

Osnovi mikroprocesorskih i mikrokontrolerskih sistema - Rad sa periferijama

prof. dr Ivan Mezei

13. maj 2022.

Glava 7

Rad sa periferijama

Povezivanje mikrokontrolera i periferija je veoma važna tema jer je to način da mikrokontroler dobije svoju svrhu - upravljanje, kontrola i rad sa različitim periferijama koje predstavljaju okruženje mikrokontrolera. Svaki mikrokontroler ima ulazno-izlazne portove koji predstavljaju osnovnu mogućnost povezivanja. Radi obezbeđenja raznih mogućnosti povezivanja proizvođači najčešće omogućavaju alternativne funkcije pinova portova. Oni se najčešće mogu softverski kontrolisati podešavanjem odgovarajućih registara specijalne namene datog mikrokontrolera. Potpuna sistematizacija različitih načina i metoda povezivanja periferija sa mikrokontrolerima prevazilazi potrebe i okvire ovog predmeta. Ovde ćemo dati osnovnu sistematizaciju i odgovarajuće primere. Periferni uređaji mogu biti veoma raznoliki od veoma jednostavnih kao što su npr. svetleće diode (LED), tasteri, prekidači, preko složenijih kao što su sedmosegmentni LED displeji, memorije, drajverska kola za motore, razni analogni i digitalni senzori, do složenih periferija koje na sebi često imaju određenu upravljačku logiku, namenski mikroprocesor kao npr. LCD ili TFT displeji itd.

7.1 Načini povezivanja periferija i mikrokontrolera

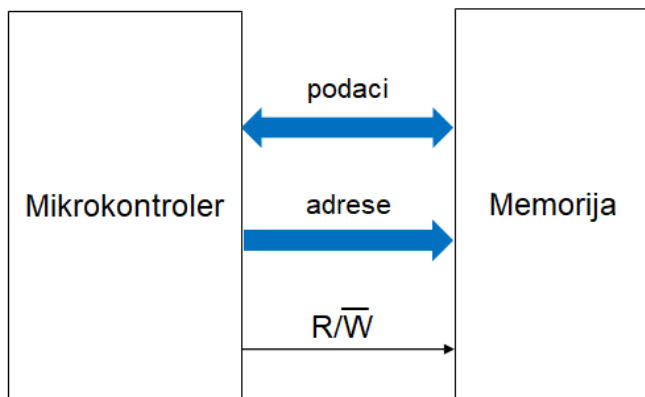
Uzimajući u obzir različite načine povezivanja mikrokontrolera i periferija možemo reći da postoje sledeći načini:

1. direktno povezivanje,

2. baferovano povezivanje,
3. multipleksno povezivanje,
4. memorijski mapirano povezivanje,
5. povezivanje uz konverzije (npr. serijsko-paralelna, paralelna-serijska, analogno-digitalna i dr.), i
6. povezivanje na bazi standardnih protokola (npr. RS232, I2C, SPI i dr.).

7.1.1 Direktno povezivanje

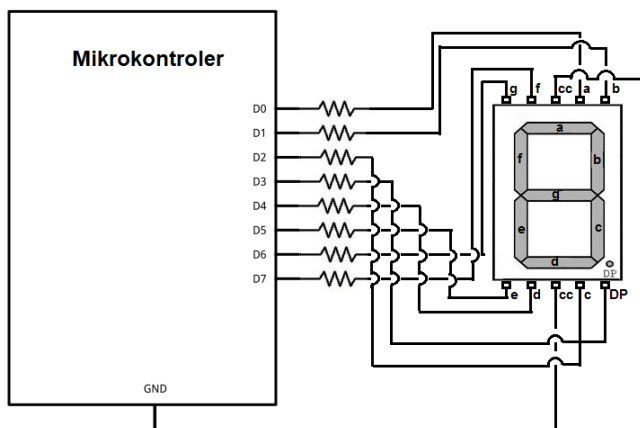
Direktno povezivanje periferija je najjednostavniji način povezivanja mikrokontrolera i periferija. Ovaj način ponekad je moguć ukoliko je mikrokontrolerski sistem koji projektujemo takav da možemo periferije direktno povezati. Direktno povezivanje se vrši direktnim povezivanjem pinova porta mikrokontrolera i periferija. Glavna prednost direktnog povezivanja je jednostavnost, a glavna mana je što je tako moguće povezati samo ograničen, najčešće mali, broj periferija.



Slika 7.1: Direktno povezivanje mikrokontrolera i memorije

Na slici 7.1 prikazan je sistem kod koga je jedna memorija direktno povezana sa mikrokontrolerom. Ovo je moguće ukoliko imamo posebnu spoljašnju magistralu podataka i posebnu spoljašnju adresnu magistralu. Ovo je kod nekih mikrokontrolera slučaj, ali relativno retko jer zahteva značajan broj dodatnih pinova na kućištu što povećava cenu mikrokontrolera. Kada mikrokontroleri nemaju posebne

magistrale za povezivanje sa spoljašnjim periferijama, što je najčešći slučaj, tada se obično radi multipleksiranje adresnih linija i linija podataka i potrebno je baferovano povezivanje.



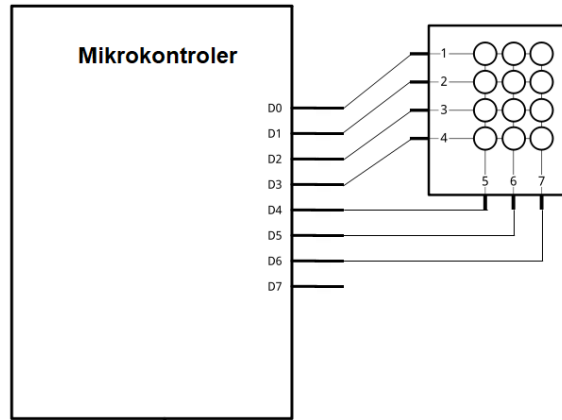
Slika 7.2: Direktno povezivanje mikrokontrolera i 7-segmentnog displeja

Za rad mikrokontrolera su uobičajene i sledeće osnovne komponente. Kvarc i dva kondenzatora koji su povezani na ulaze sa oznakom XTAL, uz unutrašnje komponente mikrokontrolera, čine oscilatorno kolo koje omogućava generisanje signala takta. Taster (ređe prekidač), kondenzator i otpornik koji su povezani na RST ulaz mikrokontrolera su komponente koje realizuju reset kolo. Povezuju se na takav način da omogućavaju aktivan signal na RST ulazu mikrokontrolera dovoljno dugo da prođu svi tranzijentni procesi koji se dešavaju tokom reseta.

Drugi primer direktnog povezivanja je dat na slici 7.2. Ovde je povezan jedan 7-segmentni displej i za to je potrebno svih osam pinova porta D (ostali detalji nisu prikazani na ovoj slici). S obzirom da je pin zajedničkih katoda LED segmenata (eng. *common cathode* - CC) povezan direktno na masu (GND) očigledno je da se segmenti uključuju odmah po pojavi odgovarajućih vrednosti na pinovima porta D. Ukoliko bi smo želeli da povežemo četiri takva displeja za potrebe realizacije, na primer, digitalnog sata, potrebno je da mikrokontroler ima bar četiri 8-bitna porta. Očigledno je da se na ovakav način većina pinova brzo iskoristi.

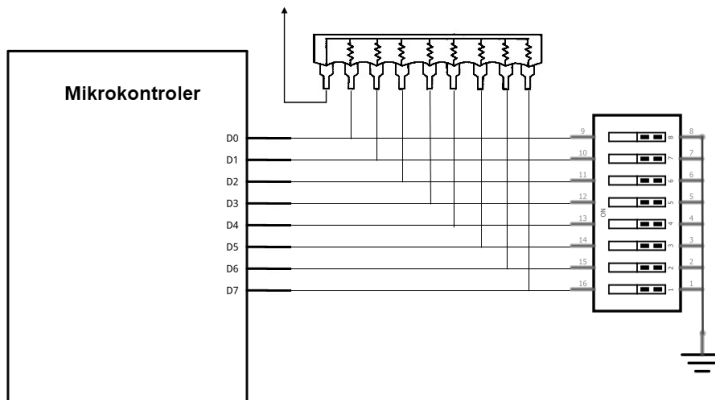
Savitljiva matrična tastatura ili matrična tastatura napravljena od tastera može se takođe direktno povezati sa mikrokontrolerom. Ovo je prikazano na slici 7.3 na primeru tastature 3x4 (3 kolone i 4 vrste). Pritiskom bilo kog tastera se

kratko spajaju odgovarajuća vrsta i kolona.



Slika 7.3: Direktno povezivanje mikrokontrolera i matrične tastature

Kao poslednji primer direktnog povezivanja periferija prikazani su na slici 7.4 8 prekidača realizovanih kao mikroprekidači. Na slici se vidi da u jednom položaju prekidač dovodi nulu na odgovarajući pin porta D, a u drugom preko otpornika za ograničenje struje dovodi jedinicu. Na isti način moguće je povezati i tastere.



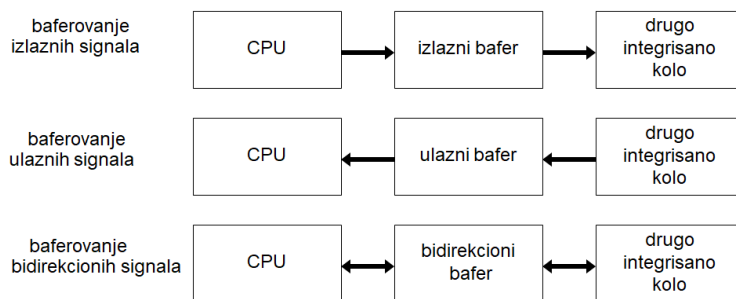
Slika 7.4: Direktno povezivanje mikrokontrolera i 8 mikroprekidača

7.1.2 Baferovano povezivanje

U realnim mikroprocesorskim i mikrokontrolerskim sistemima, signali mikroprocesora ili mikrokontrolera često se baferuju. Pojam 'baferovanje' podrazumeva da se između dve komponente koje razmenjuju signale stavlja bafersko kolo (često ovo kolo zovemo kraće bafer, eng. *buffer*) tj. posebno integrisano kolo preko koga komponente posredno razmenjuju signale. Iako bafer može da se koristi i za jedan signal u mikroprocesorskim i mikrokontrolerskim sistemima se po pravilu misli na više od jednog signala (npr. osam).

U ovoj sekciji pokažaćemo baferovanje signala mikrokontrolera, ali treba imati u vidu da mogu da se baferuju signali bilo koje druge komponente, na primer memorije.

Pošto signali mikrokontrolera mogu biti ulazni, izlazni ili bidirekcionni, razlikujemo slučajeve baferovanja ulaznih, izlaznih i bidirekcionnih signala, kao što to prikazuje slika 7.5.



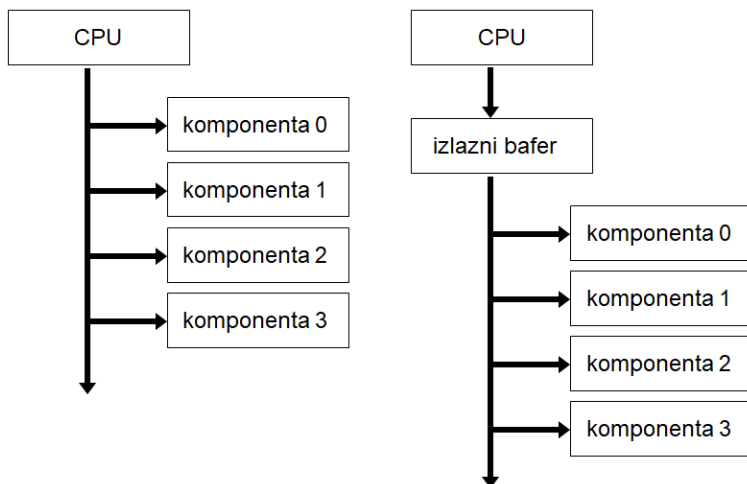
Slika 7.5: Vrste bafera

Baferi se koriste za:

- povećanje faktora grananja izlaznih signala mikrokontrolera (ili neke druge komponente na čije izlazne signale se stavlja bafer),
- zaštitu mikrokontrolera od neočekivanih strujnih ili naponskih udara, i
- privremeno prihvatanje i pamćenje signala.

Faktor grananja (eng. *fanout*) nekog kola određuje koliko je moguće opteretiti neki izlaz, a da on još uvek pravilno radi. Pod opterećenjem se obično misli na izlazne strujne mogućnosti. U slučaju logičkih kola kaže se na izlaz komponente

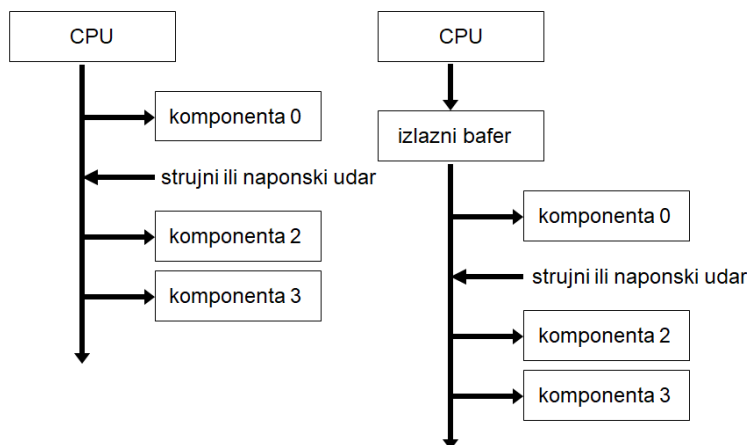
je moguće povezati, na primer, 10 logičkih kola izrađenih u istoj tehnologiji, ukoliko je faktor grananja bar 10. Povećanje faktora grananja važno je u slučajevima kada je na zajedničke linije za prenos signala vezan broj komponenata koji nadmašuje faktor grananja kola. U ovim slučajevima bafer pojačava izlazne strujne mogućnosti mikrokontrolera i tako obezbeđuje pravilan rad sistema. Idejno rešenje za povećanje faktora grananja jednosmernih signala prikazano je na slici 7.6. Na slici levo prikazan je mikrokontroler na čijim su izlaznim linijama priključene četiri komponente. Ako pretpostavimo da je izlazni faktor grananja signala mikrokontrolera jednak 3, onda izlazni stepen na spoljnjim priključcima mikrokontrolera nije u stanju da obezbedi dovoljnu struju za ulazne stepene četiri komponente. Rešenje ovoga problema prikazano je na slici 7.6 desno. Na izlazne priključke mikrokontrolera vezuje se bafer koji sa jedne strane prihvata izlazne signale mikrokontrolera, a sa druge strane ima dovoljan faktor grananja da može obezbediti dovoljnu struju za ulazne stepene priključenih komponenata.



Slika 7.6: Povećanje faktora grananja baferovanjem

Zaštita mikroprocesora i mikrokontrolera od strujnih i naponskih udara ilustrirana je na slici 7.7. Pretpostavimo da je komponenta 1 u kvaru i da se preko njenih priključaka prenosi strujni ili naponski udar za zajedničke linije za prenos signala. Očigledno je da ovaj udar u sistemu bez bafera može da ošteti mikrokontroler. Sa druge strane, bafer je prvi na udaru i tako štiti mikrokontroler, koji je obično najskuplja i vitalna komponenta za rad sistema. Ukoliko se u sistemu

predviđa pojava ovakvih udara onda je obavezna zaštita galvanskim odvajanjem (preko optokaplera ili transformatora).



Slika 7.7: Zaštita od strujnih udara

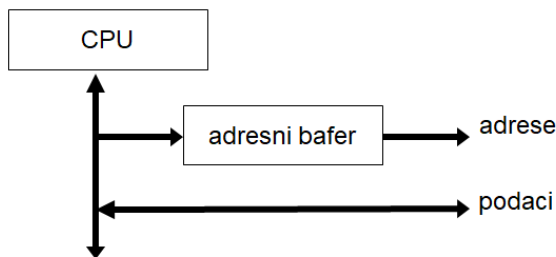
Privremeno prihvatanje signala obično se koristi u slučajevima kada se iste spoljne linije koriste u vremenskom multipleksu za prenos različitih signala. Pretpostavimo da se preko istih spoljnih linija mikrokontrolera prvo prenose adresni signali, a zatim signali za prenos podataka. Ova tehnika je česta u praksi i naziva se multipleksiranje adresa i podataka. Slika 7.8 ilustruje primenu bafera sa privremenim prihvatanjem adresnih signala (adresni bafer) čime se demultipleksiraju spoljni signali mikrokontrolera.

Za demultipleksiranje se koristi bafer koji u periodu kada se na spoljnim priključcima pojave adresni signali, prihvati i upamti ove signale i na svom izlazu formira signale adresne magistrale. Bafer pamti vrednosti adresnih signala u toku drugog dela mašinskog ciklusa kada se preko spoljnih priključaka mikrokontrolera prenose signali podataka. Na ovaj način odvajaju se dve magistrale čiji signali su multipleksirani na spoljnim priključcima mikrokontrolera.

Podela bafera

U praksi se koriste baferi sa različitim osobinama:

- u odnosu na smer prenosa signala: jednosmerni i dvosmerni baferi,



Slika 7.8: Demultipleksiranje adresa i podataka

- u odnosu na konfiguraciju izlaznog stepena: baferi koji na svojim izlazima mogu da imaju stanje visoke impedanse i koji nemaju na izlazu stanje visoke impedanse,
- u odnosu na mogućnost pamćenja signala: baferi koji imaju unutrašnje registre za privremeno prihvatanje i pamćenje ulaznih signala i baferi bez unutrašnjih registara, i
- u odnosu na invertovanje ulaznog signala: baferi koji ne invertuju i baferi koji invertuju logičke nivoe ulaznih signala.

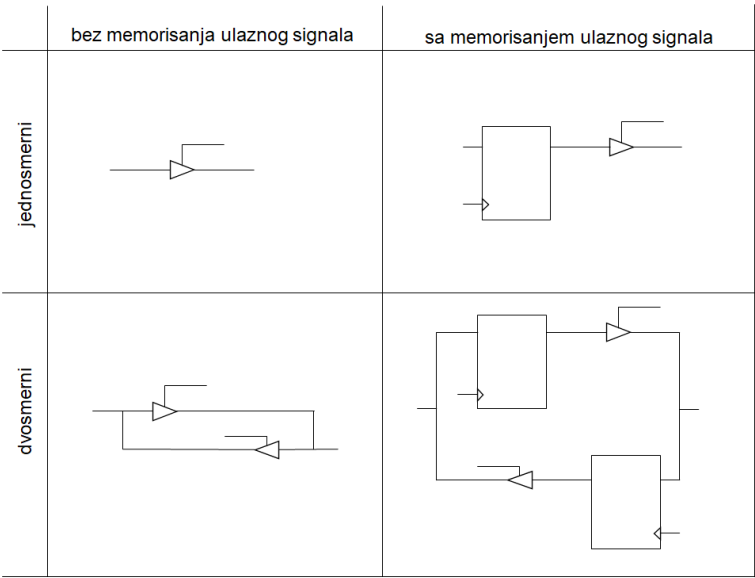
U skladu sa navedenim alternativama, baferi imaju upravljačke signale za:

- određivanje smera prenosa signala,
- prevođenje izlaza u stanje visoke impedanse, i
- za upis vrednosti ulaznih signala u unutrašnji registar bafera.

Baferi se obično sastoje od nekoliko identičnih stepeni koji imaju zajedničke upravljačke signale. Na slici 7.9 dati su primeri tipičnih baferskih stepeni koji imaju mogućnost prevođenja izlaznog signala u stanje visoke impedanse i koji ne invertuju logički nivo ulaznog signala.

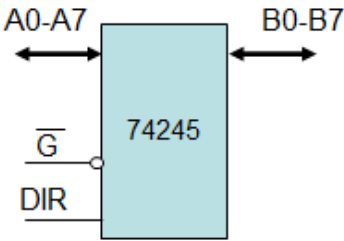
Primeri bafera

Tipičan primer 4-bitnog jednosmernog bafera bez memorisanja je kolo 74125. Unutar njega se nalazi 4 trostatička bafera. Kontrolni signal je aktivan na nizak nivo.



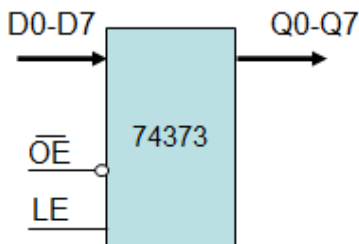
Slika 7.9: Podela bafera

Tipičan primer dvosmernog 8-bitnog bafera bez memorisanja ulaznih signala je kolo 74245 (slika 7.10). Ovo kolo pored 8-bitnih priključaka sa jedne (A0-A7) i 8-bitnih priključaka sa druge strane (B0-B7) poseduje i priključak za određivanje smeru (eng. *direction* - DIR) kao i priključak za dovodenje A i B priključaka u stanje visoke impedanse (G). G signal je aktivan na nizak nivo.



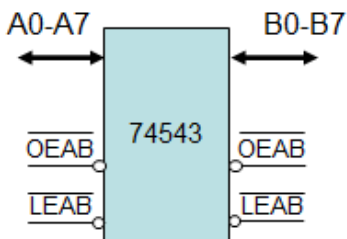
Slika 7.10: Dvosmerni bafer 74245

Tipičan primer 8-bitnog jednosmernog bafera sa memorisanjem je kolo 74373 (slika 7.11). Ovaj bafer je realizovan pomoću leč kola za pamćenje. Ovo kolo pored 8-bitnih ulaznih (D0-D7) i izlaznih (Q0-Q7) priključaka ima i priključak LE (eng. *Latch Enable*) za upis vrednosti sa ulaza u lečeve, kao i priključak OE (eng. *Output Enable*) koji omogućava postavku izlaza u stanje visoke impedanse (aktivan je na nizak nivo). Ukoliko je potrebna varijanta sa flip-flopovima tada treba koristiti kolo 74374.



Slika 7.11: Jednosmerni bafer sa unutrašnjim leč kolima – 74373

Tipičan primer 8-bitnog dvosmernog bafera sa memorisanjem je kolo 74543 (slika 7.12). Ovaj bafer je realizovan pomoću leč kola za pamćenje. Ovo kolo pored dva 8-bitna (A0-A7 i B0-B7) priključka ima i dodatna četiri priključka (pogledati i sliku 7.9). Po 2 za svaki smer (smerovi su od A ka B i od B ka A): LEAB, OEAB i LEBA, OEBA. Svi su aktivni na nizak nivo, a funkcionalnost je ista kao za LE i OE kod 74373.

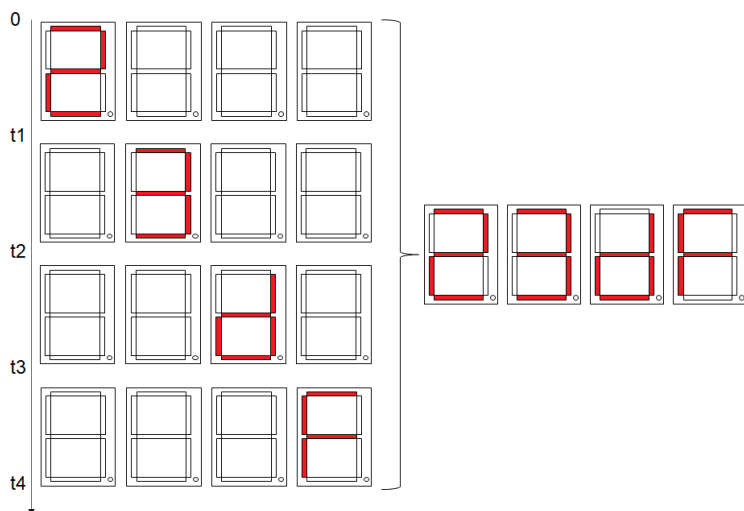


Slika 7.12: Dvosmerni bafer sa unutrašnjim leč kolima – 74543

7.1.3 Multipleksno povezivanje

U slučaju multipleksnog povezivanja se radi o povezivanju uz vremensko multipleksiranje. Tu razlikujemo dva slučaja. U prvom se koristi vremenski multipleks da bi se prikazivale odgovarajuće vrednosti na svetlećim elementima (LED, LED segmenti na 7-segmentnim displejima i sl.). U drugom slučaju se radi o vremenskom multipleksiranju signala na istom portu (npr. pri povezivanju spoljašnjih memorija sa mikrokontrolerom).

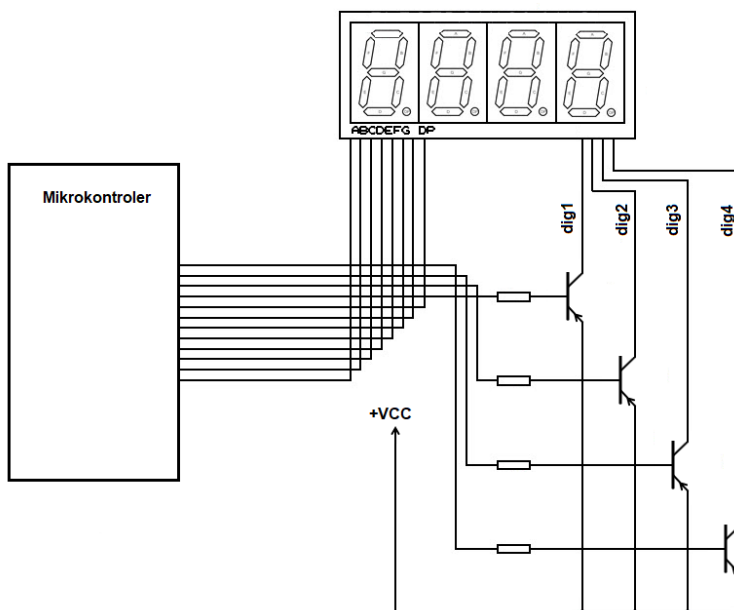
Pogledajmo ponovo slučaj direktnog povezivanja mikrokontrolera i jednog 7-segmentnog displeja prikazanog na slici 7.2. Postavlja se pitanje šta ako je potrebno imati četvorocifreni displej? Očigledno to nije moguće direktnim putem osim ukoliko mikrokontroler ima bar 4 porta. Više displeja moguće je povezati **multipleksno** uz dodavanje određenih komponenata (tranzistora, bafera i sl.). U ovom slučaju broj perifera koje možemo povezati je nešto veći od direktnog, ali je i dalje ograničen brojem slobodnih pinova.



Slika 7.13: Objašnjenje principa vremenskog multipleksiranja

Da bi bolje razumeli princip vremenskog multipleksiranja pogledajmo sliku 7.13. Pretpostavimo da na 4-cifrenom displeju želimo da prikazemo hexadecimalnu vrednost 0x23DF (D se prikazuje malim slovom d da bi se izbeglo preklapanje sa načinom prikaza broja 0). Sa slike se vidi ideja da se u trajanju vremena 0-t1

prikazuje broj 2 na 1. cifri displeja, a da su ostale isključene. Potom se u periodu t1-t2 prikazuje broj 3 na 2. cifri, a ostale su isključene. Slično se od t2-t3 prikazuje broj d na 3. cifri i potom od t3-t4 broj F na 4. cifri dok su ostale cifre isključene. Ukoliko se ovo ponavlja odgovarajućom učestanošću pokazuje se da dobijamo prikaz celog broja kako je prikazano na desnoj strani slike. Razlog je što ljudsko oko za određene učestanosti ne primećuje treperenje koje ovakvim multipleksiranjem postoji. Ukoliko je učestanost premala onda bi treperenje postalo uočljivo. Ukoliko bi učestanost bila prevelika tada bi se dobio efekat da svetle svi segmenti. U praksi se pokazalo da je vreme od nekoliko milisekundi odgovarajuće za smenu cifara u multipleksu za pravilan prikaz.

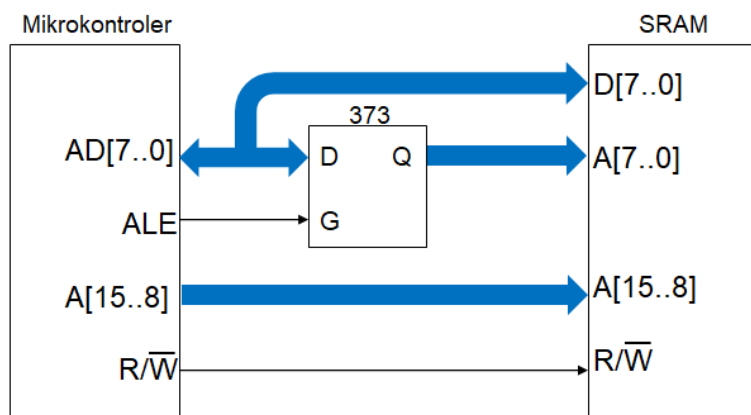


Slika 7.14: Primer povezivanja 4-cifrenog 7-segmentnog displeja sa mikrokontrolerom

Primer hardverske realizacije ovakvog sistema je dat na slici 7.14. Ovde se za uključivanje cifre (a i obezbeđenje odgovarajuće struje kroz segmente) koriste PNP tranzistori koji se uključuju nulom na bazi. Druga rešenja koriste bafere umesto tranzistora. Pri tome je neophodna upotreba displeja sa zajedničkom anodom (npr. Kingbright CA56-12EWA). To znači da se segmenti uključuju nulom (dovođenjem

na katodu LED segmenta). Alternativno je moguće koristiti displej sa zajedničkom katodom i NPN tranzistore, ali je za svaki potreban još po jedan otpornik u kolektorskom kolu za polarizaciju.

Radi objašnjenja primene vremenskog multipleksiranja na signale porta mikrokontrolera posmatračemo slučaj povezivanja mikrokontrolera i spoljašnje memorije prikazan na slici 7.15. Da bi mogla komunikacija između mikrokontrolera i memorije da radi potrebno je da se oforme magistrale podataka i adresna magistrala. Formiranje ovih magistrala vrši se pomoću portova ukoliko je na njima urađen vremenski multipleks adresa i podataka tako što jedan port (na slici označen sa AD[7..0]) služi za generisanje donjih 8 bita adrese (A7..A0) tokom prve polovine mašinskog ciklusa u kome se izvršava pristup spoljašnjoj memoriji. Tokom druge polovine mašinskog ciklusa taj port služi za prijem podatka, u slučaju čitanja spoljašnje memorije, ili slanje podatka u slučaju upisa u spoljašnju memoriju. Da bi ovo bilo ostvarivo potrebno je dodatno leć kolo (npr. tipa 373) koje će pamtit i niži bajt adrese za ovakvo demultipleksiranje adresnih linija sa linijama podataka.



Slika 7.15: Primer multipleksiranja signala adresa i podataka

Pored toga, neophodan je i signal ALE (eng. *address latch enable*) koji se generiše od strane mikrokontrolera da označi trenutak kada leć kolo treba da prihvati tih donjih osam bita adrese (A7..A0). U trenutku kada se aktivira impuls na liniji ALE, leć kolo prosleđuje na svoj izlaz 8 bita koji su u tom trenutku prisutni na njegovom ulazu. Od tog trenutka pa do kraja mašinskog ciklusa, izlazi leć kola zajedno sa linijama drugog porta (na slici označen sa A[15-8]) sačinjavaju adresnu magistralu, dok prvi port sačinjava magistralu podataka. Ovako formirane ma-

gistrale podataka i adresna magistrala se dalje mogu koristiti za povezivanje sa spoljašnjim memorijama uz korišćenje signala za čitanje/upis. Ovakav način povezivanja podržavaju samo neki mikrokontroleri kao što su npr. 8051 i ATmega8515. ATmega328P ne podržava ovakav način povezivanja ali ga je moguće softverski emulirati.

7.1.4 Memorijski mapirano povezivanje

Ukoliko mikrokontroler podržava rad sa spoljašnjim memorijama, tada najčešće ne zahteva pun opseg spoljašnje memorije, bilo da je u pitanju RAM, ROM ili kombinacija. To znači da veliki deo memorijskog prostora ostaje neiskorišćen. Ne samo da deo memorijskog prostora ostaje neiskorišćen, već su nožice portova koji nisu važne za formiranje adrese neupotrebljive. Na primer, ako nam je potrebna eksterna memorija od 1 kB, za adresiranje je potrebno 10 linija od ukupno 16. Ostale nožice, iako ne učestvuju u formiranju adresa, kod nekih mikrokontrolera ne mogu biti iskorišćene za vezivanje periferija, jer to ne dozvoljava mikrokontroler (npr. x51).

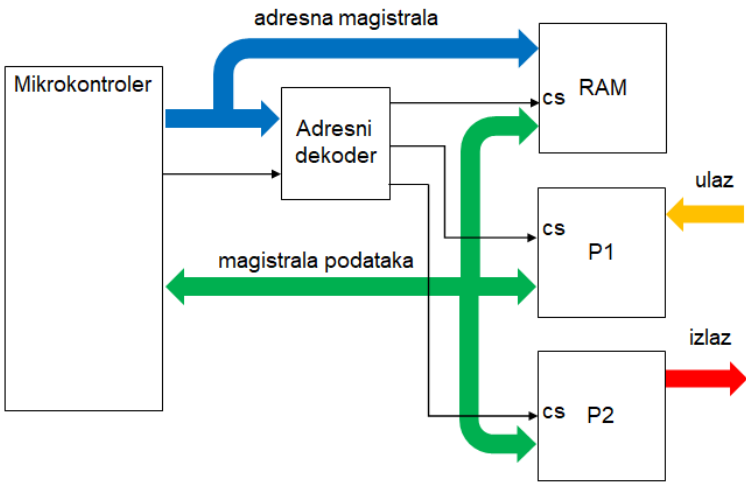
Ipak, moguće je povezati periferne uređaje ako ih posmatramo kao delove memorijskog adresnog prostora mikrokontrolera. Na ovaj način pristupamo periferiji kao da je memorija (nešto se upisuje ili čita). Kombinacijom kola za pamćenje stanja i adresnog dekodera moguće je povezati veliki broj periferija. Radi boljeg razumevanja memorijskog mapiranja, na primeru sa slike 7.16 je prikazano povezivanje spoljašnje memorije, jedne ulazne periferije P1 (npr. 8 prekidača) i jedne izlazne periferije P2 (npr. 8 LED).

Sa slike se vidi da je neophodna upotreba adresnog dekodera da bi se izvršilo nepotpuno dekodovanje (o čemu je bilo reči u poglavlju ?? o memorijskom podsi-stemu) i aktivirao odgovarajući CS (eng. *Chip Select*) signal za aktiviranje odgovarajuće periferije.

Ono što se veoma često koristi kod većine mikrokontrolera je da se memorijski mapiraju određene skupine registara. Primer kod mikrokontrolera ATmega328P, a u vezi registara za rad sa portovima (DDRx, PORTx, PINx; x - B, C ili D) prikazan je na slici 7.17.

7.1.5 Povezivanje uz konverzije

Da bi se prevazišao manjak pinova mikrokontrolera u slučaju direktnog povezivanja nekada se pristupa i upotrebi kola za serijsko-paralelnu, paralelno-serijsku konverziju ili neku drugu konverziju. O analogno-digitalnoj konverziji kod Arduina je bilo reči u prethodnom poglavlju.



Slika 7.16: Memorijsko mapiranje periferija

...

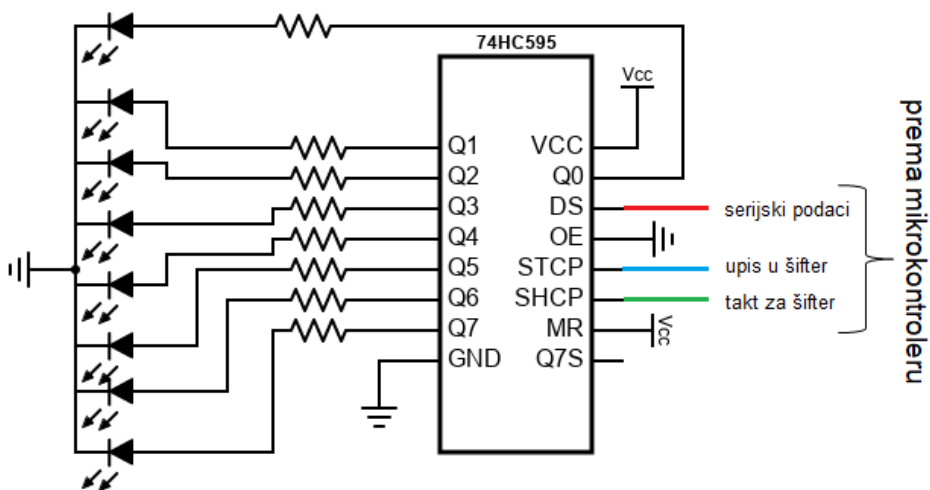
0x0C (0x2C)	Rezervisano	-	-	-	-	-	-	-	-
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0
0x0A (0x2A)	DDRD	DD07	DD06	DD05	DD04	DD03	DD02	DD01	DD00
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0
0x08 (0x28)	PORTC	-	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0
0x07 (0x27)	DDRC	-	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0
0x06 (0x26)	PINC	-	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
0x04 (0x24)	DDRB	DD07	DD06	DD05	DD04	DD03	DD02	DD01	DD00
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0
0x02 (0x22)	Rezervisano	-	-	-	-	-	-	-	-
0x01 (0x21)	Rezervisano	-	-	-	-	-	-	-	-
0x00 (0x20)	Rezervisano	-	-	-	-	-	-	-	-

Slika 7.17: Memorijsko mapiranje registara

Povezivanje uz serijsko-paralelnu konverziju

Na slici 7.18 prikazan je tipičan primer povezivanja uz serijsko-paralelnu konverziju (eng. *Serial In Parallel Out* - SIPO) konverzije gde se sa mikrokontrolera serijski šalju podaci koje treba prikazati na izlaznoj periferiji npr. 8 LED. Da bi to moglo da se uradi potrebno je koristiti dodatno elektronsko kolo za SIPO konverziju - 74HC595. Na ovaj način se umesto 8 linija za direktno povezivanje koristi

samo 3. Na slici 7.19 je prikazan unutrašnji sadržaj kola za konverziju koje sadrži 8-bitni pomerački registar, 8-bitni registar i trostatičke izlaze (izlazi koji imaju mogućnost visoke impedanse).



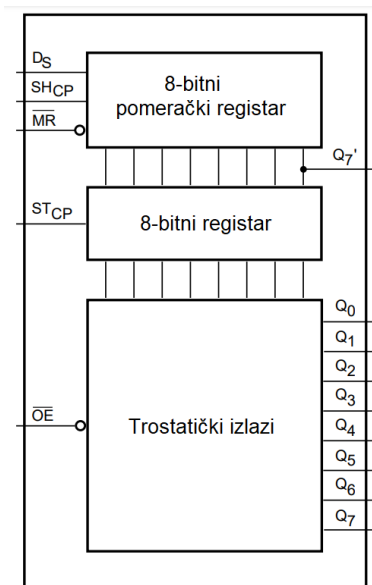
Slika 7.18: Primer kola sa SIPO konverzijom

Princip rada kola za SIPO konverziju je sledeći. Serijski podaci se dovode na DS ulaz. Na SHCP ulaz je potrebno dovoditi takt koji omogućava realizaciju pomeranja 8-bitnog pomeračkog registra. Kada se na STCP pojavi rastuća ivica vrši se upis sa izlaza pomeračkog registra ("šiftera") u 8-bitni registar koji se vodi na trostatičke izlaze. Ukoliko je OE ulaz na nuli onda su izlazi aktivni. Za detaljne vremenske dijagrame i ostale podatke najbolje je pogledati šta je proizvođač dao (npr. [36]).

Povezivanje uz paralelno-serijsku konverziju

Na slici 7.20 prikazan je tipičan primer povezivanja uz paralelno-serijsku konverziju (eng. *Parallel In Serial Out* - PISO) gde se podaci serijski šalju prema mikrokontroleru sa neke paralelne ulazne periferije npr. 8 prekidača. Da bi to moglo da se uradi potrebno je koristiti dodatno elektronsko kolo za PISO konverziju - 74HC165 ili CD4021.

Princip rada kola za PISO konverziju koje se sastoji od jednog 8-bitnog pome-

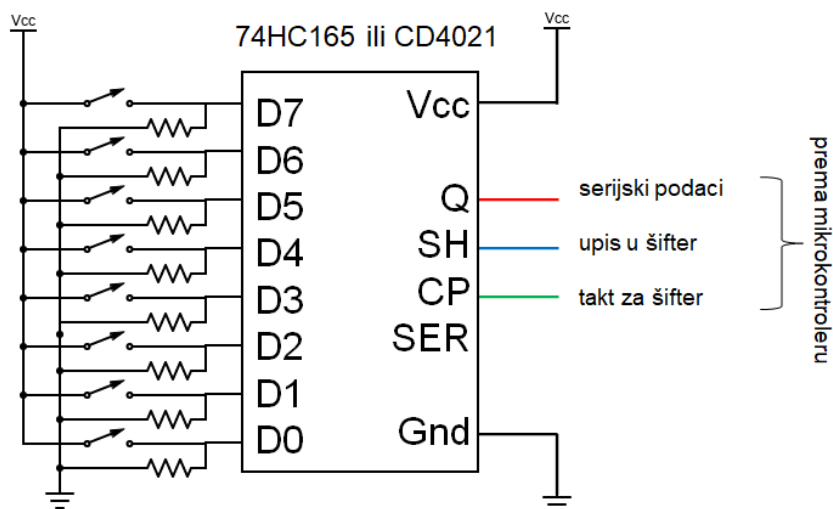


Slika 7.19: Unutrašnja struktura kola za SIPO konverziju

račkog registra je sledeći. Paralelni signali se dovode na ulaze D0-D7 (kod pojedinih proizvođača su drugačije slovne oznake), u našem primeru to su signali sa prekidača. Paralelni upis ovih signala u 8-bitni pomerački registar se vrši negativnim impulsom na SH ulazu. Nakon toga na svaku rastuću ivicu na CP signalu se po jedan bit serijski izbacuje na Q izlazu prema mikrokontroleru. Za detaljne vremenske dijagrame i ostale podatke najbolje je pogledati šta je proizvođač dao (npr. [37]). Ovaj način se koristi za čitanje stanja 8 prekidača i na jednom od šilдова na vežbama.

Povezivanje uz analogno-digitalnu konverziju

Ukoliko stanje periferija može na određeni način da bude određeno preko različitih vrednosti napona tada je moguće povezati ih na analogni ulaz mikrokontrolera i putem konverzije analogne u digitalnu vrednost odrediti status periferije. Pretpostavke su da mikrokontroler ima bar jedan analogni ulaz kao i analogno-digitalni konvertor (npr. ATmega328P ih ima). Jedan zanimljiv i efikasan način

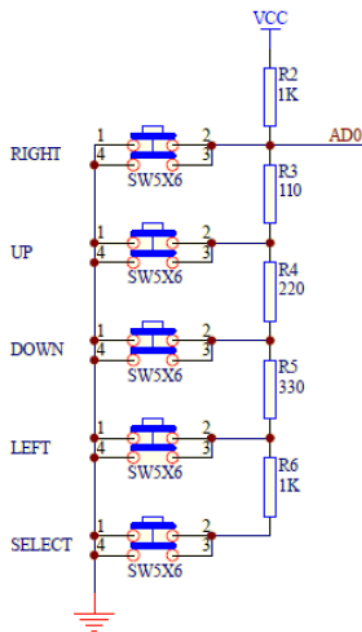


Slika 7.20: Primer kola sa PISO konverzijom

povezivanja periferija uz konverzije je povezivanje više tastera preko razdelnika napona na analogni ulaz mikrokontrolera (ako postoji takav ulaz na njemu). Ovakav primer je prikazan na slici 7.21 i on se koristi na vežbama na šildu koji sadrži LCD i tastere. Osnovna ideja je da pritisak tastera menja faktor razdelnika napona i na analogni ulaz mikrokontrolera (AD0) se vodi analogna vrednost koja zavisi od toga koji je taster pritisnut. Na mikrokontroleru se potom ta vrednost konvertuje u digitalnu i onda se na osnovu nje može znati koji je taster pritisnut. Na ovaj način je 5 tastera povezano preko 1 pina.

7.1.6 Povezivanje na bazi standardnih protokola

Pod pojmom standardnih protokola se misli npr. na standardne serijske protokole tipa I2C, SPI, RS232 itd. O osnovnoj serijskoj komunikaciji će biti više reči u poglavlju ?? o serijskoj komunikaciji, a neki od ovih protokola će se detaljno raditi na višim godinama. Danas mnogo periferija podržava serijski način povezivanja sa mikrokontrolerima.

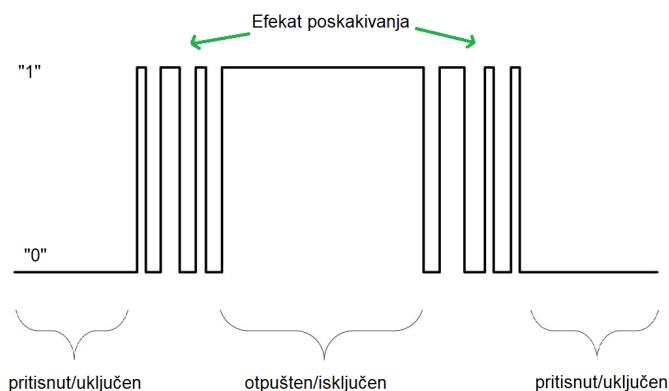


Slika 7.21: Očitavanje tastera preko razdelnika napona

7.2 Problem poskakivanja

Prilikom rada sa tasterima i prekidačima se javlja problem poskakivanja (eng. *bouncing*) kontakata. Ovaj problem se javlja prilikom promena tekućeg stanja tastera/prekidača iz otpušten/isključen u pritisnut/uključen i obrnuto. Grubi prikaz ove pojave je dat na slici 7.22. Vremenski posmatrano imamo duži period stanja kada su tasteri/prekidači otpušteni/isključeni i kada su pritisnuti/uključeni. Prilikom prelaza iz ovih stanja u ono drugo javlja se problem poskakivanja koji se oslikava pojavom lažnih nula ili jedinica što u embeded sistemima može izazvati probleme u radu jer usled poskakivanja možemo imati mnogo pogrešnih očitavanja stanja tastera/prekidača.

Problem poskakivanja se može rešiti (eng. *debouncing*) hardverskim ili softverskim tehnikama. Softverska tehnika za rešavanje problema poskakivanja kontakata se svodi na tehniku "sačekaj i vidi". Dakle, nakon detekcije tranzicije (taster pritisnut) očitavanjem pina na koji je povezan taster/prekidač potrebno je generisati



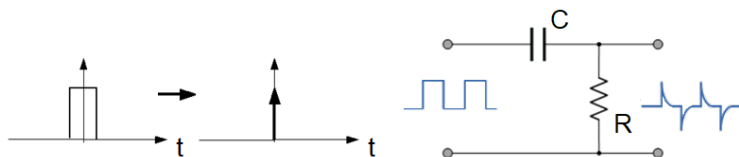
Slika 7.22: Ilustracija problema poskakivanja kontakata

odgovarajuće kašnjenje, obično reda desetak milisekundi, i potom proveriti da li je očitano stanje isto kao prethodno očitano. Ako jeste to je znak da se period poskakivanja završio i možemo reći da se pritisak desio. U suprotnom preći na početak i na čekanje pritiska. Hardverske tehnike uključuju dodatne hardverske resurse koji rešavaju ovaj problem. Više reči o tome će biti na višim godinama.

7.3 Softversko diferenciranje

Prilikom rada sa tasterima se pojavljuje potreba za tehnikom 'softverskog diferenciranja'. Ovo je potrebno kada želimo da jedan pritisak na taster prouzrokuje jedan, a ne mnogo događaja. Pre svega pritisak tastera je događaj koji traje obično nekoliko stotina milisekundi. Ovo je iz ugla čitanja mikrokontrolera veoma dugačko. Na primer, ako taster služi da aktivira operaciju sabiranja, i pretpostavimo da je trajanje pritiska tastera pola sekunde, a mikrokontroler čita stanje tastera na svaku milisekundu tada bismo imali 500 aktivacija sabiranja, a želimo samo jednu.

Ova tehnika se zove softversko diferenciranje jer podseća na operaciju diferenciranja, konkretno na efekte CR kola koje ima jako malu vremensku konstantu. Na levoj strani slike 7.23 je prikazan idealizovan signal trajanja pritiska tastera i ono što želimo da dobijemo - delta impuls. Sa desne strane slike je prikazano CR kolo i signali na ulazu i izlazu. Dakle od povorke impulsa značajnog trajanja da dobijemo povorku delta impulsa.



Slika 7.23: Ilustracija analogije sa diferencijatorom

Ovo je moguće realizovati softverski uvođenjem dodatne promenljive (dozvola) i praćenjem stanja pritisnutosti tastera (promenljiva taster) što je prikazano programskim kôdom 7.24. Ukoliko pretpostavimo da se pritisak tastera očitava kao logička jedinica tada možemo videti da se u prvi if uslov ulazi samo kada je pritisnut taster i ukoliko je dozvola = 1. Unutar uslova se odmah promenljiva dozvola resetuje i uradi se nešto korisno (npr. sabiranje). U drugi uslov nije moguće ući jer taster još nije otpušten. Setimo se da je pritisak tastera dugačak događaj za mikrokontroler! Kada zamislimo da se ovo izvršava u petlji vidimo da neko vreme ni jedan uslov nije zadovoljen. Prvi zbog dozvola = 0, a drugi jer taster još nije otpušten. Kada se taster otpusti, tada ponovo postavljamo dozvola = 1. Sada kada taster ponovo bude pritisnut prvi uslov će biti zadovoljen i **jedan** ulazak je moguć. Na ovaj način smo dugačko trajanje pritisnutosti tastera skratili na jedan događaj.

```
if (taster && dozvola)
{
    dozvola = 0;

    // radi nešto korisno
}

if (!taster) dozvola = 1;
```

Slika 7.24: Programska realizacija softverskog diferenciranja

7.4 Pitanja

1. Nabrojati i objasniti koji načini povezivanja periferija postoje?
2. Navesti primere, osnovne karakteristike, prednosti i mane kod direktnog po-

vezivanja periferija.

3. Koje vrste postoje i čemu služe baferi kod baferovanog povezivanja periferija? Objasniti.
4. Objasniti podelu bafera.
5. Nacrtati blok šemu bafera 74245, objasniti svrhu i način funkcionisanja.
6. Nacrtati blok šemu bafera 74373, objasniti svrhu i način funkcionisanja.
7. Nacrtati blok šemu bafera 74543, objasniti svrhu i način funkcionisanja.
8. Objasniti multipleksno povezivanje periferija i navesti primer.
9. Kako se formiraju spoljašnja magistrala podataka i adresna magistrala kod mikrokontrolera koji imaju multipleksiranje adresa i podataka? Nacrtati i objasniti.
10. Objasniti principe memorijskog mapiranja periferija.
11. Šta znači kada se kaže da su registri memorijski mapirani?
12. Povezivanje uz serijsko-paralelnu konverziju (SIPO). Navesti primer i objasniti.
13. Povezivanje uz paralelno-serijsku konverziju (PISO). Navesti primer i objasniti.
14. Na koji način je moguće povezati više tastera na jedan ulazni pin mikrokontrolera? Objasniti.
15. Objasniti problem poskakivanja tastera i kako se rešava.
16. Objasniti šta je, čemu služi "softversko diferenciranje" i navesti kako se realizuje.

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] B. B. Brey, *Intel Microprocessors*, 8th Edition, Pearson, 2009.
- [6] S. Harris, D. Hariss, *Digital Design and Computer Architecture*, Arm edition, Morgan Kaufmann, 2015.
- [7] 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.
- [8] M. M. Mano, C. R. Kime, T. Martin, *Logic and Computer Design Fundamentals*, Fifth edition, Pearson, 2015.
- [9] R. S. Sandige, M. L. Sandige, *Fundamentals of Digital and Computer Design with VHDL*, McGraw Hill, 2012.
- [10] L. Null, J. Lobur *Essentials of Computer Organization and Architecture*, 5th Edition, Jones and Barlett Publishers, 2018.

- [11] E.O. Hwang, Digital Logic and Microprocessor Design With VHDL with Interfacing, Cengage Learning, Second Edition, 2017.
- [12] I. Mezei "Evolution of an educational microprocessor", Computer Applications in Engineering Education, 28(5), Wiley, pp. 1265-1277, 2020.
- [13] W. Stallings, Computer organization and architecture, 9th Edition, Pearson, 2013.
- [14] M. Abd-El-Barr, H. El-Rewini, Fundamentals Of Computer Organization And Architecture, John Wiley and Sons, Inc., 2005.
- [15] C. Hamacher, Z. Vranesic, S. Zaky, N. Manjikian, Computer Organization and Embedded Systems, 6th Edition, McGraw-Hill, 2012.
- [16] Manuel Jiménez, Rogelio Palomera, Isidoro Couvertier, Introduction to Embedded Systems, Springer, 2014.
- [17] 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>
- [18] D. A. Patterson, J. L. Hennessy, Computer Organization and Design: The Hardware/Software Interface, Fourth Edition, Morgan Kaufmann, 2011.
- [19] J.L. Gustafson, Moore's Law. In: D. Padua D. (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [20] S.A. McKee, R.W. Wisniewski, Memory Wall. In: D. Padua (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [21] P. Bose, Power Wall. In: D. Padua (eds) Encyclopedia of Parallel Computing. Springer, 2011.
- [22] 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.
- [23] W.W. Royce, "Managing the Development of Large Software Systems: Concepts and Techniques", IEEE Wescon Proc., Aug. 1970.
- [24] C.G. Bell, A. Newell, Computer structures: Readings and Examples, McGraw-Hill, 1971.

- [25] S. Brown, Z. Vranesic, Fundamentals of Digital Logic with VHDL Design, McGraw Hill, 2009.
- [26] D. L. Perry, VHDL: Programming by Example, Fourth edition, McGraw Hill, 2002.
- [27] L. Null, J. Lobur, Essentials of Computer Organization and Architecture, Jones and Bartlet Learning, 3rd Ed., 2010.
- [28] 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.
- [29] S. L. Harris, D. Harris, Digital Design and Computer Architecture, RISC-V Edition, Morgan Kaufman, 1st edition, 2021.
- [30] 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.
- [31] D. Kushner, "The making of arduino", IEEE Spectrum 26, 2011.
- [32] M. Banzi, "How Arduino is open-sourcing imagination", TEDtalk, Scotland, 2012. <https://www.youtube.com/watch?v=UoBUXOOdLXY> (datum pristupa: 31.01.2022.)
- [33] Sparkfun, "Arduino shields v2", <https://learn.sparkfun.com/tutorials/arduino-shields-v2>, (datum pristupa: 01.02.2022.)
- [34] Arduino Uno Rev3 schematic, https://www.arduino.cc/en/uploads/Main/Arduino_Uno_Rev3-schematic.pdf (datum pristupa: 02.02.2022.)
- [35] Arduino software, <https://www.arduino.cc/en/Main/Software> (datum pristupa: 02.02.2022.)
- [36] 74HC595 datasheet, Philips, 1998, <https://www.arduino.cc/en/uploads/Tutorial/595datasheet.pdf> (datum pristupa: 03.03.2022.)
- [37] 74HC165 datasheet, Texas Instruments, 2015 <https://www.ti.com/lit/ds/symlink/sn74hc165.pdf> (datum pristupa: 03.03.2022.)
- [38] ATmega328 datasheet, Atmel, 2015, https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf (datum pristupa: 15.03.2022.)

- [39] Tutorials 83, Fundamentals of RS-232 serial communications, Analog Devices, 2001. <https://www.maximintegrated.com/en/design/technical-documents/tutorials/8/83.html> (datum pristupa: 31.03.2022.)