Povezovanje mikrokontrolerjev iz družine 8051 z elektronskimi komponentami

Čeprav so zelo kompleksen in sofisticiran izdelek, mikrokontrolerji sami po sebi ne veljajo veliko – koristni postanejo šele, ko jih povežemo z drugimi elektronskimi komponentami in vezji. Za to so nam na razpolago vhodno-izhodni priključki (vrata, porti), ki jih ima večina mikrokontrolerjev vsaj okoli 10. Pri "večjih" mikrokontrolerjih bo število V/I priključkov 30 ali več; velja pravilo - čim več jih je, tem boljše je! V tem prispevku bom pokazal, kako različne komponente in vezja povezujemo z mikrokontrolerjem. Večino tega smo že objavili v različnih projektih na straneh Sveta elektronike, tako da je ta prispevek svojevrstna rekapitulacija in opomnik za izkušenejše, obenem pa tudi skup koristnih informacij na enem mestu za začetnike.

Vezja so prilagojena mikrokontrolerjem iz družine 8051; mednje spadata tudi AT89C2051 in AT89C51, torej ravno tisti mikrokontrolerji, ki jih najpogosteje uporabljamo v naših projektih. Mikrokontrolerji iz družine AVR so po zgradbi (vsaj kar se tiče portov) podobni tistim iz družine 8051, in ker sta si BascomLT in BascomAVR prav tako zelo podobna, bodo primeri vezij in programov veljali tudi za AVR-je. Druge družine mikrokontrolerjev, kot so npr. popularni PIC-i, imajo svoje lastne značilnosti in bo predložene sheme potrebno bolj ali manj modificirati; spremljajoči programi prav tako lahko služijo samo kot ilustracija in niso direktno uporabni.

Osnovno o vhodno-izhodnih priključkih

AT89C51 ima, kot tipični predstavnik mikrokontrolerjev iz družine 8051, štiri dvosmerne 8-bitne porte, P0, P1, P2 in P3. Posamezni priključki (pini) porta so označeni s števili 0-7 (P1.0, P3.7 ipd.). Znotraj mikrokontrolerja so priključki izvedeni "nesimetrično". Obstaja namreč izhodni MOS tranzistor, ki lahko priključek pripelje v stanje "0" in pri tem 'požira' tok do maksimalno 15 mA, za stanje "1" pa je zadolžen sorazmerno velik *pull-up* upor, preko katerega lahko steče tok okoli 10 µA. Zaradi takšne izvedbe je lahko vsak priključek istočasno in vhod in izhod:

- če ga uporabljamo kot izhod, ga v želeno stanje postavimo direktno iz programa, npr.:

```
Set P1.0
'postavi P1.0 v stanje "1"
Reset P1.7
'postavi P1.7 v stanje "0"
P3 = 0
'postavi vseh 8 bitov
'porta P3 v stanje "0"
```

- če ga uporabljamo kot vhod, je najprej potrebno določeni priključek postaviti v stanje "1" (v tem stanju so dovoljeni kratki stiki z maso - stanje "0", nakar lahko iz programa odčitamo, v katerem stanju se priključek dejansko nahaja.

Port P0 je narejen brez internih *pull-up* uporov, zato je potrebno, če na posameznem pinu tega porta želimo dobiti logično "1" ali ga uporabiti kot vhod, to realizirati od zunaj. Upore se spaja med posameznimi pini porta P0 in + polom napajalne napetosti in imajo tipično vrednost v razponu 4,7 - 47 kohmov. Brez zunanjih *pull-up* uporov, pini porta P0, ki jih postavimo v stanje "1", pravzaprav izklopijo izhodni tranzistor in preidejo v stanje visoke vhodne upornosti (*3-state*).

Večina V/I priključkov lahko poleg funkcij vhoda in izhoda opravlja tudi različne druge oziroma alternativne funkcije. Tukaj bomo samo omenili, da se porta P0 in P2, če ju uporabljamo za naslavljanje zunanjega pomnilnika, rekonfigurirata kot *push-pull* izhodna stopnja in ju v tem primeru ne moremo uporabljati kot vhoda na prej opisani način.

AT89C2051 je tipični predstavnik mikrokontrolerjev iz družine 8051 v manjšem, 20-pinskem ohišju. AT89C2051 vsebuje samo dva porta, P1 in P3, ki sta identična enakoimenovanim portom mikrokontrolerja AT89C51, za katerega velja vse prej našteto. Pripomnimo še, da sta pina P1.0 in P1.1 izvedena brez internih *pull-up* uporov, ki ju je zato potrebno dodati od zunaj. Pin 3.6 nima zunanjega priključka. Maksimalni dovoljeni izhodni tok za posamezni pin v stanju "0" tukaj znaša 25 mA.

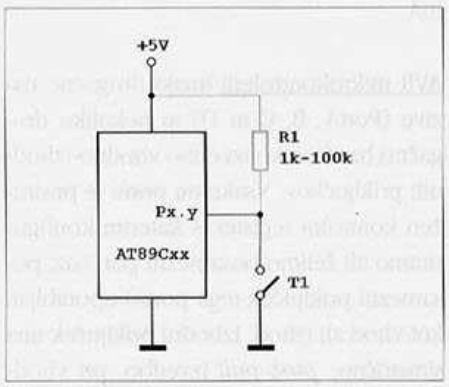
AVR mikrokontrolerji imajo drugačne nazive (PortA, B, C in D) in nekoliko drugačno hardversko izvedbo vhodno-izhodnih priključkov. Vsakemu portu je pridružen kontrolni register, s katerim konfiguriramo ali želimo posamezni port (oz. posamezni priključek tega porta) uporabljati kot vhod ali izhod. Izhodni priključek ima simetrično, *push-pull* izvedbo, pri vhodnem pa je možno po potrebi izključiti ali vključiti interni *pull-up* upor. Poglejmo si nekaj primerov, kako se to naredi v BascomAVR-ju:

```
Config Portb = Output
'Port B je izhodni
Config Pinb.0 = Input
'Pin B.0 je vhodni
Ddrb = &B11110000
'Direktni vpis v kontrolni
'register; B.0-B.3 so vhodi
'B.4-B.7 so izhodi
```

Šele ko so konfigurirani kot vhodi oziroma izhodi, lahko priključke AVR mikrokontrolerja uporabljamo na način kot je prej opisano. Tipični tok, ki ga premore izhodni priključek, znaša 20 mA.
Podrobnejše podatke o portih lahko najdete v *Help*-u programov BascomLT, Bascom8051 in BascomAVR ter v tehničnih specifikacijah določenega mikrokontrolerja. Vezave, ki sledijo, se nanašajo na mikrokontrolerje iz družine 8051, programski primeri pa so napisani v BascomLT-ju. Idejno se lahko vse primere uporabi tudi z AVR mikrokontrolerji, vendar je potrebno paziti na posebnosti (npr. konfiguriranje V/I pinov pred samo uporabo).

Osnovna vhodna vezava

Tipke, stikala in kratkostičnike je vedno potrebno vezati med vhodnimi pini (Px.y na sliki 1 označuje katerikoli V/I priključek) in maso. Stikalo, vezano med vhodnim pinom in + polom napajanja, ne bo dalo želenega efekta, lahko pa tudi uniči mikrokontroler! *Pull-up* upor R1 je obvezen samo za priključke brez internega *pull-up* upora (npr. P1.0 in P1.1 pri AT89C2051); zaradi manjše porabe je priporočljivo uporabljati upore z upornostjo 10 kohmov ali več.



Slika 1a: Osnovna vhodna vezava; tipko je potrebno vezati tako, da ustvarja kratek stik z maso.

Primer odčitavanja stanja stikala vezanega na P1.7:

```
Set P1.7
'P1.7 bo vhod
If P1.7 = 0 Then
'Če je tipka pritisnjena...
   ...
End If
```

Zadostuje, da pin definiramo za vhodnega le enkrat v programu, oziroma tega ni potrebno ponavljati pred ali po vsakem branju. Težava, ki lahko nastane pri uporabi mehanskih stikal, je iznihavanje kontaktov, zaradi česar lahko program en pritisk in sproščanje stikala ali tipke zazna kot serijo pritiskov. Tukaj si lahko pomagamo, če uporabimo ukaz Debounce:

```
Debounce P1.7 , 0 , Pritisnje-
na_P1.7, Sub
```

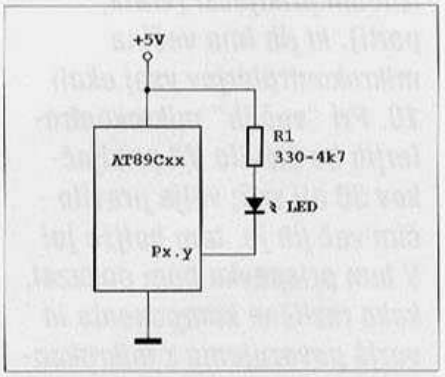
Za razliko od navadnega If bo Debounce znotraj 25 ms dvakrat preveril ali je tipka P1.7 pritisnjena in šele zatem izvršil podprogram *Pritisnjena_P1.7*. Če se nahaja znotraj zanke, bo Debounce nenehno preverjal stanje vhodnega pina P1.7 in bo šele, ko bo ugotovil, da je bila navedena tipka sproščena in nato ponovno pritisnjena, iniciral izvršitev podprograma *Pritisnjena_P1.7*. Tako nam lahko ukaz Debounce privarčuje veliko programske kode.
Včasih moramo v programu istočasno odčitati stanje več stikal. V naslednjem primeru smo predpostavili, da so stikala vezana na priključke P1.7 - P1.4:

```
Dim Pom As Byte
...
P1 = P1 Or &B11110000
'P1.7-P1.4 so vhodi
Pom = P1 And &B11110000
'Izloči P1.7-P1.4
IF Pom = &B10100000 Then
'Če je pritisnjena
...
'ustrezna kombinacija
...
```

Tukaj ni nujno, da so uporabljeni vhodni priključki "en za drugim", je pa zaradi enostavnosti zaželeno, da pripadajo istemu portu.

Osnovna izhodna vezava

Glede na izvedbo izhodnega vezja se bremena pri mikrokontrolerjih iz družine 8051 vežejo vedno med izhodnim priključkom in + polom napajanja. Poudarimo, da teče tok skozi breme, ko je izhod v stanju "0" in ne v stanju "1", kot bi morda pričakovali (to je v nesoglasju z Bascomovim simulatorjem, kjer LED dioda sveti v stanju "1"). Upornost bremena mora biti toliksna, da skozi izhodni pin ne teče tok večji od 25 mA (AT89C2051) oziroma 15 mA (AT89C51). Iz varnostnih razlogov je potrebno tok omejiti na 10-12 mA. To bo dovolj, da bo LED dioda svetila s polnim sijajem, dovolj bo tudi za vklop miniaturnih relejev, vendar ne bo dovolj za vklop klasičnih relejev, motorjev ali drugih večjih porabnikov.



Slika 1b: Osnovna izhodna vezava; maksimalni dovoljeni tok za posamezni pin je okoli 10 mA.

Tu si lahko pomagamo, če povežemo nekaj izhodnih priključkov paralelno in nato nanje vežemo breme, vsekakor pa zopet proti + polu napajanja. V primeru, ki sledi, smo to naredili s pini P1.3 do P1.0, zaradi česar lahko vklapljamo porabnike s porabo do 50 in več mA:

```
P1 = P1 And &B11110000
'P1.3-P1.0 = 0, vključeno
P1 = P1 Or &B00001111
'P1.3-P1.0 = 1, izključeno
```

Tukaj zaradi narave ukazov And in Or preostali pini porta P1 niso spremenjeni in se jih lahko uporabi bodisi kot vhode ali kot izhode. Kaj se bo zgodilo, če vseh paralelno vezanih izhodov ne postavimo istočasno v isto stanje? V tem primeru bo porabnik vključen, če je vsaj en izmed izhodov v stanju "0", vendar bi se znalo zgoditi, da bo to povzročilo preobremenitev

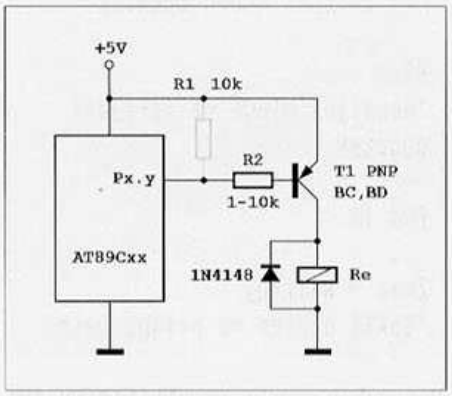
tega izhoda; dokler je tok porabnika znotraj dovoljenih meja, se ne bo zgodilo nič slabega.

Izhodna vezava s tranzistorji

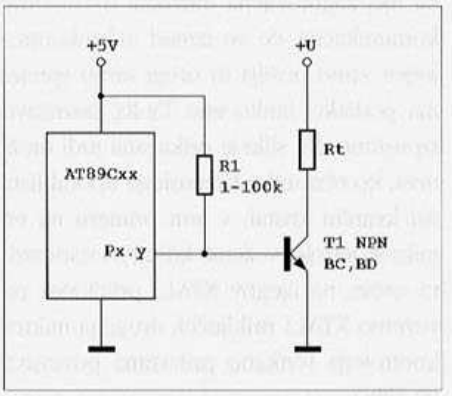
Izhode mikrokontrolerja lahko "ojačamo" z uporabo različnih tipov tranzistorjev. Porabnik na sliki 2a je rele, na vseh ostalih slikah pa je prikazan posplošeno kot upor Rt. Opazimo, da je porabnik na slikah 2a in 2c vključen, ko je izhod v stanju "0", na slikah 2b in 2d pa, ko je izhod v stanju "1". Vezavi na slikah 2b in 2d sta zanimivi tudi po tem, da se lahko napajalna napetost porabnika +U razlikuje od napajalne napetosti mikrokontrolerja (tipično +5V). Zunanji *pull-up* upor je nujen, samo kadar uporabljamo NPN tranzistor, njegovo upornost pa je potrebno določiti glede na potrebni bazni tok. Npr. če skozi porabnik teče tok 100 mA in če je tokovno ojačenje tranzistorja 200, bomo potrebovali bazni tok 0,5 mA ali večji. Če izberemo R1 = 4,7 kohmov, bo bazni tok znašal nekoliko manj kot 1 mA – zagotovili smo si tudi majhno rezervo. Podoben izračun je potreben tudi za R2 v vezavi s PNP tranzistorjem.

Za večje tokove lahko uporabimo tudi močnejše BD tranzistorje. V tem primeru vsekakor priporočam uporabo Darlington tranzistorjev z velikim tokovnim ojačenjem, da potrebni bazni tok ne bi presegal 5-10 mA.

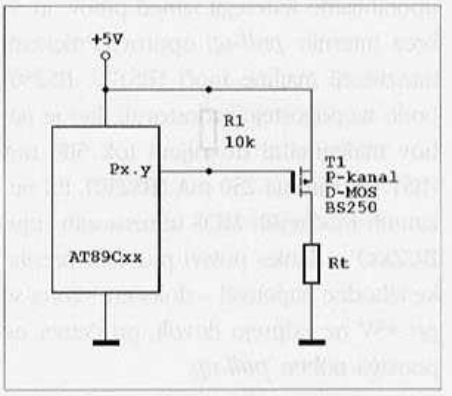
Za razliko od bipolarnih tranzistorjev so MOS tranzistorji (2c, 2d) napetostno krmiljeni in so bolj "združljivi" z izhodi mikrokontrolerja. Tukaj praviloma ne potrebujemo *pull-up* uporov R1, razen če ne



Slika 2a: Način povezovanja PNP tranzistorjev na izhodni port.



Slika 2b: Način povezovanja NPN tranzistorjev na izhodni port.



Slika 2c: Način povezovanja P-kanalnega D-MOS tranzistorja na izhodni port.

Razpis za državno tekmovanje v konstrukciji in vožnji z mobilnimi roboti RoboT 2002

Inštituta za avtomatiko in robotiko FERI, Zveza za tehnično kulturo Slovenije in revija Svet elektronike v sodelovanju s sponzorji razpisujemo tradicionalno tekmovanje iz področja mobilne robotike*:

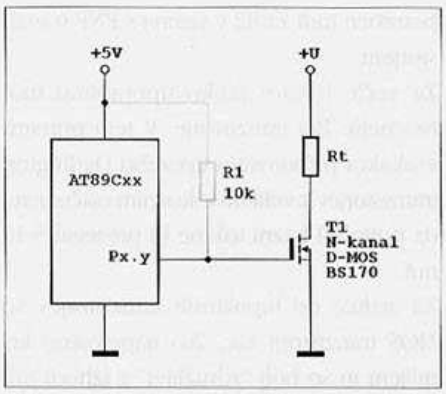
A) Tretje državno študentsko tekmovanje z mobilnimi roboti
B) Drugo državno dijaško tekmovanje z mobilnimi roboti

Prijave zbiramo **do 21. 12. 2001**, tekmovanja bodo izvedena pomladi 2002 (finale predvidoma 9. 5. 2002).
Najuspešnejšim trem tekmovalcem iz vsake kategorije bodo v finalu podeljene nagrade:

- Nagradni sklad v višini 100.000 SIT;
- Podjetje **COMTRON** prispeva praktične nagrade za A) kategorijo (računalniške komponente);
- Revija **Svet elektronike** prispeva praktične nagrade za B) kategorijo;
- Podjetje **RS Components** bo sponzoriralo enega tekmovalca in omogočilo nabavo komponent po ugodnejših cenah vsem tekmovalcem.

Podrobnejše informacije in **prijavnice** za tekmovanje: http://www.ro.feri.uni-mb.si/tekma/
Organizator: Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Inštitut za robotiko, Smetanova 17, 2000 Maribor v sodelovanju z **Zvezo za tehnično kulturo Slovenije** in revijo **Svet elektronike**.
Vodja tekmovanja: mag. Janez Pogorelc, univ. dipl. inž. el., janez.pogorelc@uni-mb.si, tel. 02 220 7304

*Namig za izdelavo mobilnega robota je v članku MIMOR v 69. številki (oktober 2000) in v članku FIRE v 78. številki (julij/avgust 2001) Sveta elektronike.

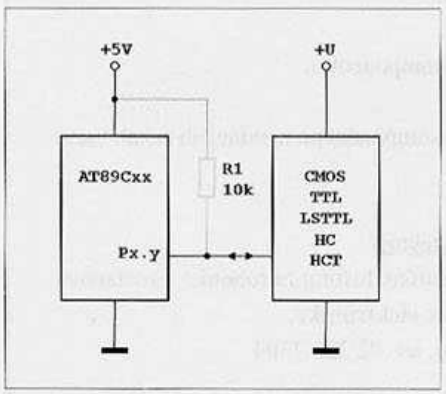


Slika 2d: Način povezovanja N-kanalnega D-MOS tranzistorja na izhodni port.

uporabljamo katerega izmed pinov, ki so brez internih *pull-up* uporov. Prikazani tranzistorji majhne moči (BS170, BS250) bodo najpogosteje zadostovali, ker je njihov maksimalni dovoljeni tok 500 mA (BS170) oziroma 250 mA (BS250). Pri nekaterih močnejših MOS tranzistorjih (npr. BUZxx) se lahko pojavi problem prenizke izhodne napetosti – določeni vzorci se pri +5V ne odprejo dovolj, pri čemer ne pomaga noben *pull-up*.

Vezava mikrokontrolerja z logičnimi vezji

Logična vezja vseh popularnih družin lahko direktno vežemo z mikrokontrolerjem. Če je mikrokontroler izhod, lahko vsak pin direktno krmili do 4 TTL vhode oziroma vsaj 10 LSTTL, HCT ali CMOS vhodov. Zaradi hitrejšega odziva lahko pri CMOS in HC vezjih postavimo *pull-up* upor R1 tipične upornosti okoli 10 kohmov. Pri vseh družinah logičnih vezij je lahko napajalna napetost mikrokontrolerja in logičnega vezja v razponu 3-6 V (preverite dovoljene vrednosti za konkretno logično vezje!).

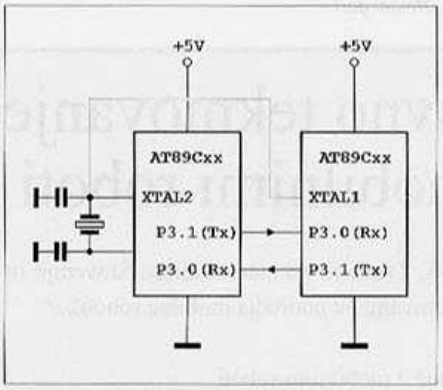


Slika 3a: Vse popularne družine logičnih vezij je možno direktno povezati na vhode in izhode mikrokontrolerja.

Če je mikrokontroler vhod, je nujno najprej postaviti vhodni pin v stanje "1", kot smo to že prej razložili.

Vezava dveh mikrokontrolerjev

Dva mikrokontrolerja lahko povežemo na različne načine, tukaj pa predlagam, da izkoristimo možnost serijske komunikacije – zato, ker nam je za povezavo dovolj že en sam vod (in skupna masa), vsi potrebni ukazi pa so že vsebovani v BascomLT-ju. Shema na sliki 3b je nekoliko kompleksnejša, vendar je to zato, ker je na njej zagotovljena možnost dvosmerne komunikacije: če en izmed mikrokontrolerjev samo pošilja in drugi samo sprejema podatke, lahko eno Tx-Rx povezavo izpustimo. Na sliki je prikazana tudi možnost, ko oba mikrokontrolerja uporabljata isti kvarčni kristal; v tem primeru na en mikrokontroler vežemo kristal na standardni način, na njegov XTAL2 priključek pa vežemo XTAL1 priključek drugega mikrokontrolerja (črtkano prikazana povezava na sliki).



Slika 3b: Povezava dveh mikrokontrolerjev preko serijskega vodila.

Na oddajni (Tx) strani pošiljamo podatke na naslednji način:

```
$crystal = 12000000
'frekvenca kvarčnega kristala
$baud = 2400
'hitrost: 2400 Baud
...
Printbin Znak
'pošlji byte
```

Na sprejemni strani lahko uporabimo dva ukaza, Inkey in Waitkey. Waitkey ustavi izvajanje programa, vse dokler ne dobi znaka po serijskem vodu (Rx vhod), lahko pa sprejme vseh 256 kombinacij (1 byte = 8 bitov = 256 kombinacij). Inkey bo nadaljeval z izvajanjem programa in bo v primeru, če ni sprejel nobenega znaka, vrnil vrednost 0; zaradi tega v primeru sprejema 0 ne vemo natančno, ali je bil sprejet ravno tak podatek ali pa ni prispel noben podatek. Ob upoštevanju te posebnosti lahko koristno uporabimo oba ukaza:

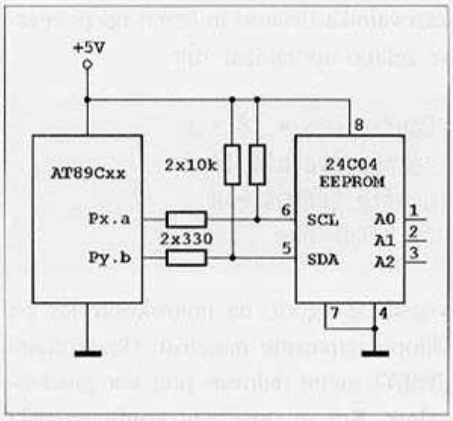
```
Znak = Inkey
'sprejmi byte
If Znak = 0
'ni prispel noben podatek
   ...
Else
'nadaljuj glede na sprejeti
podatek
   ...
End If
...
Znak = Waitkey
'čakaj dokler ne prispe byte
```

Pomemben pogoj za uspešno izvajanje takšne povezave je, da imata oba programa izbrano isto prenosno hitrost (ukaz $baud ni prikazan na sprejemni strani). Uspešno sem opravil s hitrostmi do 19200 Baudov; seveda je na obeh straneh potrebno vzpostaviti določeni protokol, da bi sprejemna stran "vedela", koliko znakov pošilja oddajna stran. Razen s pomočjo Inkey in Waitkey, lahko sprejem podatka opravimo tudi z aktiviranjem serijske prekinitvene rutine, kar bomo pustili za kakšno drugo priložnost.

Enako kot lahko serijsko komunikacijo izkoristimo ne le za izmenjavo podatkov med dvema mikrokontrolerjema, ampak tudi za npr. povezavo mikrokontrolerja in PC-ja (kot je to npr. narejeno v programatorju PG302), lahko za povezavo dveh mikrokontrolerjev uporabimo še druga dva protokola, ki jih podpira Bascom, I2C in 1-wire. V nadaljevanju bomo pokazali, kako lahko ta način povezave izkoristimo za komunikacijo z nekaterimi posebnimi elektronskimi komponentami.

I²C komunikacija

I²C protokol je bil razvit za potrebe komunikacije med integriranimi vezji v kompleksnejših napravah (npr. v televizorjih).



Slika 3c: Primer povezave preko I²C vodila.

Vodilo prenaša dva signala, SCL in SDA, nanj pa se lahko paralelno priključi večje število I²C komponent. Vsaki komponenti je določen edinstveni naslov, eno izmed integriranih vezij v omrežju pa je nadrejeno (*master*) in upravlja s pretokom podatkov.

V primeru na sliki 3c je nadrejen mikrokontroler, periferna komponenta pa je serijski EEPROM 24C04. Vsaka I²C komponenta ima svojo identifikacijsko številko, ki se za 24C04 glasi "1010abcd". Tukaj sta "a" in "b" stanji naslovnih pinov A1 in A2 (A0 se pri 24C04 ne uporablja), "c" naslavlja eno izmed dveh 256-bitnih *databank* znotraj EEPROMA, "d" pa določa ali gre za branje ali vpis. Princip I²C komunikacije je naslednji:

- *master* generira START signal (SCL = "1", SDA "1"->"0"),
- *master* naslovi neko izmed perifernih komponent, naslov za 24C04 bo 10100001 za branje in 10100000 za vpis, ob predpostavki, da je A1 = "0", A2 = "0",
- komunikacija je naprej odvisna od periferne komponente; če vpisujemo v 24C04, bo *master* poslal še 2 byta: naslov in podatek, ki ga je treba vpisati,
- *master* zaključi komunikacijo s STOP signalom (SCL = "1", SDA "0"->"1").

Bascom ima vgrajene vse potrebne rutine I²C protokola, zato je delo z njim zelo enostavno. Najprej se moramo odločiti, katere priključke na portih bomo uporabljali za signale SCL in SDA. To lahko naredimo na dva načina: z vpisom ustreznih nastavitev v meniju Options->Compiler->Misc ali, kar je boljše, z uporabo Config ukaza v samem programu:

```
Config Sda = P3.0
Config Scl = P3.1
Sedaj bomo na naslov 100
vpisali 255:
I2cstart
'start
I2cwbyte &B10100000
'naslovi 24C04, vpis
I2cwbyte 100
'naslov = 100
I2cwbyte 255
'podatek = 255
I2cstop
'stop
Waitms 10
'EEPROM potrebuje čas
'za vpis
```

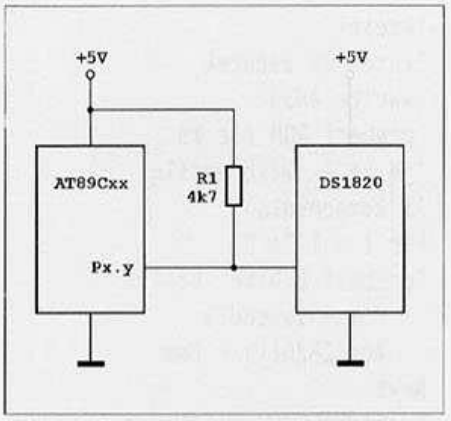
Pri branju najprej vpišemo želeni naslov (100), nakar prebrani podatek shranimo v spremenljivko Aa:

```
I2cstart
'start
I2cwbyte &B10100000
'naslovi 24C04, vpis
I2cwbyte 100
'naslov = 100
I2cstart
'start
I2cwbyte &B10100001
'naslovi 24C04, branje
I2crbyte Aa , 9
'preberi podatek v Aa
I2cstop
'stop
```

Ob upoštevanju posebnosti posameznih I²C komponent, se komunikacija z vsemi praviloma izvaja na enak način.

1-wire komunikacija

1-wire protokol je razvila firma *Dallas Semiconductor* za komunikacijo z nekaterimi svojimi komponentami. Omrežje je sestavljeno iz ene nadrejene (mikrokontroler) in ene ali več perifernih komponent, vse skupaj pa je povezano s samo enim vodom – vodilom, po katerem se prenašajo potrebni kontrolni in naslovni signali ter podatki. Zanimivo je, da je posamezne komponente (temperaturni senzor DS1820 v našem primeru) možno napajati neposredno preko vodila, kot tudi sugerira slika 3d. Podobno kot pri I2C komunikaciji, *master* upravlja s prometom v omrežju. Vsaka izmed perifernih komponent ima v ROM vpisan edinstven 64-bitni naslov naslednjega formata:



Slika 3d: Primer povezave preko 1-wire vodila.

- 8-bitna koda, ki identificira tip komponente (=0001000 za DS1820);
- 48-bitna koda, ki enoznačno identificira vsako komponento;
- 8-bitna kontrolna (CRC) koda.

Protokol pozna več ukazov za naslavljanje, odčitavanje ROM naslova, vpis in branje RAM-a periferne komponente. Komunikacija se začenja z RESET signalom ("0" v trajanju 480 µs), ki ga generira *master*, nakar sledi branje. Na RESET odgovorijo periferne komponente s signalom prisotnosti ("0" v trajanju 60-240 µs, po pavzi 15-60 µs). Nato *master* naslavlja posamezne komponente iz omrežja in z njimi vzpostavlja komunikacijo. Za uspešno komunikacijo mora *master* vnaprej poznati naslove vseh komponent v omrežju. V posebnem primeru, ko se v omrežju nahaja le ena periferna komponenta, lahko naslavljanje preskočimo, komunikacijo pa vzpostavimo neposredno. To je zelo koristno v primeru, ko želimo odčitati ROM naslov periferne komponente, da bi jo lahko kasneje naslavljali. Sledi primer programa za odčitavanje naslova; vse potrebne rutine so vgrajene v Bascom.

```
Dim Adr_1820(8) As Byte , I As
Byte , Pom As Byte
Config 1wire = P3.0
'P3.0 služi za
'1-wire komunikacijo
...
```

```
1wreset
'reset za začetek
1wwrite &H33
'preberi ROM (le ko
'je na 1 -wire vodilu
'1 komponenta)
For I = 1 To 8
'preberi 8-bitni naslov
   Pom = 1wread()
   Adr_1820(i) = Pom
Next
1wreset
'komunikacija je končana
```

Ko poznamo njen naslov, lahko periferno komponento pokličemo direktno:

```
1wreset
'reset za začetek
1wwrite &H55
'pozor, sledi naslov
For I = 1 To 8
   Pom = Adr_1820(i)
   1wwrite Pom
Next
```

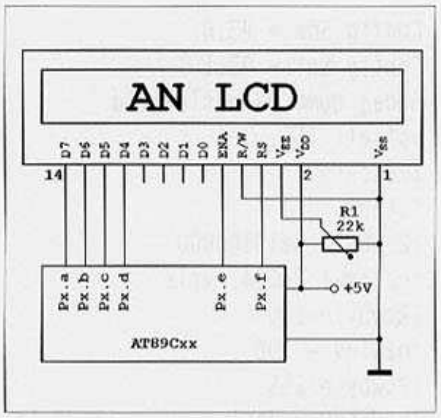
Na naslednji ukaz bo odgovorila samo komponenta, ki je naslovljena; v primeru DS1820 je ukaz lahko:

```
1wwrite &H44
'začni z merjenjem
Waitms 250
'čas potreben za
Waitms 250
'merjenje temperature
```

Na podoben način bi sedaj lahko odčitali temperaturo ali vzpostavili komunikacijo z drugimi 1-wire komponentami.

Alfanumerični LCD

Alfanumerične prikazovalnike zelo pogosto uporabljamo v mikrokontrolerskih projektih. Ne glede na izvedbo (1, 2 ali 4 vrstice, 16, 20 ali 40 znakov v vrstici), sta razpored priključkov in način povezovanja enaka: poleg 8 *data* bitov (D7-D0) so tu še trije kontrolni signali: ENA, R/W in RS. R/W običajno fiksno vežemo na maso (na prikazovalnik samo pišemo), zaradi varčevanja s številom linij pa prikazovalnik najpogosteje uporabljamo v 4-bitnem režimu (uporabljamo samo D7-D4). Ravno takšna vezava je prikazana na sliki 4a. S trimerjem R1 nastavljamo kontrast; tukaj je potrebno biti pazljiv, ker je področje regulacije precej ozko (nepravilno nastavljen kontrast bo rezultiral s popolnoma belim ali popolnoma črnim prikazom). Prikazovalniki z osvetlitvijo ozadja imajo še dva priključka, za anodo in katodo osvetlitvene ploščice. Odvisno od izvedbe prikazovalnika sta ta dva priključka dodana ali kot priključka 15 in 16 ali pa "desno" od priključka 1, kar lahko uporabnika precej zmede. Tokokrog osvetlitve ozadja je običajno popolnoma ločen od tokokroga prikazovalnika, vendar sta na tiskanem vezju predvideni spajkalni mesti, s katerih lahko uporabimo skupno maso, skupno napajanje ali oboje. Takšne stvari najpogosteje niso nikjer dokumentirane, temveč jih je potrebno ugotoviti z merilnim instrumentom in zdravim razumom. Osvetlitev ozadja je znatno večji porabnik od samega prikazovalnika (50 - 100 mA ali tudi več od tega), kar lahko povzroči morebitno neželeno pregretje napetostnega stabilizatorja, iz katerega se običajno napaja celotno vezje. Vendar pa za napajanje osvetlitve niti ne potrebujemo stabilizirane napetosti. Dobra rešitev je, če v serijo z osvetlitvijo namestimo upor reda 47 - 200 ohmov /1 W; ustrezno vrednost določimo eksperimentalno in je odvisna od delovne napetosti in želene intenzitete osvetljenosti.



Slika 4a: Povezava mikrokontrolerja in alfanumeričnega LCD prikazovalnika.

Vse verzije Bascoma imajo veliko izbiro ukazov za delo z alfanumeričnimi prikazovalniki (pozicioniranje, izpis, premikanje teksta). Vendar je pred njihovo uporabo potrebno definirati, s katerim tipom prikazovalnika delamo in kateri tip povezave želimo uporabljati, npr.:

```
Config Lcd = 16 * 2
'prikazovalnik 16x2
Config Lcdbus = 4
'4-bitni bus
```

Včasih se zgodi, da mikrokontroler po vklopu napajalne napetosti (kratkotrajni RESET) začne delovati prej kot prikazovalnik. Ker se omenjeni konfiguracijski ukazi običajno nahajajo na samem začetku programa, jih prikazovalnik ne utegne sprejeti in se zato kasneje "čudno" obnaša. Zato je dobro na sam začetek Bascom programa postaviti Wait 1. Še ena pripomba, ki velja za Config Lcd ukaz: na trgu sta dve izvedbi 16x1 prikazovalnika, zato sta v Bascomu predvideni tudi dve konfiguracijski opciji:

```
Config Lcd = 16 * 1
```

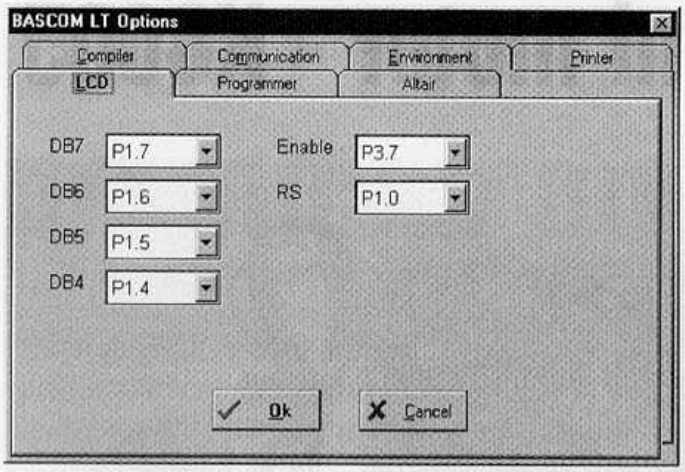
in

```
Config Lcd = 16 * 1a.
```

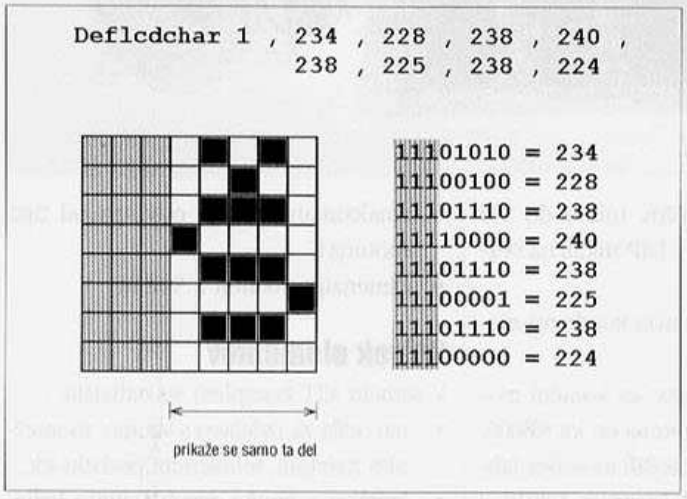
Ob izbiri napačnega konfiguracijskega ukaza desni del prikazovalnika ne bo deloval. Po mojih izkušnjah so pogostejši prikazovalniki, katerim bolj ustreza opcija "16*1a".

Poleg konfiguracije je v Bascomu še potrebno opisati, katere priključke uporabljamo za komunikacijo s prikazovalnikom. To naredimo v meniju Options->LCD (slika 4b). Ti podatki so potrebni Bascom prevajalniku. Naj ponovim, če uporabljate P1.0 in P1.1 ali kateri drugi pin brez internega *pull-up* upora, je obvezno potrebno predvideti zunanje *pull-up* upore, sicer prikazovalnik ne bo deloval pravilno (vhodi nekaterih tipov prikazovalnika imajo lastne *pull-up* upore, vendar tega ne moremo vzeti kot pravilo).

Po pravilnem konfiguriranju lahko uporabljamo vse LCD ukaze; delo z njimi je enostavno, ker so dobro razloženi v *Help*-u in spremljajočih primerih, zato se ne bom predolgo zadrževal ob njihovi razlagi. Omenil bom le še možnost generiranja lastnih znakov, ker gre za nekoliko kompleksnejši postopek, ki v Bascom *Help*-u ni



Slika 4b: Konfiguriranje alfanumeričnega prikazovalnika v meniju Options->LCD.



Slika 4c: Generiranje znaka "š" s pomočjo ukaza Deflcdchar.

popolnoma korektno opisan. V naslednjem primeru bomo generirali dva znaka, "š" in "č":

```
Deflcdchar 1 , 234 , 228 , 238 , 240 , 238 , 225
, 238 , 224
Home U
Deflcdchar 2 , 234 , 228 , 238 , 240 , 240 , 241
, 238 , 224
Home U
```

Izkušnje so pokazale, da je ukaz Home ali Cls nujno uporabiti po vsakem generiranem karakterju, čeprav to v Bascomu ni dokumentirano. Oglejmo si malo natančneje, kako je nastal znak "š" (slika 4c). V Bascomu je za to predvideno posebno orodje, LCD Designer, ki ga lahko najdete v meniju Edit. Postopek je sestavljen iz risanja želenega znaka, nakar Bascom zgenerira ustrezni Deflcdchar ukaz. Pozor, v starejših verzijah programa LCD Designer ni pravilno deloval, zato je dobljeni rezultat potrebno preveriti še "ročno"!

V LCD prikazovalnik lahko vpišemo do 8 lastnih karakterjev (0-7). Ker se opis znaka shranjuje v RAM prikazovalnika, je potrebno definicije lastnih karakterjev namestiti na začetek programov, v katerih jih želimo uporabljati, še najbolje takoj za Config ukazom. Po potrebi je posamezne znake možno med izvajanjem programa tudi redefinirati. Poglejmo si, kako uporabljamo uporabniško definirane karakterje:

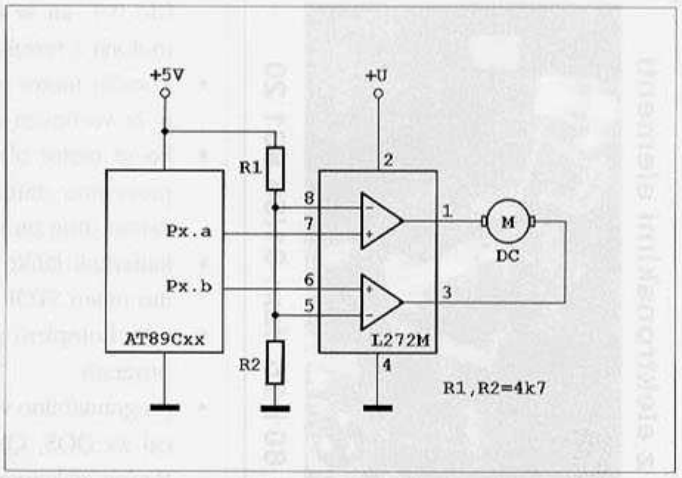
```
Lcd "mi" ; Chr(1) ; "ka!" 'miška!
```

DC motor

Mikrokontroler ni zmožen direktno poganjati motorja, saj tipičen majhen DC motor za svoje delovanje potrebuje tokove do 100 in več mA. Ena od možnostih za to je, da si zgradimo prilagoditveno vezje s tranzistorji na osnovi slik 2a in 2b, vendar pa je boljša rešitev, če za krmiljenje motorja uporabimo specialna, posebej za ta namen projektirana integrirana vezja. Eno takšnih vezij, IC L272M, je uporabljeno v primeru na sliki 5. L272M je v principu sestavljen iz 2 operacijskih ojačevalnikov in močne izhodne stopnje. To vezje poleg vzbujanja motorja opravlja tudi potrebno konverzijo napetostnega nivoja (+U napetost, s katero napajamo DC motor, je običajno višja od napajalne napetosti mikrokontrolerja). Za ta namen sta uporabljena upora R1 in R2, ki postavljata referenčno napetost na "-" vhodih približno na polovico napetosti, s katero se napaja mikrokontroler. Upravljanje z delovanjem motorja je sestavljeno iz naslednjega:

- če sta oba kontrolna priključka, Px.a in Px.b, na nizkem nivoju (ali na visokem nivoju), je motor izklopljen;
- če je Px.a = "1" in Px.b = "0", se motor vrti v eni smeri;
- če je Px.a = "0" in Px.b = "1", se motor vrti v drugi smeri.

Torej nam izdelava ustreznega Bascom programa za upravljanje z delovanjem motorja na osnovi navedenih specifikacij očitno ne bo delala težave.



Slika 5: Mikrokontroler krmili DC motor.

Vse opisane vezave in programski primeri so preizkušeni v praksi in zanesljivo delujejo. Predstavljeni so primeri le nekaterih od načinov povezav mikrokontrolerjev z različnimi komponentami, možne pa so seveda tudi mnoge druge variante, ki so odvisne od vaše domišljije in potreb. Poskusite še sami zgraditi kakšno originalno vezavo in se nam oglasite – v reviji jo bomo z veseljem objavili.

Avtor: mag. Vladimir Mitrović