Izdelava mobilnega robota z razvojnim sistemom roboPIC (6)

Avtor: Silvan Bucik
E-pošta: silvan.bucik@tscng.net
www.svet-el.si/phpBB2/index.php

Pretekli mesec smo se igrali z enosmernimi motorčki. V tokratnem pisanju si bomo ogledali uporabo servomotorjev. Ti dve vrsti motorjev se med seboj razlikujeta tako v namenu kot načinu uporabe. Enosmerne motorje v večini primerov uporabljamo kot pogonska sredstva, medtem ko nam servomotorji v glavnem služijo kot pozicionerji; na primer pomik zakrilc modelov letal, pomični sistem robotske roke, upravljanje volana daljinsko vodenih avtomobilčkov ...

Rešitve domačih nalog

Preden pričnemo z »obdelavo« servomotorjev, predlagam, da si za začetek pogledamo rešitve domačih nalog iz prejšnje številke.
Naloga 1: Spremenite program tako, da se bo namesto motorja M2 vrtel motor M1, ki je vezan na priključek X4.
Naloga ni posebno zahtevna; pogledati je potrebno v električno shemo ter ustrezne priključke definirati kot izhode in jih seveda ustrezno krmiliti.

```
void main (void)
{
TRISA = 0B11111111;    // PORTA = vhod, ga ne uporabljamo
ADCON1 = 0x07;         // PORTA definiramo kot digitalna
                          vhodno-izhodna vrata
TRISB = 0B11111111;    // PORTB = vhod, ga ne uporabljamo
TRISC = 0B11110000;    // *** sedaj so RC0, RC3, RC2, RC1
                          izhodni priključki
TRISD = 0B11111111;    // PORTD = vhod, ga ne uporabljamo
TRISE = 0B00000111;    // PORTE = vhod, ga ne uporabljamo
RC2 = 1;               // postavitev bita ENABLE1;
                       // omogočeno delovanje sklopa MOT1 v
                          int. vezju L293
RC1 = 0;               // brisanje bita ENABLE2;
                       // onemogočeno delovanje sklopa MOT2 v
                          int. vezju L293

// GLAVNI PROGRAM
RC0 = 1;               // ***vrtenje motorja
RC3 = 0;               // ***v izbrano smer
wait_sec(5);           // počakaj 5 sekund
RC0 = 0;               // ustavi
RC3 = 0;               // motor

stop:
goto stop;
}
```

Naloga 2: Spremenite program, da bo motor tekel le v primeru, ko bo pritisnjena tipka RA2.
V programu bomo uporabili pogojni stavek if-else, s katerim bomo določili, kdaj naj se motor vrti in kdaj ne.

```
void main (void)
{
TRISA = 0B11111111;    // PORTA = vhod, ga ne uporabljamo
ADCON1 = 0x07;         // PORTA definiramo kot digitalna
                          vhodno-izhodna vrata
TRISB = 0B11111111;    // PORTB = vhod, ga ne uporabljamo
TRISC = 0B11001001;    // RC4, RC5, RC2, RC1 so izhodi
TRISD = 0B11111111;    // PORTD = vhod, ga ne uporabljamo
TRISE = 0B00000111;    // PORTE = vhod, ga ne uporabljamo
RC2 = 0;               // brisanje bita ENABLE1;
                          // onemogočeno delovanje sklopa
MOT1 v                       int. vezju L293
RC1 = 1;               // postavitev bita ENABLE2;
                       // omogočeno delovanje sklopa MOT2 v
                       int. vezju L293
// GLAVNI PROGRAM
while (1>0)
{
if (RA2==0)
{
RC4 = 0;               // vrtenje motorja
RC5 = 1;               // v izbrano smer
}
else
{
RC4 = 0;               // ustavi
RC5 = 0;               // motor
}
}
}
```

Naloga 3: Napišite program, ki bo povzročil zamenjavo smeri vrtenja motorja vsake 3 sekunde.
Napisali bomo neskončno zanko while (1>0), znotraj zanke pa bomo komutirali polariteto priključkov motorja.

```
void main (void)
{
TRISA = 0B11111111;    // PORTA = vhod, ga ne uporabljamo
ADCON1 = 0x07;         // PORTA definiramo kot digitalna
                          vhodno-izhodna vrata
TRISB = 0B11111111;    // PORTB = vhod, ga ne uporabljamo
TRISC = 0B11001001;    // RC4, RC5, RC2, RC1 so izhodi
TRISD = 0B11111111;    // PORTD = vhod, ga ne uporabljamo
TRISE = 0B00000111;    // PORTE = vhod, ga ne uporabljamo
RC2 = 0;               // brisanje bita ENABLE1;
```

```
                        // onemogočeno delovanje sklopa MOT1 v
                           int. vezju L293
RC1 = 1;                // postavitev bita ENABLE2;
                        // omogočeno delovanje sklopa MOT2 v
                           int. vezju L293

// GLAVNI PROGRAM
while (1>0)
{
RC4 = 1;                // vrtenje motorja
RC5 = 0;                // v levo
wait_sec(3);            // počakaj 3 sekunde
RC4 = 0;                // vrtenje motorja
RC5 = 1;                // v desno
wait_sec(3);            // počakaj 3 sekunde
}
}
```

Naloga 4: Napišite program, ki bo povzročil vrtenje motorja le s četrtino moči.
Vsa umetnost naloge je, da nastavimo prevajalno razmerje PWM signala na vrednost 25%.

```
void main (void)
{
TRISA = 0B11111111;     // PORTA = vhod, ga ne uporabljamo
ADCON1 = 0x07;          // PORTA definiramo kot digitalna
                           vhodno-izhodna vrata
TRISB = 0B11111111;     // PORTB = vhod, ga ne uporabljamo
TRISC = 0B11110010;     // RC0, RC3, RC2 so izhodi
TRISD = 0B11111111;     // PORTD = vhod, ga ne uporabljamo
TRISE = 0B00000111;     // PORTE = vhod, ga ne uporabljamo
RC2 = 1;                // postavitev bita ENABLE1;
                        // omogočeno delovanje sklopa MOT1 v
                           int. vezju L293

//GLAVNI PROGRAM
pwm1_out(5000,25);      // izbrana frekvenca znaša
                        // 200Hz (perioda=5000us);
                        // širina impulza pa je 25%
RC0 = 1;
RC3 = 0;
stop:
goto stop;
}
```

Naloga 5: Napišite program, ki bo motor pospeševal iz mirovanja do maksimalne hitrosti v treh stopnjah (mehki zagon).
V časovnih razmakih po 2 sekundi bomo povečevali prevajalno razmerje PWM modula, in sicer od vrednosti 0 do 100% v korakih po 33%.

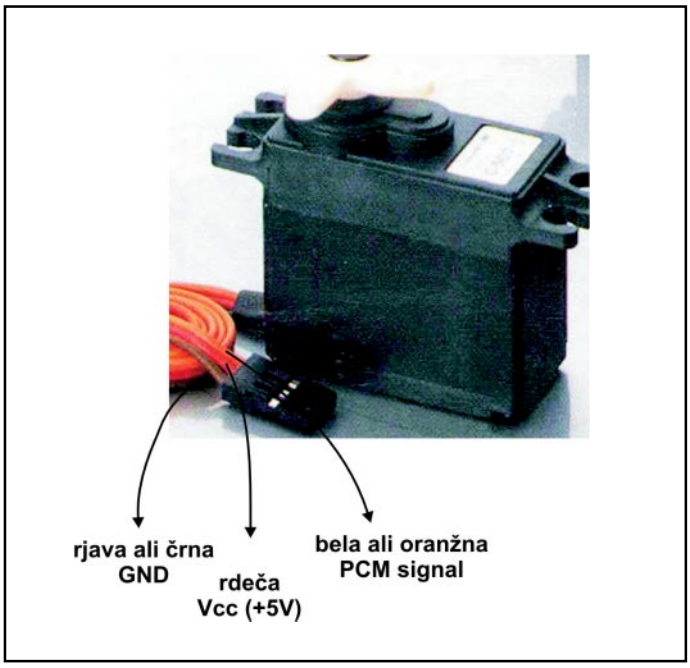
```
void main (void)
{
TRISA = 0B11111111;     // PORTA = vhod, ga ne uporabljamo
ADCON1 = 0x07;          // PORTA definiramo kot digitalna
                           vhodno-izhodna vrata
TRISB = 0B11111111;     // PORTB = vhod, ga ne uporabljamo
TRISC = 0B11110010;     // RC0, RC3, RC2 so izhodi
TRISD = 0B11111111;     // PORTD = vhod, ga ne uporabljamo
TRISE = 0B00000111;     // PORTE = vhod, ga ne uporabljamo
RC2 = 1;                // postavitev bita ENABLE1;
// omogočeno delovanje sklopa MOT1 v
                           int. vezju L293

//GLAVNI PROGRAM
RC0 = 1;                // vrtenje motorja
RC3 = 0;                // v izbrano smer
start:
pwm1_out(5000,0);       // širina impulza pa je 0% - motor
                           stoji
wait_sec(2);
pwm1_out(5000,33);      // širina impulza pa je 33%
wait_sec(2);
pwm1_out(5000,67);      // širina impulza pa je 67%
wait_sec(2);
pwm1_out(5000,100);     // širina impulza pa je 100%
wait_sec(2);
goto start;
}
```

Servomotorji

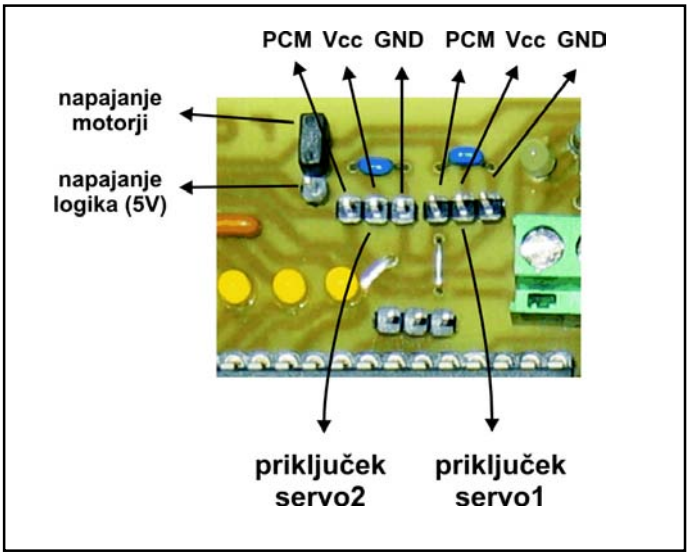
Sedaj pa k servomotorjem. Servomotorje bomo pogosto srečali v radijsko vodenih modelih. Uporabljajo se za krmiljenje krilc letal, za premikanje rok dvigal in še marsikje. Ti motorčki niso prosto vrteči, pomikajo se lahko le levo-desno v območju kota približno 270°. Za napajanje in krmiljenje motorčkov se uporabljajo trije priključki:

- dva priključka sta napajalna (Vcc in GND),
- tretji pa je namenjen krmiljenju (PCM).

Način priključitve servomotorčkov na razvojno ploščo ponazarja spodnja slika. Ob tej priložnosti bi želeli opozoriti na postavitev mostička (jumperja), s katerim določimo izvor napajanja servomotorjev. Kot smo že v predstavitvi razvojnega sistema omenili (Svet elektronike, maj 2005 in julij/avgust 2005), imamo na tiskanem vezju ločeni napajanji senzorskega in pogonskega dela. Vse pogonske motorje naj bi napajali preko napajalne linije »napajanje motorji«, v električni shemi označene s potencialom



Slika 1: Priključki servomotorja

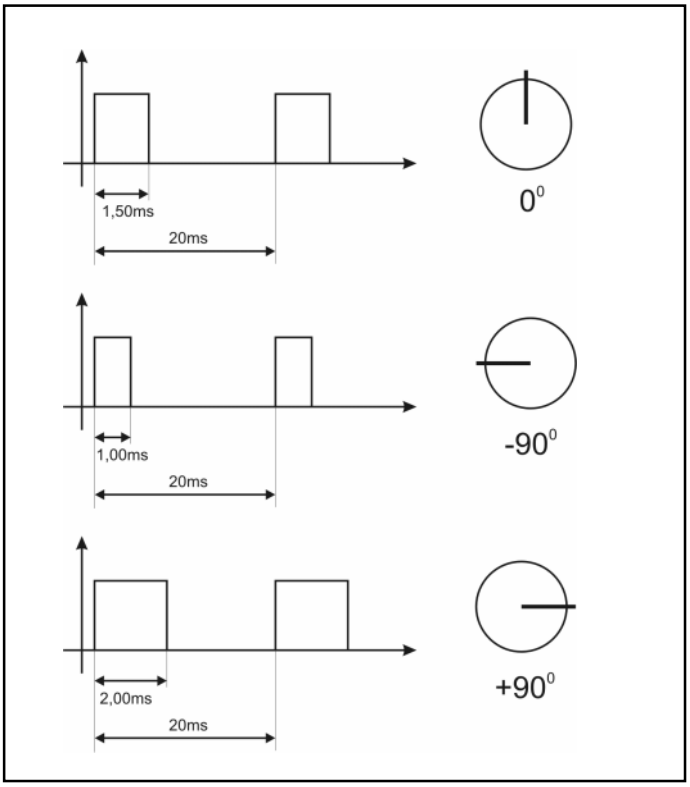


Slika 2: Razporeditev priključnih sponk na razvojni plošči

+12 V. Vendar pa se lahko primeri; tu imamo v mislih konfiguracijo sistema en DC motor in en servomotor, da postane vrednost napajalne napetosti za servomotorje previsoka. V teh situaciji moramo prestaviti mostiček na pozicijo »napajanje logika (+5 V)«. S tem sicer nimamo več ločenega napajanja senzorjev in servomotorjev, zato pa lahko napajalno napetost DC motorja dvignemo na poljubno želeno vrednost; robot pa bo hitreje dirkal.

Krmiljenje servomotorja

Servomotorji se krmilijo s PCM (pulse coded modulation) moduliranim signalom. Ta signal je v principu PWM signal frekvence 50Hz, katerega prevajalno razmerje se giblje v območju od 5% do 10%. Prevajalno razmerje 5% (dolžina impulza 1 ms) postavi os motorja v skrajno levo pozicijo, vrednost 10% (dolžina impulza 2 ms) pa v skrajno desno pozicijo. Ostale vmesne vrednosti pa postavijo os motorja nekje vmes. Omenjene vrednosti so zgolj orientacijske in se od motorja do motorja precej razlikujejo.



Slika 3: Princip krmiljenja servomotorjev s PCM modulacijo

Napišimo program, ki bo omogočil pomik osi servomotorja, priključenega na konektor SL4 (izhod PWM2), v skrajno levo lego. Čez pet sekund naj se os motorja postavi v nevtralno lego in po naslednjih petih sekundah naj se pomakne v skrajno desno pozicijo. Omenjeni postopek naj se ponovi trikrat. Priključek za priklop servomotorja 2 je na sponki SL4, izhod za servomotor 1 pa na sponki SL3.

Najprej izračunamo pripadajoče vrednosti registrov CCPR (v mislih imamo register CCPRxL ter bita 5 in 4 registra CCPxCON) za osne pozicije zasuka motorja 90° v levo, 90° v desno ter za mirovno lego. Podrobnosti uporabe PWM modula so opisane v novembrskem prispevku. Uporabili bomo enačbo:

$$vrednost\_CCPR = prevajalno\_razmerje(v\_\%) \cdot 1024$$

Kot rezultat dobimo sledeče vrednosti registrov CCPR:

- vrednost 51 pri zasuku v levo,
- vrednost 102 pri zasuku v desno ter
- vrednost 77,v mirovni legi.

Poglejmo si postopek izgradnje programa:

- priključek mikrokontrolerja RC1 definiramo kot izhod, vsi ostali priključki so vhodi,
- nastavimo prevajalno razmerje modula PWM2 za zasuk v levo (uporabimo knjižnično funkcijo servo2_out(vrednost_CCPR)),
- počakamo 5 s (uporabimo knjižnično funkcijo wait_sec (zakasnitev_v_s)),
- nastavimo prevajalno razmerje modula PWM2 za nevtralno lego,
- počakamo 5 s,
- nastavimo prevajalno razmerje modula PWM2 za zasuk v desno,
- počakamo 5 s,
- vse skupaj naj se izvede trikrat (uporabimo programsko zanko for).

Rešitev problema v programskem jeziku C:

```
// INICIALIZACIJA
void main (void)        // IME GLAVNE FUNKCIJE
{
TRISA = 0B11111111;     // PORTA definiramo kot vhod, ga ne
                           uporabljamo
ADCON1 = 0x07;          // PORTA definiramo kot digitalna
                           vhodno-izhodna vrata
TRISB = 0B11111111;     // PORTB definiramo kot vhod, ga ne
                           uporabljamo
TRISC = 0B11111101;     // RC1 definiramo kot izhod
TRISD = 0B11111111;     // PORTD definiramo kot vhod, ga ne
                           uporabljamo
TRISE = 0B00000111;     // PORTE definiramo kot vhod, ga ne
                           uporabljamo

//GLAVNI PROGRAM
for (st_ponovitev=0;st_ponovitev<3;
st_ponovitev=st_ponovitev+1)
{                       // zanka se bo izvršila 3-krat
servo2_out(51);         // skrajna leva lega
```

```
                        // PCM impulz = 5% (1ms)
wait_sec(5);            // počakaj 5 sekund
servo2_out(77);         // nevtralna lega
                        // PCM impulz = 7,5% (1,5ms)
wait_sec(5);            // počakaj 5 sekund
servo2_out(102);        // skrajna desna lega
                        // PCM impulz = 10% (2ms)
wait_sec(5);            // počakaj 5 sekund
}
stop:
goto stop;
}}
```

Kot smo že v začetku omenili, servomotorje krmilimo s pulzno-širinskim modulatorjem. Perioda PWM oscilatorja je konstantna in znaša 16,4 ms. Prevajalno razmerje je poljubno nastavljivo od vrednosti 0 do 1023. Za tiste, ki servomotorjev doslej še niste uporabljali, naj velja majhna previdnost. Aktivni impulz PWM signala se mora nahajati v območju od 5% do 10% celotne periode PWM signala, kar ustreza nastavitvam vrednosti registrov CCPR v območju od 51 do 103. Vrednosti izven navedenega intervala lahko povzročijo zasuk osi motorja v skrajno lego (motor bo »zabilo«), s čimer lahko motor poškodujemo. Omenjeni mejni vrednosti prevajalnega razmerja sta zgolj informativnega značaja, zato ju upoštevajte le ob prvem preizkusu. Nato servomotor skalibrirajte, tako da eksperimentalno poiščete njegovi pravi mejni vrednosti.

Servomotorji za »telebane«: funkciji servo1_out() in servo2_out()

V opisanem programu ste opazili uporabo funkcije servo2_out(). S pomočjo funkcij servo1_out() in servo2_out() lahko na enostaven način upravljamo s servomotorji, ne da bi nam bilo potrebno poznati nastavitev posameznih registrov. Funkciji se nahajata v knjižnični datoteki servo.c.

Uporaba:

```
servo1_out(vrednost_CCPR);      // primer uporabe modula PWM1
servo2_out(vrednost_CCPR)       // primer uporabe modula PWM2
```

Samostojno rešite sledeči nalogi:

1. Napišite program, ki bo postavil os servomotorja na poziciji, ki sta pod kotom 45° glede na nevtralno lego. Os naj v vsaki legi ostane za 2 sekundi.
2. Napišite program, ki bo kot rezultat postavil os motorja v skrajno levo lego, ob pogoju, da je pritisnjena tipka RA2. Os naj se pomakne v skrajno desno lego, če bo pritisnjena tipka RA5. Sicer naj os ostane v nevtralni legi.

Poglejmo še v »črevesje« PWM generatorja

Za tiste, ki vas zanimajo podrobnosti upravljanja servomotorja s PCM signalom, naj namenimo še nekaj vrstic. Osnovni podatki, ki jih moramo poznati so:

- perioda PCM signala znaša okoli 20 ms,
- PCM impulz za skrajno levo lego znaša približno 1 ms (5%) in
- za skrajno desno lego 2 ms (10%).

Periodo PCM signala določa register PR2 in jo izračunamo po sledeči enačbi:

$$perioda = 4 * (PR2 + 1) * delilno\_razmerje\_timer2/frekvenca\_oscilatorja$$

Po premetavanju enačbe kot rezultat dobimo ustrezno vrednost registra PR2; za periodo 20 ms ta vrednost znaša 312,5. Ker je PR2 8-bitni register, lahko vanj vpišemo kot največjo možno vrednost le 255, kar posledično pomeni, da je najdaljša možna perioda PCM signala le 16,384 ms. To pa ni nič katastrofalnega, saj je praksa pokazala, da servomotorji tudi tej vrednosti periode še vedno kvalitetno delujejo.

Dolžino aktivnega impulza (prevajalnega razmerja) določajo register CCPR1L (CCPR2L) ter bita 4 in 5 v registru CCP1CON (CCP2CON):

$$PCM\_impulz = (4 * CCPR1L + (bita\_5\_4\_CCP1CON))$$

$$* delilno\_razmerje\_timer2/frekvenca\_oscilatorja$$

Če v enačbi vstavimo vrednosti 5% (0,8192 ms) ter 10% (1,6384 ms), dobimo sledeče vrednosti registrov:

- za prevajalno razmerje 5% je vrednost CCPR1L enaka decimalni vrednosti 12; bita 5 in 4 CCP1CON sta enaka binarni vrednosti 11;
- za prevajalno razmerje 10% je vrednost CCPR1L enaka decimalni vrednosti 25; bita 5 in 4 CCP1CON sta enaka binarni vrednosti 10.

Primer uporabe v programskem jeziku C:

Poglejmo si še praktični primer vrtenja osi servomotorja iz ene skrajne lege v drugo.

```
char st_ponovitev;      // INICIALIZACIJA
void main (void)        // IME GLAVNE FUNKCIJE
{
TRISA = 0B11111111;     // PORTA definiramo kot vhod, ga ne
                           uporabljamo
ADCON1 = 0x07;          // PORTA definiramo kot digitalna
                           vhodno-izhodna vrata
TRISB = 0B11111111;     // PORTB definiramo kot vhod, ga ne
                           uporabljamo
TRISC = 0B11111011;     // RC2 = izhod
TRISD = 0B11111111;     // PORTD definiramo kot vhod, ga ne
                           uporabljamo
TRISE = 0B00000111;     // PORTE definiramo kot vhod, ga ne
                           uporabljamo
T2CON = 0B00000100;     // vklop časovnika Timer2
CCP1CON = 0B00001100;   // izbran je PWM način delovanja
                           sklopa CCP1 modula
//GLAVNI PROGRAM
T2CON = 0B00000111;
PR2 = 255;              // perioda znaša 16,384ms
for (st_ponovitev=0;st_ponovitev<3
;st_ponovitev=st_ponovitev+1)
{
CCPR1L = 0B00001100;    // skrajna leva lega (51 = 12*4 + 3)
CCP1CON = (CCP1CON&0B11001111)|(3<<4);
                        // PCM impulz = 5%
wait_sec(2);
CCPR1L = 0B00011001;    // skrajna desna lega (102 = 25*4 + 2)
CCP1CON = (CCP1CON&0B11001111)|(2<<4);
                        // PCM impulz = 10%
wait_sec(2);
}
stop:
goto stop;
}
```

CYPRESS SEMICONDUCTOR

IR Electronic d.o.o.
Ziherlova ulica 2,
1000 Ljubljana, Slovenija
Tel.: 1.2835604,
Fax: 1.2835605
E-mail: info@irelectronic.si

CYPRESS
Connecting From Last Mile to First Mile.®

PSoC (Programable System On-Chip)

PSoC krmilniki so 8-bitni mikrokontrolerji, ki nam v enem ohišju ponujajo mikroprocesorsko »jedro« ter različno število analognih in digitalnih blokov potrebnih v naprednih aplikacijah.
Konfiguracija krmilnika ni vnaprej določena in je prepuščena uporabniku. Spreminja se lahko (do tri različne konfiguracije) tudi med delovanjem.V PSoC krmilnikih najdemo do 32kB Flash programskega spomina, 2kB SRAM spomina, ohišja pa so 8 do 100 pinska.
PSoC krmilniki bodo v prihodnosti dobavljivi samo v »PB Free« verzijah.
Razdeljeni v skupine, ki se med seboj razlikujejo v številu anal. in digitalnih blokov, številu V/I in velikosti spomina:

1.CY8C21xxx

OZNAKA	# V/I	ANALOGNI BLOKI	DIGITALNI BLOKI	SPOMIN	OHIŠJE	RAM	NAPAJANJE
CY8C21123-24SXI	6	4 Type "E"	2-Basic 2-Comms	4KB Flash	8	256bytes	2.4V to 5.25V
CY8C21323-24LFXI	16	4 Type "E"	2-Basic 2-Comms	4KB Flash	24	256bytes	2.4V to 5.25V
CY8C21323-24PVXI	16	4 Type "E"	2-Basic 2-Comms	4KB Flash	20	256bytes	2.4V to 5.25V
CY8C21223-24SXI	12	4 Type "E"	2-Basic 2-Comms	4KB Flash	16	256bytes	2.4V to 5.25V

2.CY8C22xxx

OZNAKA	# V/I	ANALOGNI BLOKI	DIGITALNI BLOKI	SPOMIN	OHIŠJE	RAM	NAPAJANJE
CY8C22113-24PI	6	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	8	256bytes	3.0V to 5.25V
CY8C22113-24SI	6	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	8	256bytes	3.0V to 5.25V
CY8C22213-24LFI	28	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	32	256bytes	3.0V to 5.25V
CY8C22213-24LFIT	28	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	32	256bytes	3.0V to 5.25V
CY8C22213-24PI	16	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	20	256bytes	3.0V to 5.25V
CY8C22213-24PVI	16	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	20	256bytes	3.0V to 5.25V
CY8C22213-24SI	16	3:1-CT 2-SC	2-Basic 2-Comms	2KB Flash	20	256bytes	3.0V to 5.25V

3.CY8C24xxx

OZNAKA	# V/I	ANALOGNI BLOKI	DIGITALNI BLOKI	SPOMIN	OHIŠJE	RAM	NAPAJANJE
CY8C24123-24PI	6	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	8	256bytes	3.0V to 5.25V
CY8C24123-24SI	6	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	8	256bytes	3.0V to 5.25V
CY8C24423-24PI	24	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	28	256bytes	3.0V to 5.25V
CY8C24423-24PVI	24	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	28	256bytes	3.0V to 5.25V
CY8C24423-24SI	24	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	28	256bytes	3.0V to 5.25V
CY8C24223-24PI	16	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	20	256bytes	3.0V to 5.25V
CY8C24223-24PVI	16	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	20	256bytes	3.0V to 5.25V
CY8C24223-24SI	24	6:2-CT 4-SC	2-Basic 2-Comms	4KB Flash	28	256bytes	3.0V to 5.25V

4.CY8C27xxx

OZNAKA	# V/I	ANALOGNI BLOKI	DIGITALNI BLOKI	SPOMIN	OHIŠJE	RAM	NAPAJANJE
CY8C27143-24PI	6	12:4-CT 8-SC	4-Basic 4-Comms	16KB Flash	8	256bytes	3.0V to 5.25V
CY8C27643-24LFI	40	12:4-CT 8-SC	4-Basic 4-Comms	16KB Flash	48	256bytes	3.0V to 5.25V
CY8C27643-24PVI	44	12:4-CT 8-SC	4-Basic 4-Comms	16KB Flash	48	256bytes	3.0V to 5.25V
CY8C27243-24PVI	16	12:4-CT 8-SC	4-Basic 4-Comms	16KB Flash	20	256bytes	3.0V to 5.25V
CY8C27443-24PI	24	12:4-CT 8-SC	4-Basic 4-Comms	16KB Flash	28	256bytes	3.0V to 5.25V
CY8C27543-24AI	40	12:4-CT 8-SC	4-Basic 4-Comms	16KB Flash	44	256bytes	3.0V to 5.25V

5.CY8C29xxx

OZNAKA	# V/I	ANALOGNI BLOKI	DIGITALNI BLOKI	SPOMIN	OHIŠJE	RAM	NAPAJANJE
CY8C29466-12PVXE	24	12:4-CT 8-SC	8-Basic 8-Comms	32KB Flash	28	2K	4.75V to 5.25V
CY8C29666-12PVXE	44	12:4-CT 8-SC	8-Basic 8-Comms	32KB Flash	48	2K	4.75V to 5.25V

IR electronic d.o.o.
je z vzorci podprl prvo proizvodnjo umLab PSoC programatorja!

PRoC : PSoC (8k Flash, 512 bytov SRAM, 4 x anal.bloki, 4 x dig.bloki)
+ RF transceiver 2.4GHz

Več informacij:
IR electronic d.o.o.
g. Armin Čatak
E-mail: ACatak@arrowslovenia.com

Do prihodnje številke

Tako, s pogoni smo zaključili. Prihodnjič bomo pričeli spoznavati senzoriko. En lep pozdrav ... do prihodnje številke. ●

Literatura:

[1] Microchip, PIC16F87x data sheet, Microchip Technology Incorporated, 2001, http://ww1.microchip.com/downloads/en/DeviceDoc/30292c.pdf;
[2] B. Peršič, Gradnja mikroprocesorskih sistemov, Ljubljana: Fakulteta za elektrotehniko, 1998,
[3] HI-TECH, PICC Lite C Manual, HI-TECH Software, 2002, http://www.htsoft.com/downloads/manuals.php;
[4] B.W. Kernighan, D.M. Ritchie, Programski jezik »C«, Ljubljana: Fakulteta za elektrotehniko, 1991,
[5] STMicroelectrinics, L293 datasheet, push-pull four channel drivers, http://www.st.com/stonline/books/pdf/docs/1328.pdf;

Programska oprema:

[1] Microchip, MPLAB IDE v6.61, http://ww1.microchip.com/downloads/en/DeviceDoc/mp661.zip;
[2] HI-TECH software, PICC lite COMPILER v8.05PL2, http://www.htsoft.com/products/PICClite.php;
[3] Microchip, PIC18F/PIC16F Quick Programmer, http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1824&appnote=en012031;
[4] Petr Kolomaznik, EHL elektronika, PIC downloader v1.08, http://www.ehl.cz;
[5] Herman Aartsen, TNO - The Netherlands, PIC Bootloader +, http://www.microchipc.com/

Nova križna stikala so idealna za aplikacije z več V/I priključki

www.analog.com

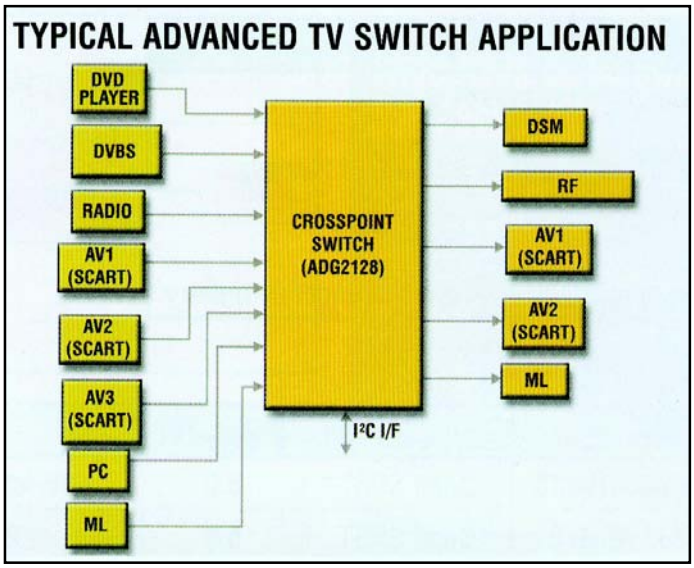
Za aplikacije z velikim številom vhodnih in izhodnih signalov je pomembna zmožnost učinkovitega preklapljanja več vhodov na več izhodov. Nova družina križnih stikal ADG2128 prinaša matrice 8 x 8 do 12 x 8 s krmiljenjem prek vodila I2C® v izjemno majhnem ohišju LFCSP 5 mm x 5 mm. Izjemna procesna tehnologija ADI poleg tega zagotavlja najboljše parametre delovanja za ta razred naprav, kar pomeni, da je mogoče družino ADG2128 uporabljati za aplikacije, kot je preklapljanje signalov AV v televizorjih, nadzornih videonapravah in avtomobilskih medijskih sistemih.

Funkcije:

- izredno nizek presluh: -70 dB pri 5 MHz,
- v razredu vodilna izolacija,
- dvosmerni signali,
- najmanjša velikost,
- najmanjša poraba,
- nadzor z vodilom I2C pri 400 kHz (na voljo je tudi 3,4 MHz),
- podpora za 2 Vrms pri VDD >8 V.

Aplikacije:

- Avtomobilski medijski sistemi
- Preklapljanje signalov AV
- Televizorji
- Domača medijska omrežja
- Videonadzorna oprema
- Preklapljanje signalov KVM
- Preizkusna oprema
- Ultrazvočni sistemi ●



Oznaka	Konfiguracija	Izolacija (5 MHz)	Presluh (5 MHz)	R_{ON} (Ω)	I_{DD} (μA)	Temperaturna območja	Ohišje (mm x mm)	Cena ($)
DG2108	10 x 8	**-65 dB**	**-70 dB**	45	**2**	Industrijsko, avtomobilsko	5 x 5 LFCSP	4,68
DG2128	12 x 8	**-65 dB**	**-70 dB**	45	**2**	Industrijsko, avtomobilsko	5 x 5 LFCSP	5,62
DG2188	8 x 8	**-65 dB**	**-70 dB**	45	**2**	Industrijsko, avtomobilsko	5 x 5 LFCSP	3,75