrednost taktnog perioda je **taktna frekvencija**. **Širina taktnog impulsa** je vreme u toku koga je vrednost taktnog signala jednaka 1.

Faktor popune taktnog signala je količnik širine taktnog impulsa i taktnog perioda.

Kod asinhronih sekvencijalnih kola ne postoji spoljna sinhronizacija (taktni signal) tako da se prelazi kola sa jednog stanja u drugo iniciraju promenom primarnih ulaza. S obzirom da se promene stanja ne dešavaju u specificiranim vremenskim trenucima, asinhrona kola rade sopstvenom brzinom.



Slika 1.4 Karakteristike taktnog signala. [Izvor: Autor]

Poglavlje 2

Memorijska kola

BISTABILANA KOLA

U digitalnoj tehnici kao memorijski elementi koriste se bistabilana kola koja imaju dva stabilna stanja. Bistabilno kolo može da memoriše informaciju od jednog bita.

Element koji zadržava, tj. pamti uspostavljeno stanje i po prestanku dejstva pobudnih signala koji su ih prouzrokovali, naziva se memorijski element. Osnovna karakteristika memorijskih elemenata jeste postojanje stabilnih stanja u kojima mogu ostati neograničeno vreme i koja se mogu menjati pod uticajem ulaznih signala.

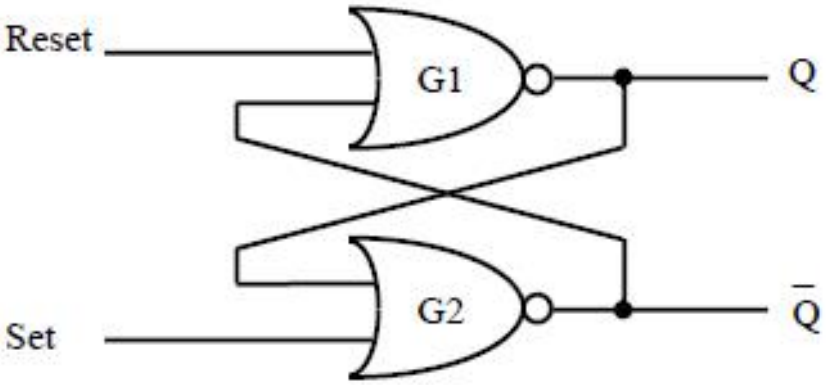
U digitalnoj tehnici kao memorijski elementi koriste se ***bistabilana kola*** koja imaju dva stabilna stanja. Bistabilno kolo može da memoriše informaciju od jednog bita. Dva osnovna tipa bistabilnih kola su: leč (en. latch) kola i flip-flopovi (en. flip-flops).

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

SR LATCH KOLO

SR leč je najjednostavniji memorijski element koji se koristi za projektovanje digitalnih sistema

SR leč **-** SR leč je najjednostavniji memorijski element koji se koristi za projektovanje digitalnih sistema. SR leč čine dva unakrsno spregnuta NOR kola. Leč je logičko kolo memorijskog tipa sa dva izlaza koji su komplementarni jedan u odnosu na drugi. Osnovni leč se može realizovati unakrsnim povezivanjem dva NOR kola kako je to prikazano na slici 1 .



Slika 2.1 Leč kolo sa NOR logičkim kolima. [Izvor: Autor]

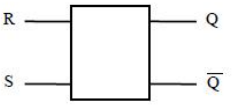
Ulazi leč kola se označavaju sa *Set* i *Reset*. U toku normalnog rada izlazi Q i Q su uvek komplementarni jedan u odnosu na drugi. Usvojimo da su oba ulaza na 0, izlaz Q na 1 a Q na 0. Izlaz kola G1 biće na 1, pa kako izlaz Q preko povratne grane pobuđuje ulaz kola G2, izlaz G2 biće 0. Kolo će zbog toga biti stabilno sa Q na 1 i Q na 0, kako smo i pretpostavili na početku. Ako se sada Reset ulaz postavi na 1, izlaz G1 će se promeniti na 0. Oba ulaza kola G2 biće na 0 tako da će se njegov izlaz promeniti na 1. Leč kolo će sada postati stabilno sa Q=0 i Q=1.

Ponašanje leč kola se može opisati istinitosnom tablicom datoj na slici 2 a. Ukršteni NOR je poznat kao SR (Set-Reset) leč. Logički simbol koji se koristi za predstavljanje SR leča prikazan je na slici 2 b.

a)

Set	Reset	Q	$\overline{Q}$
0	0	nema promene	
0	1	0	1
1	0	1	0
1	1	nedefinisano	

b)



Slika 2.2 (a) Istinitosna tablica i (b) logički simbol SR leča. [Izvor: Autor]

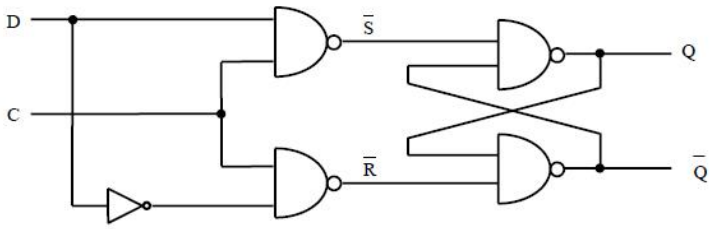
Analizirajući istinitosnu tablicu sa slike 2a uočavamo sledeće. Ulazna kombinacija Set=1 i Reset=1 nije dozvoljena, jer će oba izlaza Q i Q biti postavljena na 0, što je u kontradikciji sa uslovom da su Q i Q komplementarni.

D LEČ

Leč ima samo dva ulaza: D (Data - podaci) i C (Control - upravljački).

Jedan od načina da se eliminiše neželjeno nedefinisano stanje kod SR leča je da se obezbedi da ulazi S i R ne budu nikada istovremeno jednaki 1. Ovo se izvodi kod D leča kako je to prikazano na slici 3.

Leč ima samo dva ulaza: D (en. Data) ulaz za podatke i C (en. Control) upravljački ulaz.



a) Logički dijagram.

C	D	naredno stanje za Q
0	x	nema promene
1	0	Q=0; Reset stanje
1	1	Q=1; Set stanje

b) Funkcionalna tabela.

Slika 2.3 D leč. [Izvor: Autor]

Komplement ulaza D dovodi se preko NI kola na ulaz–S, a ulaz D preko invertora i NI kola na ulaz–R. Sve dok je upravljački ulaz C=0 oba ulaza SR leča su na visoko i kolo ne može da promeni svoje stanje nezavisno od vrednosti D. Ulaz D se uzorkuje (odmerava) kada je C=1. Ako je D=1, Q se postavlja na 1, tj. za kolo kažemo da je u stanju set. Kada je D=0, izlaz Q=0 i kolo je u stanju reset. Logički simbol D leča prikazan je na slici.

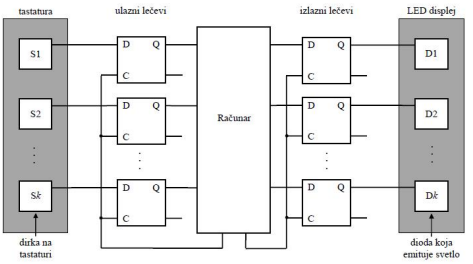
SR i D lečevi su sigurno najjednostavniji i najjevtiniji tipovi memorijskih elemenata koji se koriste kod logičkih kola. Oni se koriste kao elementi za memorisanje (pamćenje) u kolima gde ne postoji direktna povratna veza sa izlaza bilo kog leča preko spoljnih kola na ulaze lečeva (u okviru leča postoje povratne veze). Ovaj zahtev je ispunjen kod strukture prikazane na slici 9. Kao što se vidi sa slike 9, računar prihvata podatke sa ulaznog uređaja (tastatura), a generiše podatke na izlaznom uređaju (LED displej) koristeći lečeve. Svaka kolona lečeva ima zajedničku upravljačku liniju. Lečevi se koriste kao privremni memorijski elementi podataka na putu ulazni uređaj - računar ili računar - izlazni uređaj. Podatak se pamti u leč (lečuje) i ostaje tamo sve dok ne naiđe novi podatak i zameni ga.

FLIP-FLOPOVI

Flip-flop se definiše kao bistabilno kolo koje koristi specijalni upravljački signal C (može i nekoliko takvih signala) radi specificiranja trenutaka u kojima se memorija odaziva na promene.

Kao što smo napomenuli, zbog svoje transparentnosti u radu (kada je C=1, izlaz Q prati stanje na ulazu D) lečevi nisu pogodni za korišćenje kod sekvencijalnih kola koja imaju povratnu strukturu, kao što je ona prikazana na slici 4, tj. kada se izlaz leča preko kombinacione logike dovodi na njegov ulaz. Naime, kada je C=1, izlazni signal (Q ili Q) se vraća preko povratne grane na ulaz D. Neželjene kombinacije propagacionog kašnjenja (kašnjenja signala kroz kola) mogu da uzrokuju višestruke promene, što rezultira novom neodređenom stanju leča. Ovaj fenomen se naziva *problem trke* (en. race problem) i direktna je posledica transparentnosti leča koja omogućava da kada je C=1 (rad leča dozvoljen) kroz leč prođe neograničen broj promena podataka.

Da bi ograničili memorijski element na samo jednu promenu stanja po koraku kada je leču dozvoljen rad, neophodno je koristiti netransparentne memorijske elemente kao što su flip-flopovi.



Slika 2.4 Tipičan način korišćenja lečeva kod računarskih ulazno/izlaznih kola. [Izvor: Autor]

Flip-flopovi

Posmatrajmo bistabilno memorijsko kolo ME koje ima ulaz(e) X, izlaz podataka Q i upravljački(e) ulaz(e) (takt) C. Broj ulaznih i upravljačkih linija, kao i način na koji se upravlja kolom, zavisi od tipa memorije. Kada je rad ME-u dozvoljen, prelaz iz jednog stabilnog stanja u drugo opisuje se relacijom na Slici 5.

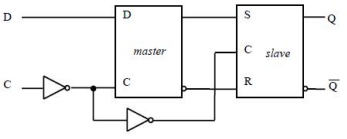
$$Q(t) \xrightarrow{X(t)} Q(t+\tau).$$

Slika 2.5 Relacija za ME kolo .[Izvor: Autor]

IVIČNO OKIDANI D FLIP-FLOP

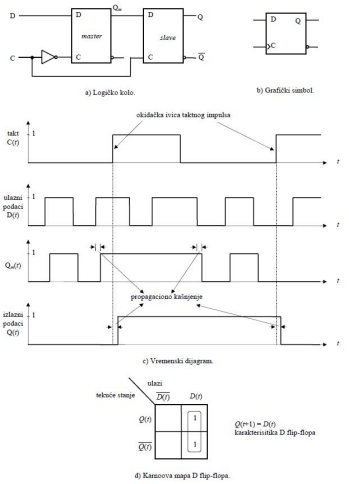
Stanje master leča koje treba kopirati je stanje koje je prisutno u trenutku pojave pozitivne ivice taktnog impulsa.

Logički dijagram D tipa flip-flopa koji se okida na prednju ivicu prikazan je na slici 6 . Flip-flop čine dve celine: ***master*** deo realizovan D lečom i ***slave*** deo koji može biti SR ili D leč. Na ulazu taktnog signala se dodaje invertor. S obzirom da je ***master*** leč D leč, flip-flop ima osobinu da se okida na ivicu, a ne na nivo kao što je to slučaj sa *master-slave*. Kada je C=0, rad ***master*** leča je dozvoljen i transparentan, tj. njegov izlaz sledi stanje na D ulazu. Rad ***slave*** leča je zabranjen i on održava nepromenjeno stanje flip-flopa. Kada se javi pozitivna ivica taktni ulaz se promeni na 1. Rad ***master*** leča se zabranjuje, njegov izlaz zamrzava, a ***slave*** leču je dozvoljen rad tako da on kopira na svom izlazu stanje koje je prisutno na izlazu ***master*** leča. Stanje ***master*** leča koje treba kopirati je stanje koje je prisutno u trenutku pojave pozitivne ivice taktnog impulsa. Zbog ovoga se ponašanje flip-flopa opisuje kao okidanje na ivicu. Kada je C=1, rad ***master*** leča je zabranjen i on ne može da se promeni, tako da stanja oba leča (i *master*-a i *slave*-a) ostaju nepromenjena.



Slika 2.6 D tip flip-flopa koji se okida na pozitivnu ivicu. [Izvor: Autor]

Konačno, kada se C promeni sa 1 na 0, rad *master*-a je dozvoljen i njegov izlaz počinje da sledi vrednost na ulazu D. U toku prelaza sa 1 na 0 rad *slave*-a se zabranjuje tako da bilo kakva promena na *master*-u nema efekta na izlaz *slave*-a. Zbog toga, vrednost koja je memorisana u *slave*-u ostaje nepromenjena u toku ove promene. Kao što smo napomenuli, ivično okidani D flip-flop (ili ***delay*** flip-flop) se može realizovati i pomoću dva D leča i jednog invertora (slika 7). Ovo rešenje se uglavnom koristi kod logičkih kola koja pripadaju CMOS tehnologiji, kao što je 74HC74. Flip-flopovi se obično projektuju sa jednim ili dva dodatna upravljačka ulaza koji su namenjeni za inicijalizaciju početnog stanja flip-flopa. Upravljački signal koji dovodi flip-flop u stanje Q=0 se naziva clear (CLR') ulaz, a onaj koji postavlja flip-flop u stanje Q=1 se naziva preset (PR') ulaz. Uticaj PR' i CLR' upravljačkih signala je nezavisan od taktnog signala pa zbog toga za PR' i CLR' kažemo da su kao ulazi *asinhroni*. Nasuprot njima, D je ***sinhroni*** ulaz u odnosu na taktni signal.



Slika 2.7 D flip-flop koji se okida ivično. [Izvor: Autor]

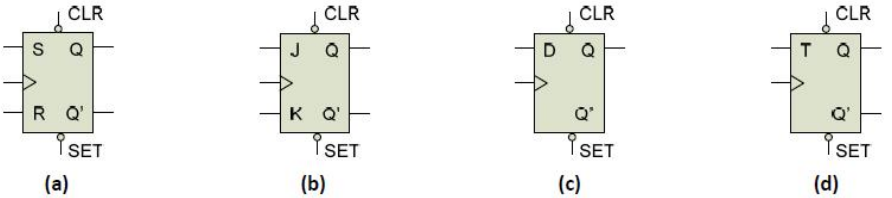
ČETRI TIPA FLIP-FLOPOVA SA ASINHRONIM ULAZIMA

Četri tipa flip-flopova sa asinhronim ulazima su: (a) SR flip-flop; (b) JK flip-flop; (c) D flip-flop; (d) T flip-flop.

Svaki flip-flop je obično dostupan u dve varijante: sa ili bez ulaza za direktno (tj. asinhrono) postavljanje, koji se koriste za setovanje (ulaz *SET*) i resetovanje (ulaz *CLR*) flip-flopa nezavisno od takta i ostalih ulaza. Ovi ulazi se koriste za postavljanje flip-flopa u poznato početno (tj. inicijalno) stanje. Na primer, stanje u koje će se flip-flop spontano postaviti nakon što je uključeno napajanje ne može se predvideti. Zato je neophodno da se pre početka normalnog, sinhronog rada, flip-flop postavi u odgovarajuće početno stanje posredstvom asinhronih ulaza. Ulazi SET i CLR se zovu asinhroni zato što ne zavise od taktnog signala i zbog toga imaju prioritet nad svim ostalim sinhronim ulazima.

Drugim rečima, dok je asinhroni ulaz aktivan, vrednosti ostalih ulaza flip-flopa se ignorišu.

Dejstvo asinhronog ulaza počinje onog trenutka kada se na ulaz dovede aktivan naponski nivo (0 ili 1), i traje sve dok se ulaz ne deaktivira. Ako flip-flop poseduje oba asinhrona ulaza, SET i CLR, njihovo istovremeno dejstvo nije dozvoljeno. Na Sl. 24 su prikazani grafički simboli flip-flopova sa ulazima za direktno postavljanje sa aktivnim niskom naponskim nivoom, što je naznačeno kružićima na odgovarajućim ulazima. Kod asinhronih ulaza sa aktivnim visokim naponskim nivoom, kružići bi bili izostavljeni.



Slika 2.8 Četri tipa flip-flopova sa asinhronim ulazima: (a) SR flip-flop; (b) JK flip-flop; (c) D flip-flop; (d) T flip-flop [Izvor: Autor]

Poglavlje 3

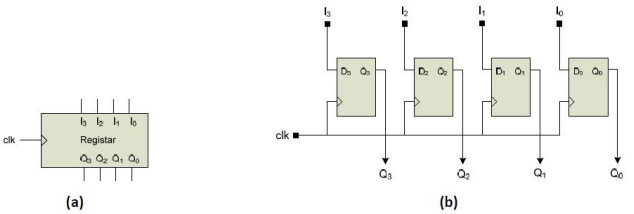
Registri

REGISTAR KAO MEMORIJSKA KOMPONENTA

Sinhronizovano sa aktivnom (rastućom ili opadajućom) ivicom taktnog signala, u svaki flip-flop upisuje se jedan bit informacije.

Registar (en. register) je memorijska komponenta koja se sastoji od *n* flip-flopova sa zajedničkim taktnim signalom.

Sinhronizovano sa aktivnom (rastućom ili opadajućom) ivicom taktnog signala, u svaki flip-flop upisuje se jedan bit informacije. U svom osnovnom obliku, pored taktnog signala, registar ima *n* ulaza i *n* izlaza. Na Sl. 1 prikazan je primer 4-bitnog registra. Grafički simbol 4-bitnog registra dat je na Sl. 1 (a), dok je na Sl. 1 (b) prikazana njegova unutrašnja struktura, koju čine četiri paralelno povezana D flip-flopa.

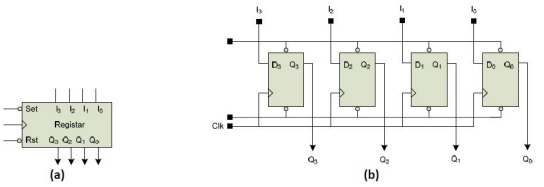


Slika 3.1 4-bitni registar: (a) grafički simbol; (b) unutrašnja struktura [Izvor: Autor]

Funkcionalnost osnovne varijante registra, prikazane na Sl. 1, može se proširiti dodavanjem različitih upravljačkih signala.

Na primer, ako registar treba biti resetovan ili setovan nezavisno od taktnog signala, bilo pri uključenju napajanja, bilo pri pojavi nekih specifičnih događaja, mogu se dodati signali za ***asinhrono resetovanje*** i *setovanje*. Takvo jedno proširenje se postiže zamenom jednostavnih flip-flopova sa Sl. 1(b), flip-flopovima sa ulazima za direktno postavljanje, kao što je prikazano na Sl. 2(b).

Kao što se vidi na Sl. 2 , kratkotrajnim aktiviranjem signala *Rst* sadržaj registra se briše ili resetuje, tj. postavlja na “sve nule”. Slično, registar se setuje, tj. njegov sadržaj postaviti na “sve jedinice”, kratkotrajnim aktiviranjem signala *Set*. (S obzirom da je aktivni nivo signala ***Rst*** i ***Set*** nizak, aktiviranje jednog od ova dva signala znači postavljanje 0 na odgovarajući ulaz). Ulazi ***Rst*** i ***Set*** su nezavisni od taktnog signala i imaju prioritet nad njim. To znači da ako je u trenutku pojave rastuće ivice taktnog signala, ***Set*** ili ***Rst*** jednak 0, ulaz *I* se ignoriše, a registar se setuje, odnosno resetuje.



Slika 3.2 4-bitni registar sa asinhronim resetovanjem i setovanjem: (a) grafički simbol; (b) unutrašnjastruktura. [Izvor: Autor]

REGISTAR SA DOZVOLOM

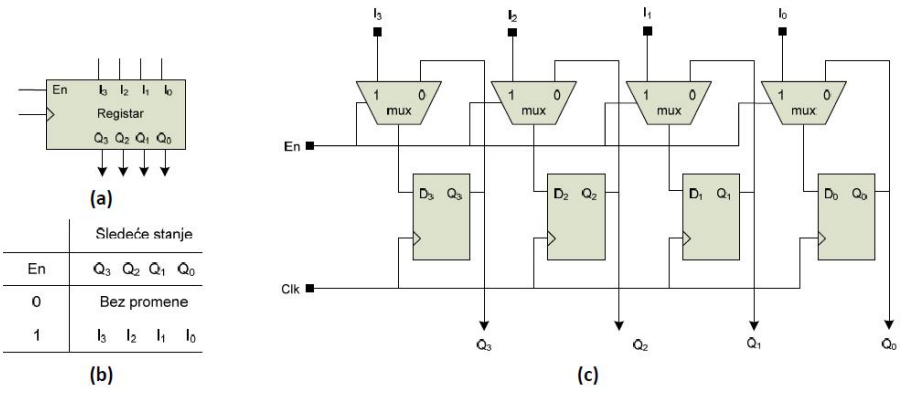
Multiplekseri se mogu iskoristiti i za pomeranje podatka zapamćenog u registru. Ovakav tip registra se zove pomerački registar.

Kod obe varijante registra, prikazane na Sl. 1 i Sl. 2, novi podatak se automatski upisuje u registar sa svakom rastućom ivicom takta. Međutim, kod mnogih digitalnih sistema, podatak koji je upisan u registar ostaje u registru nekoliko taktnih ciklusa pre nego što se upiše novi podatak. Iz tog razloga, mogućnost kontrole upisa predstavlja korisnu funkciju registra. Kontrola upisa se postiže korišćenjem upravljačkog signala *Enable* (*En*) koji kada je 1 dozvoljava upis novog podatka u registar.

Ovakav tip registra se zove **registar sa dozvolom**.

Na slici 3 su prikazani grafički simbol, tabela operacija i unutrašnja struktura registar sa dozvolom. Registar sadrži multipleksere 2-u-1 koji omogućavaju izbor između ulaznog podatka i podatka koje je već u registru.

Signal En upravlja multiplekserima na takav način da kada je En=1, u registar se upisuje novi, tj. ulazni podatak. U suprotnom, ako je En=0, podatak koji je prethodno upisan u registar se vraća na ulaze flip-flopova i sa sledećom rastućom ivicom takta ponovo upisuje u registar - dakle, sadržaj registra ostaje neizmenjen.



Slika 3.3 Registar sa dozvolom: (a) grafički simbol; (b) tabela operacija; (c) unutrašnja struktura.[Izvor: Autor]

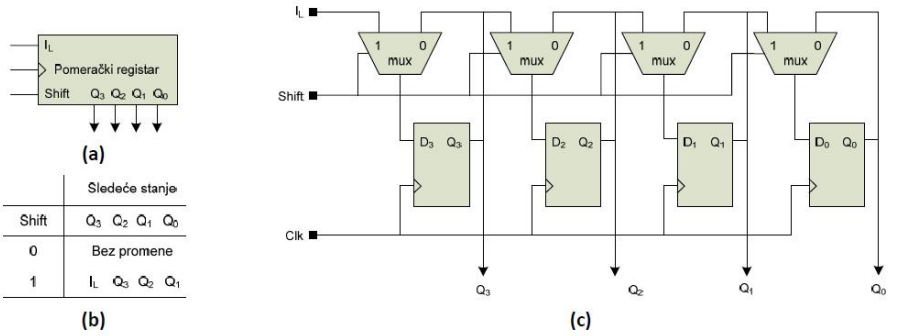
POMERAČKI REGISTAR

Pomerački registaa omogućava pomeranje upisanog sadržaja za jednu bit-poziciju.

Pomerački registar

Na primeru registra sa dozvolom pokazano je kako se postavljanjem multipleksera ispred ulaza flipflopova može kontrolisati upis u registar. Sledeći sličnu logiku, multiplekseri se mogu iskoristiti i za pomeranje podatka zapamćenog u registru. Ovakav tip registra se zove pomerački registar i omogućava pomeranje upisanog sadržaja za jednu bit-poziciju. U osnovnoj varijanti, pomerački registar poseduje upravljački signal *Shift* koji kada je jednak 1 postavlja registar u režim pomeranja.

Na slici 4 je prikazan primer 4-bitnog pomeračkog registra.



Slika 3.4 Četvorobitni pomerački registar sa serijskim ulazom i paralelnim izlazom: (a) grafički simbol; (b) tabela operacija; (c) unutrašnja struktura. [Izvor: Autor]

Poglavlje 4

RAM memorija

RAM (RANDOM-ACCESS MEMORY, ILI MEMORIJA SA PROIZVOLJNIM PRISTUPOM)

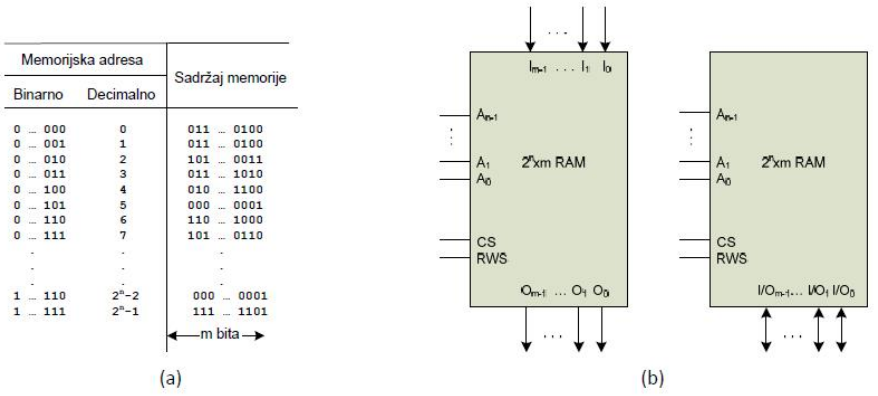
RAM memoriju čine: polje memorijskih ćelija, adresni dekoder i ulazno/ izlazni (U/I) baferi.

U prethodnom odeljku su registri, koji predstavljaju brzu memoriju malog kapaciteta pogodnu za privremeno čuvanje vrednosti promenljivih u toku nekog složenijeg izračunavanja.

S druge strane, memorija sa proizvoljnim pristupom (en. Random-access memory, *RAM*) predstavlja sporiju memoriju daleko većeg kapaciteta pogodnu za dugotrajno smeštanje programa i podataka koji se koriste tokom izračunavanja. Slično registarskom fajlu, RAM je organizovan u vidu polja od 2^n vrsta sa *m* bita u svakoj vrsti.

U opštem slučaju, *n* se kreće između 16 i 32, dok je *m*, obično 1, 4, 8, 16 ili 32. Tipična memorija nalikuje onoj sa Sl. 1 (a). S obzirom da memorija ima 2^n vrsta, za jednoznačnu identifikaciju svake vrste potrebno je *n* adresnih linija.

Pored adresnih linija, memorija poseduje ulazni signal *Chip_select (CS)* koji se koristi prilikom konstrukcije većih memorija na bazi memorijskih čipova manjeg kapaciteta. Uvek kada je *CS*=1, memorija normalno funkcioniše.



Slika 4.1 RAM memorija: (a) memorijske adrese i sadržaj; (b) grafički simboli. [Izvor: Autor]

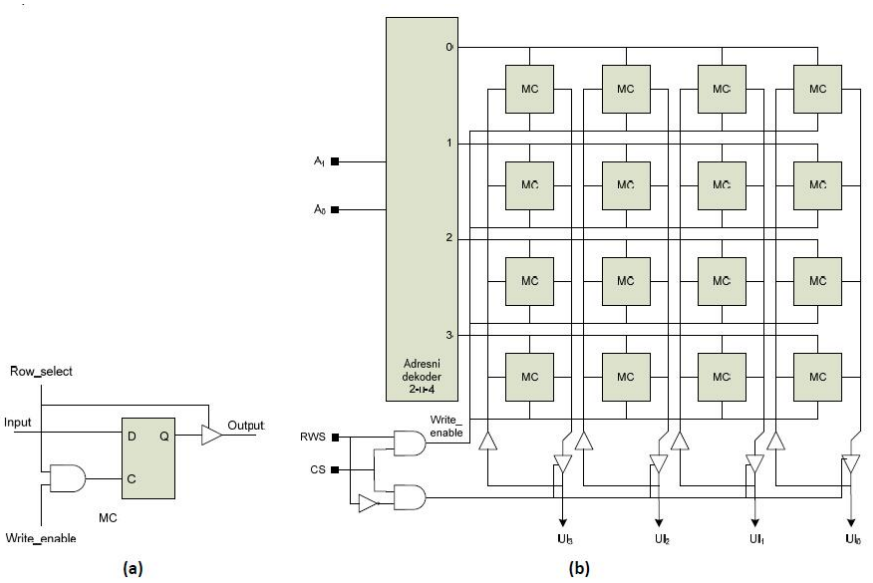
PREDSTAVLJANJE RAM MEMORIJE

RAM memorija može se simbolički predstaviti kao struktura koju čine: taktovani D leč, jedno AND kolo i izlazni trostatički bafer.

Međutim, kada je *CS*=0, pristup memoriji radi čitanja ili upisa je onemogućen. Takođe, memorijski čip ima još jednu upravljačku liniju, *Read/write_select (RWS),* koja bira jednu od dve memorijske operacije: upis ili čitanje. Kada je *RWS*=0, memorija čita svoj sadržaj sa lokacije određene adresinim linijama koji postaje dostupan na izlaznom portu. S druge strane, kada je *RWS*=1, memorija upisuje sadržaj prisutan na ulaznom portu u lokaciju određenu adresnim linijama. RAM, takođe, ima *m*-bitni ulazni i *m*-bitni izlazni port. Za male vrednosti *m* (npr. 1 ili 4), memorija može imati razdvojene ulazne i izlazne portove.

Međutim, da bi se smanjio broj pinova na memorijskom čipu, ulazni i izlazni port su obično objedinjeni u jedinstven ulazno/izlazni port. U opštem slučaju, broj pinova na čipu određuje površnu koju čip zauzima na štampanoj ploči. Grafički simboli oba tipa pakovanja memorije su prikazana na Sl. 1 (b).

RAM memoriju čine: polje memorijskih ćelija, adresni dekoder i ulazno/izlazni (U/I) baferi. Kao što je prikazano na Sl. 2 (a), memorijska ćelija (*memory cell – MC*) može se simbolički predstaviti kao struktura koju čine: taktovani D leč, jedno AND kolo i izlazni trostatički bafer.



Slika 4.2 Organizacija RAM memorije: (a) memorijska ćelija; (b) unutrašnja struktura. [Izvor: Autor]

STATIČKI I DINAMIČKI RAM

SRAM memorija čuva upisani sadržaj sve dok se ne upiše novi ili isključi napajanje.

Kada je signal *Row_select* jednak 1, bit informacije zapamćen u leču se prenosi na izlaz *Output*. Ako je pri tom i signal *Write_enable* jednak 1, vrednost sa ulaza *Input* se pamti u leču. Uočimo da signal *Write_enable* služi kao signal takta za leč. Iako je memorijska ćelija predstavljena uz pomoć leča i dva gejta, treba razumeti da se ona realizuje sa daleko manjim brojem tranzistora u odnosu na registarsku ćeliju.

Shodno načinu implementacije, memorije se dele na statički i dinamički RAM. *Statički RAM (SRAM)* se konstruiše na bazi memorijskih ćelija sa četiri do šest tranzistora kod kojih se leč realizuje uz pomoć unakrsno spregnutih invertora (2 tranzistora), dok se za AND kolo i

trostatički bafer koristi još po jedan tranzistor. SRAM memorija čuva upisani sadržaj sve dok se ne upiše novi ili isključi napajanje. S druge strane, kod *dinamičke RAM memorije (DRAM),* za realizaciju memorijske ćelije koristi se samo jedan tranzistor. Takva memorijska ćelija gubi upisani sadržaj pri svakom čitanju, te zbog toga nakon svakog čitanje mora da sledi upis upravo pročitanog podataka.

Takođe, kao posledica nesavršenog postupka fabrikacije, sadržaj memorijske ćelije se spontano i nepovratno gubi nakon izvesnog vremena po upisu. Da bi memorijska ćelija uspela da sačuva svoj sadržaj, neophodno joj je pristupati sa nekom određenom frekvencijom, ili periodično obnavljati (ili osvežavati) upisani sadržaj.

U toku osvežavanja, operacije čitanja i upisa se suspenduju, što može u nekim slučajevima biti problem. Međutim, bez obzira na sve to, zahvaljujući superiornim karakteristikama u pogledu gustine pakovanja i cene, DRAM memorije se veoma često koriste za projektovanje najrazličitijih elektronskih uređaja.

S druge strane, SRAM memorije su, iako skuplje, brže i zbog toga pogodne za primene koje ne zahtevaju veliku količinu memorije, kao i tamo gde brzi pristup memoriji predstavlja imperativni zahtev. SRAM i DRAM memorije su tzv. nepostojane memorije (en. volatile memories), s obzirom da se njihov sadržaj gubi kada se isključi napajanje. S druge strane, ROM i PROM memorije su tzv. postojane memorije, s obzirom da zadržavaju sadržaj čak i nakon isključenja napajanja.

Na Sl. 2(b) je prikazan primer memorije 4x4 koja ima 16 memorijskih ćelija. Radi pristupa svakoj memorijskoj ćeliji, adresni dekoder dekodira adresu i selektuje jednu od vrsta. Pri tome, ako su oba signala *RWS* i *CS* jednaka 1, u selektovanu vrstu upisuje se novi sadržaj. Iako su sadržaji svih ćelija prisutni na izlaznim linijama, izlazni tro-statički baferi su zakočeni što omogućava novom podatku koji je prisutan na ulazno/izlaznom portu da bude upisan. Međutim, ako je *RWS*=0 i *CS*=1, podatak iz selektovane vrste se kroz tro-statičke bafere prosleđuje na U/I port.

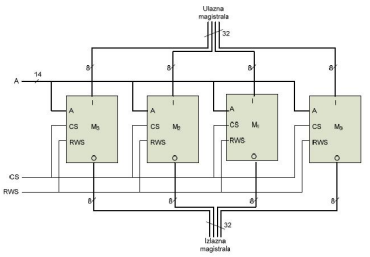
PROŠIRENJE MEMORIJSKE REČI

Proširenje memorijske reči, tj. formiranje memorije sa većim m, postiže se paralelnim vezivanjem nekoliko memorijskih čipova.

Kao što je već rečeno, memorijske komponente se po pravilu proizvode u veličinama 2^nxm, gde n i m mogu da variraju unutar širokog opsega brojeva. Međutim, uopšteno govoreći, za konkretnu implementacionu tehnologiju i godinu proizvodnje, proizvod 2^nxm, tj. kapacitet memorije, je konstanta. Imajući to u vidu, u slučajevima kada memorija potrebnog kapaciteta nije dostupna u vidu monolitnog čipa, ona se mora konstruisati pomoću memorijskih čipova manjeg kapaciteta koji su dostupni na tržištu u vremenu projektovanja. U nastavku ovog odeljka biće opisano kao se realizuju “šire” i veće memorije, tj. kao postići da m i n budu veći od kapaciteta koji je raspoloživ na jednom memorijskom čipu.

Proširenje memorijske reči, tj. formiranje memorije sa većim m, postiže se paralelnim vezivanjem nekoliko memorijskih čipova. Na Sl. 3 je prikazan primer konstrukcije RAM

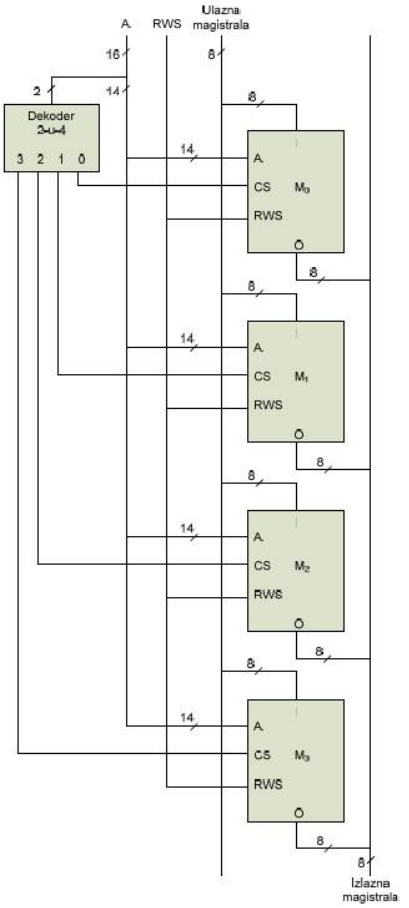
memorije kapaciteta 16Kx32 korišćenjem RAM čipova kapaciteta 16Kx8 (K je oznaka za *kilo*, tj. 210).



Slika 4.3 16Kx32 RAM, realizovan pomoću 16Kx8 RAM. [Izvor: Autor]

Kao što se vidi, kod ovog rešenja, adresne linije, kao i linije *CS* i *RWS*, povezane su sa svim memorijskim čipovima. Uočimo da su ulazna i izlazna magistrala podeljene na četiri grupe od po 8 linija, pri čemu su linije iz iste grupe povezane sa jednom memorijom. Korišćenjem ovog postupka u mogućnosti smo da konstruišemo memoriju bilo koje širine.

Da bi smo realizovali veću memoriju, tj. memoriju sa većim brojem memorijskih lokacija, neophodno je povezati nekoliko memorijskih čipova na red, tako da svaki čip sadrži jedan deo od ukupnog broja memorijskih reči. Princip rešenja prikazan je na Sl. 4 , gde je pomoću četiri RAM čipa kapaciteta 16Kx8 realizovana RAM memorija kapaciteta 64Kx8. Uočimo da u ovom slučaju svi memorijski čipovi dele istu ulazne i istu izlaznu magistralu, kao i zajednički upravljački signal RWS.



Slika 4.4 64Kx8 RAM realizovan pomoću 16Kx8 RAM. [Izvor: Autor]

Poglavlje 5

Stack i queue memorija

STACK MEMORIJA

Stek (stack) ili magacin je memorijska struktura koja se često koristi kako u softveru tako i u hardveru.

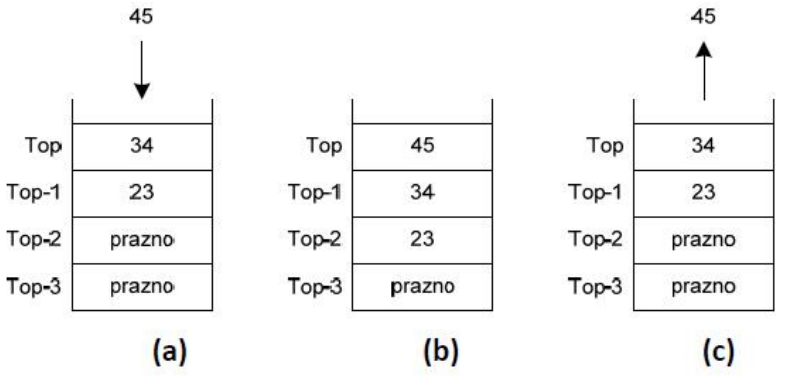
Stek

*Stek (*stack*)* ili magacin je memorijska struktura koja se često koristi kako u softveru tako i u hardveru. Po definiciji, stek je memorija sa ograničenim pristupom. Za razliku od RAM-a gde se bilo kom zapamćenom podatku može pristupati u bilo kom vremenu, podacima zapamćenim u steku pristupa se isključivo preko jedne lokacije: vrh steka.

Drugim rečima, kada se podatak upisuje u stek, ili stavlja na stek (operacije *push*), on se smešta na vrh steka i pri tome se svi prethodno upisani podaci spuštaju za jednu poziciju niz stek.

Suprotno tome, kada se podatak čita iz steka ili uzima sa steka (operacija *pop*), on se sklanja sa vrha steka i pri tome se svi ostali podaci podižu za jednu poziciju naviše uz stek. Na Sl. 1a je prikazan stek dubine 4 (tj. kapaciteta 4 reči) koji inicijalno sadrži dva broja: 34 na lokaciji Top i 23 na lokaciji Top-1. Na Sl. 1b može se videti da stavljanje broja 45 na stek zahteva da brojevi 34 i 23 budu premešteni na lokacije Top-1 i Top-2.

S druge strane, kada se broj 45 uzima sa steka, brojevi 34 i 23 se pomeraju naviše, tako da ponovo zauzimaju lokacije *Top* i *Top*-1 (Sl. 1c). U ovom konkretnom primeru, na stek se može staviti najviše četiri broja, pre nego što se stek napuni. Nakon toga, svaki novi upis u stek znači gubitak podatka sa dna steka.



Slika 5.1 Rad steka: (a) sadržaj steka pre upisa broja 45; (b) sadržaj steka nakon upisa broj 45; (c) sadržaj steka nakon čitanja broja 45. [Izvor: Autor]

QUEUE MEMORIJA

Red čekanja ili queue je struktura koja se često koristi kada treba uravnotežiti zahteve za nekom obradom.

FIFO

FIFO (First-In-First-Out), ili,red čekanja (en. queue), je struktura koja se često koristi kada treba uravnotežiti zahteve za nekom obradom. Zamislimo, na primer, ljude kako stoje ispred šaltera u banci ili kako ulaze u autobus, koji moraju čekati u redu dok ne stignu na red da budu opsluženi.

Slična situacija se javlja kod različitih procesora, nekih integrisanih kola ili bilo kog uređaja koji šalje podatke nekom drugom uređaju radi dalje obrade, u smislu da onda kada u jednom trenutku brzina generisanja podataka nadmaši brzinu kojom se podaci obrađuju, neophodno je između proizvođača i potrošača umetnuti red čekanja, tj. FIFO.

Naravno, u takvim situacijama, brzina kojom proizvođač generiše podatke ne može u nedogled biti veća od brzine kojom potrošač može da prihvata podatke, jer bi to zahtevalo red čekanja beskonačne dužine.

STACK NASPRAM QUEUE MEMORIJE

Stack naspram queue memorije (video objašnjenje)

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

Poglavlje 6

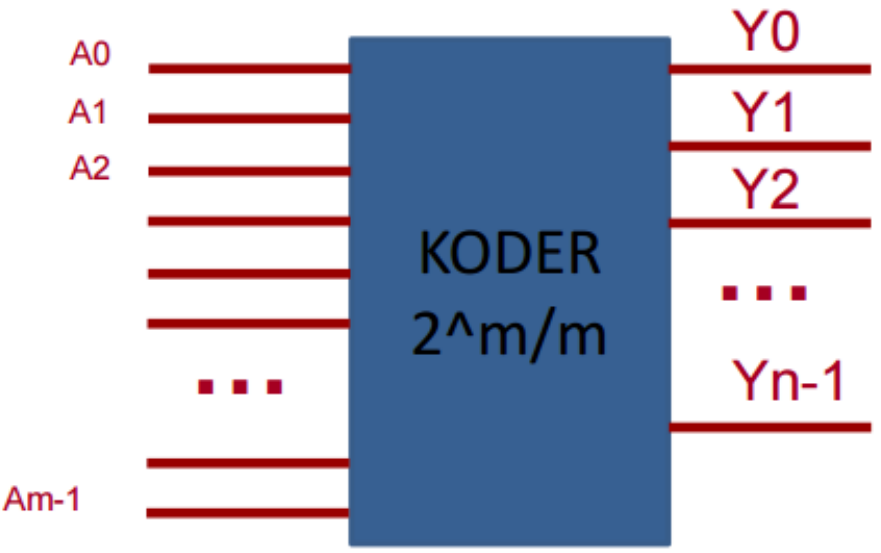
Pokazne Vežbe

KODER

Vežba iz kodera (10 min)

Koder (en. coder) je kombinacioni modul sa više ulaza (m) i više izlaza (n). On obavlja funkciju kodiranja informacija. Kada dovedemo signal na samo jedan ulaz modula, dobijamo izlaznu informaciju u vidu binarnog koda sa (n) cifara.

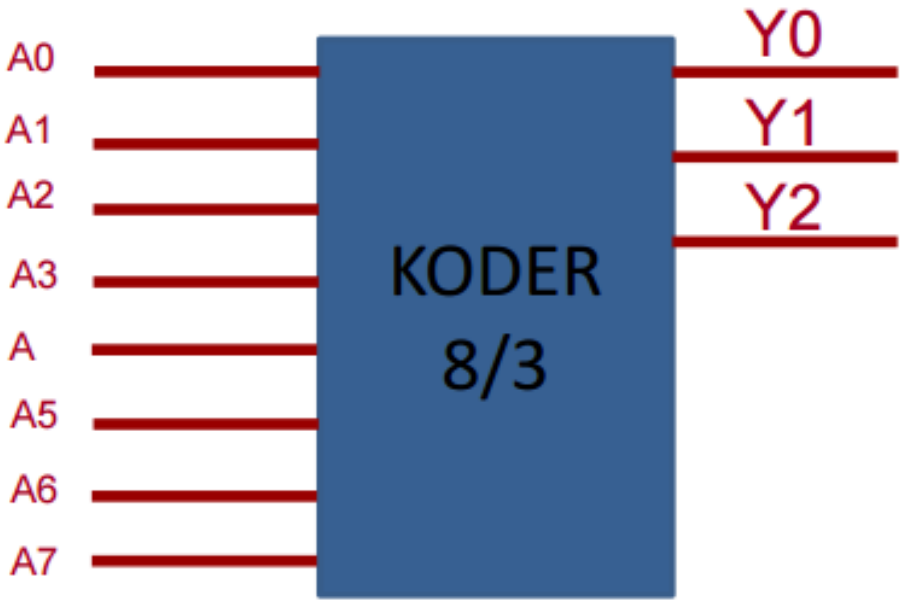
Koder koji ima 2^m ulaza ima m izlaza.



Slika 6.1 Koder [Izvor: Autor]

Koder 8/3

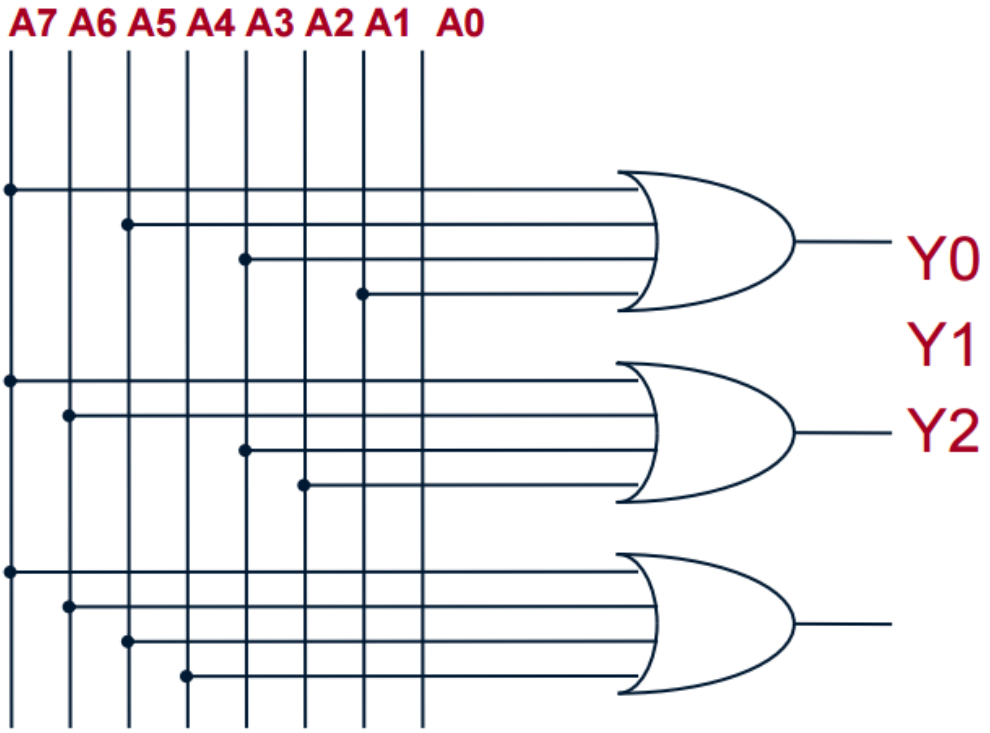
Koder 8/3 ima 8 (2^3) ulaza i 3 izlaza. Signal se može dovesti samo na jedan od ulaza. Kada dovedemo signal do određenog ulaza, na izlazu se generiše binarna kombinacija bitova koja odgovara broju ulaza. Npr. ako je signal na ulazu 3, dobijamo 011 na izlazu.



Slika 6.2 Koder 8/3 [Izvor: Autor]

REALIZACIJA KODERA

Realizacija kodera preko logičkih elemenata (10 min)



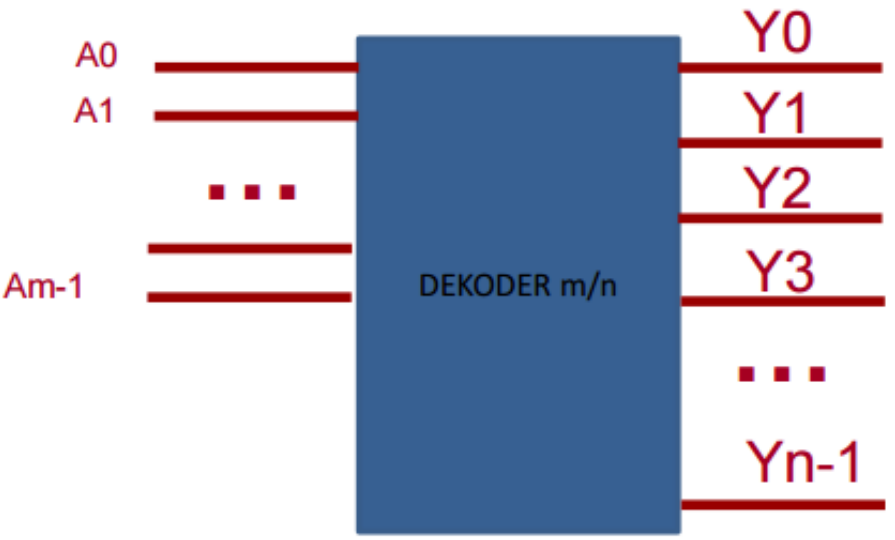
Slika 6.3 Koder realizovan uz pomoć ILI logičkih kola [Izvor: Autor]

DEKODER

Vežba iz Dekodera (8 min)

Dekoder (en. decoder) je kombinacioni modul sa više ulaza (m) i više izlaza (n). On obavlja funkciju dekodiranja binarno kodirane informacije koja je dovedena na ulaz. Na ulazu se aktivira jedan ulaz koji odgovara ulaznoj kombinaciji brojeva.

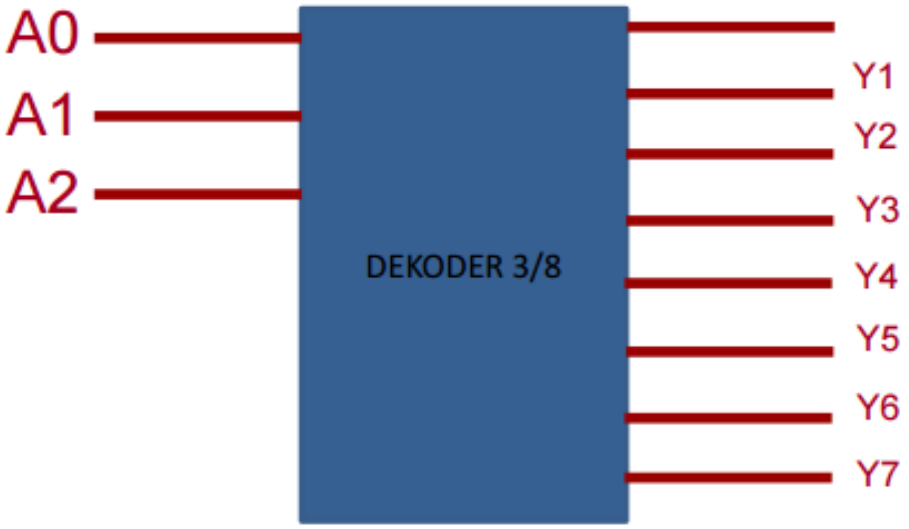
Dekoder koji ima n ulaza, ima 2^n izlaza.



Slika 6.4 Dekoder [Izvor: Autor]

Dekoder 3/8

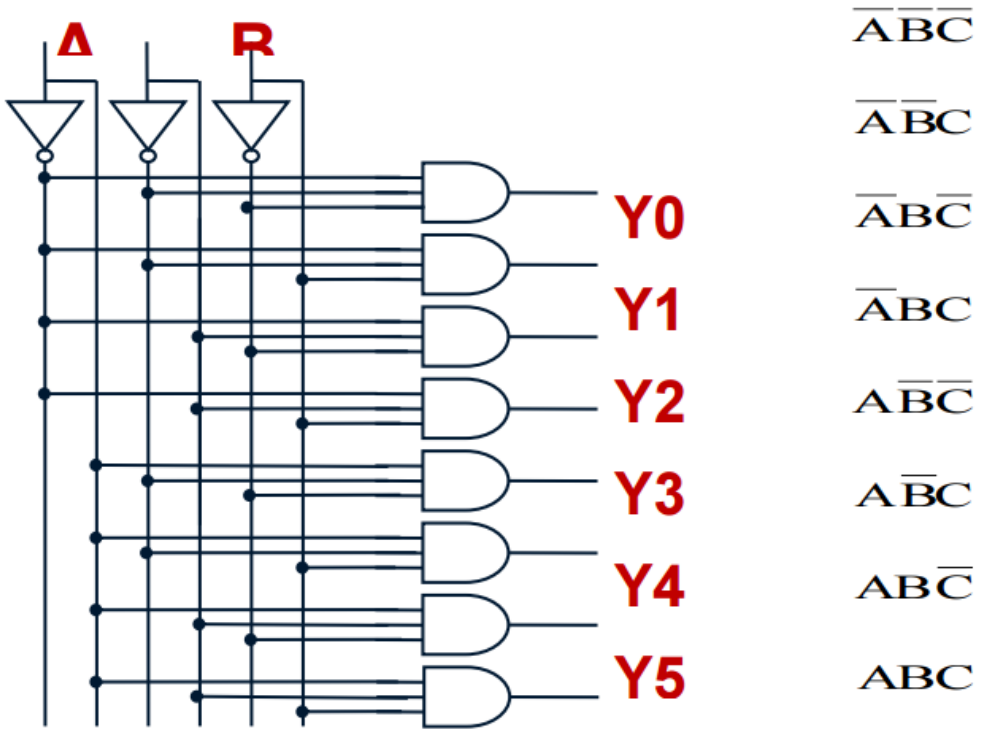
Dekoder 3/8 ima 3 ulaza i 8 (2^3) izlaza. Na ulaz se dovodi binarna kombinacija 3 bita. Kao izlaz se aktivira samo jedan izlaz koji odgovara toj binarnoj kombinaciji. Npr. ako je binarna kombinacija na ulazu 011, izlaz Y3 će se aktivirati.



Slika 6.5 Dekoder 3/8 [Izvor: Autor]

REALIZACIJA DEKODERA

Realizacija dekodera preko logičkih elemenata (7 min)



Slika 6.6 Realizacija dekodera uz pomoć AND i NOT logičkih kola [Izvor: Autor]

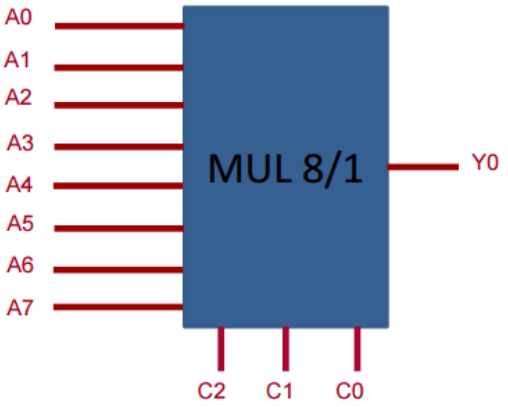
MULTIPLEKSER I DEMULTIPLEKSER

Zadaci iz Multipleksera i Demultipleksera (10 min)

Multiplekser (en. multiplexer, *MUX*) je kombinacioni modul koji ima (n) ulaza, (m) selekcionih signala i jedan izlaz Y. U zavisnosti od binarne kombinacije selekcionog signala, on šalje određeni ulaz direktno na izlaz Y. Multiplekser za 2^n ulaza mora da ima n selekcionih signala, a uvek ima samo jedan izlaz.

Multiplekser 8/1

Multiplekser 8/1 ima 8 (2^3) ulaza i 3 selekciona signala. Kada se na selekcione ulaze C2, C1 i C0 dovede određena binarna kombinacija, ulaz koji odgovara toj binarnoj kombinaciji direktno se prosleđuje na izlaz.

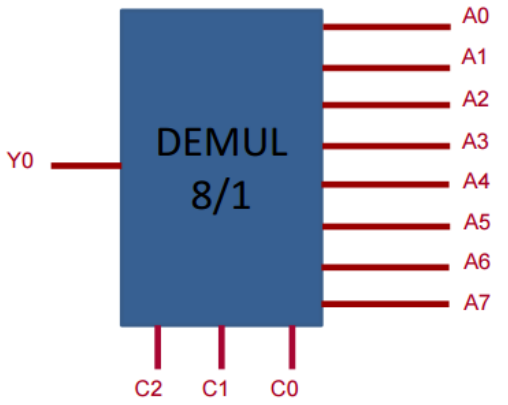


Slika 6.7 Multiplekser [Izvor: Autor]

Demultiplekser (en. demultiplexer, DMUX) je kombinacioni modul koji ima (n) izlaza, (m) selekcionih signala i jedan ulaz Y. U zavisnosti od binarne kombinacije selekcionih signala, on šalje ulaz Y na neki od izlaza. Demultiplekser za 2^n izlaza mora da ima n selekcionih signala, a uvek ima samo jedan ulaz.

Demultiplekser 1/8

Demultiplekser 1/8 ima 8 (2^3) izlaza i 3 selekciona signala kao i jedan ulaz. Kada se na kontrolne signala C2, C1 i C0 dovede određena binarna kombinacija signal koji je na ulazu se šalje uzlazu koji odgovara binarnoj kombinaciji kontrolnih signala. Npr. ako se dovede na C2, C1 i C0 binarna kombinacija 101, ulaz Y će biti prosleđen na izlaz A5.



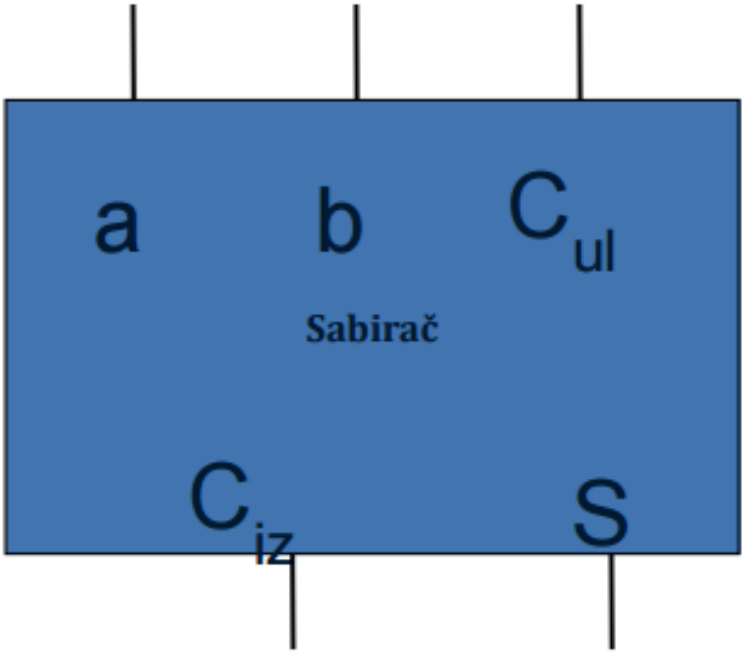
Slika 6.8 Demultiplekser [Izvor: Autor]

POTPUNI SABIRAČ I BROJAČ

Zadaci iz potpunog sabirača i brojača (10 min)

Potpuni sabirač ima 3 ulazna signala. Dva ulazna signala (a i b) su brojevi koji se sabiraju, a treći signal je prenos iz prethodnog sabiranja (C_{ul}) (Carry In).

Potpuni sabirač ima 2 izlaza, jedan je rezultat sabiranja, a drugi signal predstavlja prenos (C_{iz}) (Carry Out).

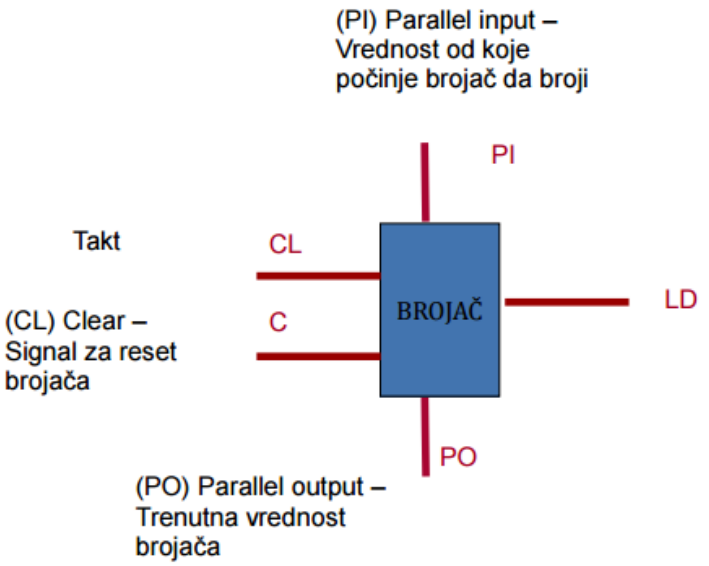


Slika 6.9 Potpuni jednobitni sabirač [Izvor: Autor]

Brojači su sekvencijalni moduli koji nailaskom na takt daju binarni izlaz povećan ili smanjen u odnosu na prošli za jedan i to u zavisnosti da li je brojač ikrementalni ili dekrementalni (rastući ili opadajući).

Broj različitih cifara koje brojač može da generiše nazivamo moduo brojača.

Brojač po modulu M broji od 0 do M-1 nakon čega se resetuje i broji ponovo.

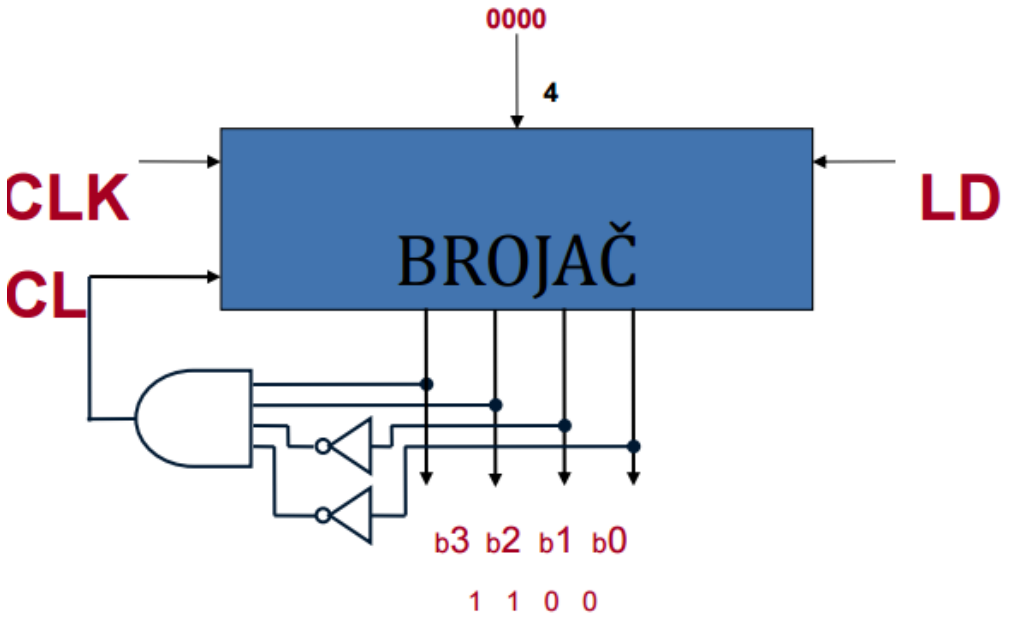


Slika 6.10 Brojač [Izvor: Autor]

BROJAČ PO MODULU 13

Zadatak: Brojač po modulu 13 (10 min)

Brojač po modulu 13 broj od 0 do 12 pa se zatim resetuje. Pošto brojač broji do 13 treba nam brojač sa 4 bita. Brojač počinje od 0000 (0) a resetuje se na takt posle 1100 (12).



Slika 6.11 Brojač po modulu 13 [Izvor: Autor]

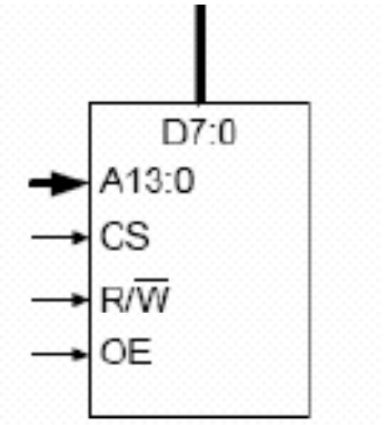
PROŠIRENJE MEMORIJSKIH MODULA

Zadatak iz proširenja memorijskih modula 1/2 (15 minuta)

Zadatak 1. (15 minuta)
Dat je memorijski modul 16Kx8.
a) Skicirati modul.b) Koristeći date memorijske module i potrebna kola realizovati memorijski modul 64Kx24

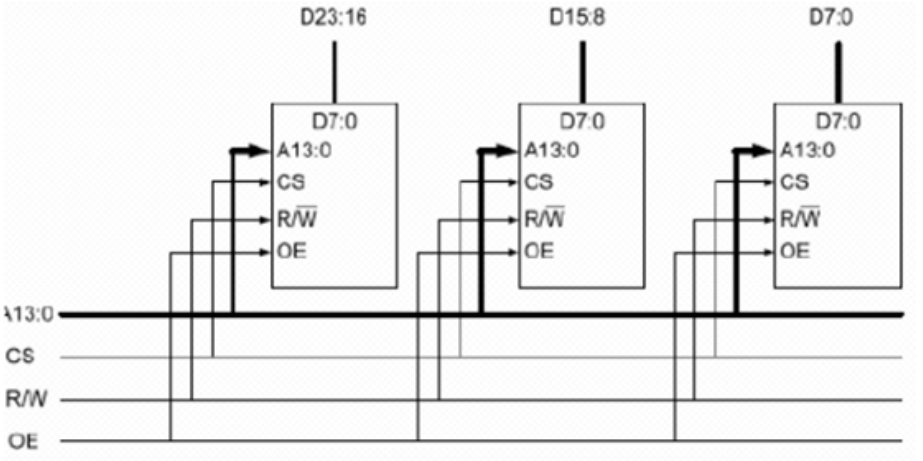
Rešenje:
a) Da bi skicirali modul, neophodno je obaviti 2 koraka
1. odrediti adresni prostor modula (adresne linije)
2. odrediti veličinu data magistale
3. Tu su svakako 3 neophodna signala, CS, R/W, OE

U konkretnom slučaju
1. 16K=2^{14} , što znači da modul ima 14 adresnih linija, A0-A13
2. Iz specifikacije da je to modul 16Kx8, zaključujemo da modul ima 8 bita podataka, D0-D7



Slika 6.12 Memorijski modul 16Kx8. I[Izvor: Autor]

b) Proširenje: od modula 16Kx8, napraviti modul 64Kx24
Moguća su sledeća proširenja:
1. proširenje magistrale podataka (po horizontali) 2. proširenja adresnog prostora (po vertikali) 3. oba proširenja
U ovom slučaju imamo oba proširenja
data proširenje: 8bita->24bita trebaju nam po 3 modula po horizontali
adresno proširenje: 16K ->64K: trebaju nam 4 vertikalne grupe
ukupno nam treba 4x3=12 modula
Prvo radimo proširenje po horizontali



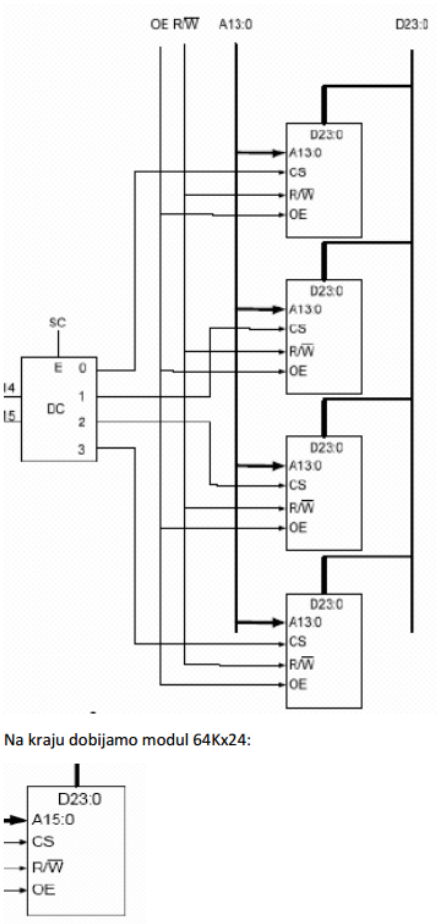
Slika 6.13 Proširenje po horizontali [Izvor: Autor]

PROŠIRENJE MEMORIJSKIH MODULA - PROŠIRENJE PO VERTIKALI

Zadatak iz proširenja memorijskih modula, proširenje po vertikali

Zatim vršimo nad tim modulom proširenje po vertikali.

Za proširenje od 16K na 64K, trebaju nam četiri vertikalne grupe, ali i obavezan dekoder, i to dekoder 2/4 u ovom slučaju. Obratite pažnju, na sve horizontalne grupe se vode niži bitovi adrese, od A0-A13, viši delovi adrese A14-A15 se vode na dekoder koji generiše četiri CS signala za svaku vertikalnu grupu.



Slika 6.14 Proširenje memorijskog modula po horizontali. [Izvor: Autor]

Poglavlje 7

Zadaci za samostalni rad

ZADACI ZA SAMOSTALNI RAD IZ KARNOOVIH MAPA

Zadaci za samostalni rad rade se ukupnp 120 minuta

Na osnovu datih funkcija, uraditi minimalizaciju koristeći Karnoove mape, i zatim realizovati logičku funkciju.

```
f = P{0,1,2,3,8,13}
f = P{0,3,4,6,7,8,9,15}
f = P{1,3,5,7,9,11,13}
f = P{1,2,3,6,7,8,14,15}
f = P{1,3,4,7,8,9,10,12}
f = P{0,2,4,6,13,14,15}
f = Q{0,1,2,3,4,6,8,10}
f = Q{0,3,4,6,12,13,14}
f = Q{1,3,5,7,8,9,10,11}
f = Q{1,2,3,6,7,10,11}
f = Q{1,3,4,7,12,14,15}
f = Q{0,2,4,6,8,10,12}
```

DODATNI MATERIJALI

Dodatni materijali iz 4. lekcije

Karnoove mape

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

SR Latch | NOR and NAND SR Latch

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

Poglavlje 8

Domaći zadatak

DOMAĆI ZADATAK #4

Domaći zadatak #4 okvirno se radi 45 min

Napomena:

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

1. Realizovati modul brojača po modulu M. Moduo M računate tako što poslednju vrednost broja indeksa pomnožite sa 2.

Studenti kojima se indeksa završava na 0 ili 1 uzimaju predposlednju vrednost.
Primer:
• Ako je broj indeksa 4087 onda treba projektovati sinhroni brojač po modulu 14.
• Ako je broj indeksa 3960 onda treba projektovati sinhroni brojač po modulu 12.

Početno stanje brojača je 0.

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.

Poglavlje 9

Zaključak

ZAKLJUČAK

Rezime lekcije #4

U ovoj lekciji bilo je reči najpre o sekvencijalnim kolima, čiji se trenutni izlaz računa ne samo u zavisnosti od trenutnih stanja na ulazu, već i od prethodnih izlaza.

Predstavljena su latch kola i flip-flopovi, i predstavnici istih jesu SR-latch i D-flip-flip.

Ova kola su pogodna za skladištenje informacija, pa se koriste za realizaciju memorijskih i registarskih elemenata.

Zatim, bilo je reči o stack i queue memoriji, koje se uveliko koriste u različitim realizacijama.

Literatura:
A. Tanenbaum, Structured Computer Organization, Chapter 03, pp. 169 - 184.