# UVOD V PROGRAMIRANJE MIKROKRMILNIKOV

TILITI.

Učni vodnik in navodila za vaje

Janez Pogorelc UM-FERI, 2005



Univerza v Mariboru Fakulteta za elektrotehniko, računalništvo in informatiko



http://www.ro.feri.uni-mb.si http://www.hipulab.uni-mb.si/

http://www.feri.uni-mb.si



# Naslov: UVOD V PROGRAMIRANJE MIKROKRMILNIKOV, Učni vodnik in navodila za vaje

Prva izdaja, marec 2005 Avtor: višji predavatelj mag. Janez Pogorelc, univ. dipl. inž. Strokovni recenzent: doc. dr. Martin Terbuc, univ. dipl. inž., Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko Vrsta učnega gradiva: zbrana gradivo za predavanja Uredil in oblikoval: Janez Pogorelc Slike: Janez Pogorelc Oblikovanje naslovnice za CD zgoščenko: Janez Pogorelc Naklada: 11 izvodov CD zgoščenk Obseg: 31 strani Izdala: Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Inštitut za robotiko Tisk: Janez Pogorelc ISBN 86-435-0695-8

## Kazalo

1.	PRE	DSTAVITEV MPU-PIC16F876 UČNEGA KOMPLETA	6
	1.1.	MODUL MPU-PIC16F876	7
	1.2.	OPOZORILO O MOŽNOSTI PREGREVANJA IN NEVARNOSTI DOTIKA	8
	1.2.1	. Kratek opis ukazov mikrokrmilnika PIC 16F87x	
	1.2.2	. Podroben prikaz segmentne organizacije podatkovnega pomnilnika	10
2.	UPO	RABA PROGRAMSKEGA ORODJA MICROCHIP MPLAB	11
2	2.1.	KONFIGURIRANJE MPLAB PROGRAMSKEGA OKOLJA	11
2	2.2.	NALAGANJE PROGRAMA V CILJNI SISTEM IN TESTIRANJE	
3.	UPO	RABA PROGRAMSKEGA JEZIKA C V MPLAB OKOLJU	16
3	3.1.	Konfiguriranje C-prevajalnika	16
3	3.2.	PREVAJANJE IN TESTIRANJE PROGRAMA	19
4.	VPR/	AŠANJA ZA UTRJEVANJE IN UČNI VODNIK	
2	4.1.	PREDSTAVITEV MPU-PIC16F876 UČNEGA KOMPLETA	
2	4.2.	PROGRAMIRANJE PIC16F87X V ZBIRNEM JEZIKU	
4	4.3.	ORGANIZACIJA POMNILNIKA	
2	4.4.	OSNOVNE VHODNO/IZHODNE ENOTE	
4	4.5.	ANALOGNI VHODI	
4	4.6.	CASOVNIK TIMERO IN PERIODIČNO GENERIRANJE PREKINITEV	
4	4.7.	GENERIRANJE PULZNO-ŠIRINSKIH SIGNALOV (PWM)	
5.	DOD	ATEK	
Ę	5.1.	PRIMER POPOLNEGA ZGLEDA PROGRAMA V C-JEZIKU ZA MPU-16F876	
6.	LITE	RATURA	

#### Seznam slik

Slika 1-1: Fotografija MPU PIC 16F876 kompleta	6
Slika 1-2: Blokovna shema MPU PIC 16F876 modula	7
Slika 1-3: Možnost pregrevanja MPU PIC 16F876 kompleta	8
Slika 2-1: Okno za »nov projekt«	11
Slika 2-2: Okno »urejanje lastnosti projekta«	12
Slika 2-3: Okno za izbiro razvojnega okolja	12
Slika 2-4: Okno za izbiro COM vrat ICD modula	13
Slika 2-5: Okno za izbiro/vpis imena izvorne datoteke	13
Slika 2-6: Okno za nastavitev možnosti delovanja ICD	14
Slika 2-7: Okno za »razhroščanje« (kontrolirano izvajanje programa	15
Slika 3-1: Okno za vključitev HI-TECH PICC Lite prevajalnika	16
Slika 3-2: Okno za pripravo projekta	17
Slika 3-3: Okno za izbiro ICD orodja in čip PIC16F877	17
Slika 3-4: Okno za nastavitev opcij C-prevajalnika (1. del)	18
Slika 3-5: Okno za nastavitev opcij C-prevajalnika (2. del)	18
Slika 3-6: Okno, kjer testno izvajamo program	19
Slika 3-7: Okno z izpisom statusa C-prevajalnika	20

### Seznam tabel

Tabela 1-1: Tabela ukazov s kratkimi opisi	9
Tabela 1-2: Razporeditev registrov po pomnilniških segmentih	10

#### Seznam programov

Program 5-1: Popoln zgled programa v C-jeziku – dvotočkovna histe	erezna regulacija
temperature	30

# Namen in cilji učnega gradiva

Učno gradivo predstavlja dopolnitev »zbranega gradiva za predavanje« z naslovom **Uvod v programiranje mikrokrmilnikov** in sicer kot učni vodnik z vprašanji in odgovori za utrjevanje ter priročnik za uporabo programskih orodij Microchip MPLAB in C-prevajalnika za programiranje 8-bitnih PIC mikrokrmilnikov srednje kategorije. Namenjeno je predvsem študentom 2. letnika visokošolskega strokovnega programa Elektrotehnika, smer avtomatika pri podpori vajam za predmeta Mikroelektronika in Mikroračunalniški sistemi.

V uvodu je na kratko predstavljen mikrokrmilniški učni komplet **MPU-PIC16F876**, ki je bil razvit na Inštitutu za robotiko UM-FERI za podporo pedagoškemu procesu. Sledijo tabela z opisom vseh ukazov zbirnega jezika in tabela s prikazom datotečnih registrov.

V nadaljevanju sledi podroben opis dela z integriranim programirnim okoljem Microchip **MPLAB V5.70** v kombinaciji z razvojnim kompletom **ICD(1)** ter napotki za konfiguriranje orodja za programiranje bodisi v zbirnem jeziku, bodisi v C-jeziku.

Del gradiva je namenjen tudi »učnemu vodniku«, napotkom za iskanje dodatnih virov ter vprašanjem za utrjevanje po poglavjih, kot je to širše opisano v »zbranem gradivu za predavanje« z naslovom **Uvod v programiranje mikrokrmilnikov.** 

Za nevešče uporabnike programskega jezika C, je priporočljiv predhodni obisk tečaja iz osnov programiranja v ANSI C jeziku, ali da se naučijo osnov iz ustreznih priročnikov.

Zbrano gradivo je primerno tudi za tiste, ki uporabljajo druge tipe Microchip PIC mikrokrmilnikov srednje kategorije (s 14-bitno programsko besedo), npr.: 16F84, 16F877A in drugi.

V učnem gradivu je izpis popolnega (delujočega) programa v C jeziku. Več zgledov kot tudi opisov študentskih projektov, ki temeljijo na uporabi PIC mikrokrmilnikov, je na voljo uporabnikom na spletnih straneh predmetov Mikroelektronika in Mikroračunalniški sistemi v okviru spletišča Inštituta za robotiko (<u>http://www.ro.feri.uni-mb.si/izobrazevanje/</u>).

Pričujoče učno gradivo je bilo oblikovano »po merilih za e-učna gradiva« iz **Poslovnika založniške dejavnosti v e-izobraževanju na Univerzi v Mariboru** (aktivno kazalo, nadpovezave na poglavja, slike, tabele in HTML reference).

# 1. Predstavitev MPU-PIC16F876 učnega kompleta



Slika 1-1: Fotografija MPU PIC 16F876 kompleta

Mikrokrmilniški učni komplet MPU-PIC 16F876 je sestavljen iz treh modulov (Slika 1-1):

- MPU-PIC16F876 modul
- ICD(1) modul
- MPU-GREL aplikacijski modul

## 1.1. Modul MPU-PIC16F876

Zgradbo modula in povezavo vhodno/izhodnih naprav na priključke vrat prikazuje blokovna shema (Slika 1-2):



Slika 1-2: Blokovna shema MPU PIC 16F876 modula

Modul **MPU-PIC16F876** je bil razvit na UM-FERI, Inštitutu za robotiko za potrebe učenja osnov programiranja in uporabe PIC mikrokrmilnikov [4]. Zgrajen je na osnovi mikrokrmilnika **Microchip PIC16F876.** Na kartici so na voljo tudi:

- 4 enote LED indikatorjev;
- 4 enote 7-segmentni LED prikazovalnik;
- 4 tipke;
- rele z delovnim in mirovnim kontaktom;
- 14-polni razširitveni konektor (priključki za analogne vhode, PWM izhod, časovniki, ...) (J4);
- 10-polni konektor za priključitev ICD(1) modula (J1);
- konektor za povezavo z osebnim računalnikom (RS232C oz. COMx vrata);
- 14-polni konektor za priključitev LCD prikazovalnika (J3).

### 1.2. Opozorilo o možnosti pregrevanja in nevarnosti dotika

<u>OPOZORILO:</u> Izhod **RA5 (LD4)** je priključen na krmilni vhod grelnega tranzistorja. Če je **LED4 vključena več kot minuto,** lahko temperatura na hladilnem telesu naraste preko 60 °C. V takšnem primeru je najbolje **prekiniti program** (Reconnect na ICD) ali **izklopiti napajanje**.



Slika 1-3: Možnost pregrevanja MPU PIC 16F876 kompleta

#### 1.2.1. Kratek opis ukazov mikrokrmilnika PIC 16F87x

ukaz,	operand	opis ukaza	STATUS	cikli		
UKAZI, KI SO POVEZANI Z ZLOGI (BYTE)						
ADDW	Ff, d	seštej <b>W</b> in register <b>f</b>	C, DC, Z	1		
ANDW	Ff, d	»logična in« operacija med W in registrom f	Z	1		
CLRF	f	register <b>f</b> naj dobi vrednost 0	Z	1		
CLRW	-	register <b>W</b> naj dobi vrednost 0	Z	1		
COMF	f, d	eniški komplement registra f (zamenjava 0 in 1)	Z	1		
DECF	f, d	zmanjšaj za ena vsebino registra f	Z	1		
DECFS	Z f, d	zmanjšaj vsebino reg. f za ena in preskoči naslednji ukaz, če f=0	-	1(2)		
INCF	f, d	povečaj za ena vsebino registra f	Z	1		
INCFS	Zf,d	povečaj vsebino reg. f za ena in preskoči naslednji ukaz, če f=0	-	1(2)		
IORW	f, d	»logična ali« operacija med W in registrom f	Z	1		
MOVF	f, d	kopiraj vsebino registra f (d=0 ->W; d=1 ->f)	Z	1		
MOVW	Ff	kopiraj vsebino W v register f	-	1		
NOP	-	ne naredi ničesar	-	1		
RLF	f, d	krožno premakni vse bite v registru f za eno mesto v levo	С	1		
RRF	f, d	krožno premakni vse bite v registru f za eno mesto v desno	С	1		
SUBWI	F f, d	odštej vsebino reg. W od vsebine v reg. f (d=f-W)	C, DC, Z	1		
SWAPE	f, d	zamenjaj zgornje in spodnje 4 bite v registru f	-	1		
XORW	Ff, d	»izključno ali« logična operacija med W in reg. f	Z	1		
UKAZI	, KI SO P	OVEZANI Z BITI				
BCF	f, b	briši bit b v registru <b>f</b>	-	1		
BSF	f, b	postavi bit b v registru f	-	1		
BTFSC	f, b	testiraj bit b v registru f in preskoči naslednji ukaz, če je bit b=0	-	1(2)		
BTFSS	f, b	testiraj bit b v registru f in preskoči naslednji ukaz, če je bit b=1	-	1(2)		
SISTER	NSKI IN U	JKAZI, KI SO POVEZANI S KONSTANTAMI				
ADDLV	Vk	registru <b>W</b> prištej konstanto <b>k</b>	C, DC, Z	1		
ANDLV	Vk	»logična in« operacija med <b>W</b> in konstanto <b>k</b>	Z	1		
CALL	k	klic podprograma	-	2		
CLRW	DT -	vsebina »časovnega stražnika« (Watch Dog Timer) naj bo enaka nič	TO, PD	1		
GOTO	k	skoči na naslov k (11-bitni naslov)	-	2		
IORLW	/ k	»logična ali« operacija med <b>W</b> in konstanto <b>k</b>	Z	1		
MOVL	N k	vpiši konstantno vrednost k v W (W=k)	-	1		
RETFIE	-	vrni se iz podprograma za strežbo prekinitvene zahteve	-	2		
RETLW	/ k	vrni se iz podprograma s konstantno vrednostjo k v W	-	2		
RETUR	N -	vrni se iz podprograma	-	2		
SLEEP	-	postavi mikrokrmilnik v stanje mirovanja - »sleep« način	TO, PD	1		
SUBLW	/ k	odštej vsebino <b>W</b> od konstantne vrednosti k ( <b>W</b> =k- <b>W</b> )	C, DC, Z	1		
XORLV	Vk	»izključni ali« logična operacija med <b>W</b> in konstanto <b>k</b>	Z	1		

Tabela 1-1: Tabela ukazov s kratkimi opisi

f - ime ali naslov datotečnega registra (file register)

k - konstantna (8-bitna) vrednost med 0 in 255

d - cilj hrambe rezultata operacije (d=0 - shrani se v W, d=1 - shrani se v f)

b – pozicija ali oznaka bita med 0 in 7

1(2) – trajanje ukaza je 1 (ali 2) ukazna cikla (pri 20 MHz taktu je trajanje enega ukaznega cikla (4\*50) ns = 0,2  $\mu s)$ 

# 1.2.2. Podroben prikaz segmentne organizacije podatkovnega pomnilnika

	naslov registra		naslov registra		naslov registra		naslov registra
Indirect addr (*)	00b	Indirect addr (*)	80b	Indirect addr (*)	100h	Indirect addr (*)	1805
TMR0	01h	OPTION REG	81h	TMR0	101h	OPTION REG	181h
PCI	02h	DCI	82b	PCI	102h	PCI	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
ESR	04h	FSR	84h	ESR	104h	FSR	184b
PORTA	05h	TRISA	85h		105h	101	185h
PORTR	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h	TORTE	107h	11100	187h
PORTD <sup>(1)</sup>	08h	TRISD <sup>(1)</sup>	88h		108h		188h
PORTE <sup>(1)</sup>	09h	TRISE <sup>(1)</sup>	89h		109h		189h
PCLATH	0Ah	PCLATH	84h	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	FEDATA	10Ch	EECON1	18Ch
PIR2	ODh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved <sup>(2)</sup>	18Eh
TMR1H	0Fh	10011	8Eh	FEADRH	10Fh	Reserved <sup>(2)</sup>	18Eb
T1CON	10h		90h	LEADIGH	110h	1100011001	190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	926		112h		192h
SSPBUE	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1I	15h	0010171	95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		976	General	117h	General	197h
ROSTA	18h	TXSTA	98h	Purpose	118h	Purpose	198h
TYREG	19h	SPBRG	99h	16 Bytes	119h	16 Bytee	199h
RCREG	1Ah	GFBRG	аль 	10 Dytes	11Ah	To Dyies	194h
CCPR2	1Bh		ORL		11Bh		19Bh
CCPR2H	1Ch		9Ch		11Ch		19Ch
CCP2CON	1Dh		ann		11Dh		19Dh
ADRESH	1Eh	ADRESI	aEb		11Eh		19Eh
ADCONO	1Eh	ADCON1	9Eh		11Fh		19Eh
ADCONO	20h	ADCONT			120h		1401-
	2011		A0h		12011		1A0h
General		General		General		General	
Purpose		Purpose		Purpose		Purpose	
Register		Register		Register		Register	
96 Bytes		80 Bytes	EFh	80 Bytes	16Fh	80 Bytes	1EFh
		accesses	F0h	accesses	170h	accesses	1F0h
		70h-7Fh		70h-7Fh		70h - 7Fh	
Bk A	7Fh	Bank 4	FFh	Baals 2	17Fh	Bask 2	1FFh
Bank U		Dank I		Darik Z		Darik S	
neuporal * ni fizični	oljene lok rogistor	acije, čitajo se	kot '0'				
Onomba 1. ti registri se ne unorabliaje v DIC16E976/2 ampak v DIC16E977/4							
					V FICIOF	ד// יט	
Spomba 2: ti i	egistri so	rezervirani, vre	eunost je	U			

Tabela 1-2: Razporeditev registrov po pomnilniških segmentih

# 2. Uporaba programskega orodja Microchip MPLAB

Spoznali boste programska orodja integriranega programskega okolja Microchip MPLAB. Naučili se boste kreirati projekt, konfigurirati ICD ali Simulator ter sprožiti postopek generiranja izvršljivega programa za PIC mikrokrmilnik.

**Opomba:** Opis se nanaša na programsko orodje **Microchip MPLAB V5.70** (zadnje različice, ki še omogoča delo v kombinaciji z **ICD(1)**. V kolikor ima uporabnik možnost dela z **ICD2** (ali naprednejšim razvojnim sistemom), je priporočljiva uporaba novejše različice **MPLAB** iz serije **V6.X**.

V nadaljevanju bo podrobno opisan postopek od kreiranja projekta do stopnje, ko lahko program izvajamo na ciljnem mikrokrmilniku (npr.: MPU-PIC16F876) ali kar na osebnem računalniku z orodjem MPLAB Simulator.

Modul **MPU-PIC16F876** povežite z modulom **ICD**. Pazite, da bo kabel ustrezno priključen (glejte napise!). Nato modul **ICD** povežite z računalnikom preko serijskega vodila (COM1 ali COM2, lahko tudi COM4). Nazadnje priključite še napajanje (AC ali DC, 7,5 – 15 V).

Poženite program [10][9][16] *Microchip MPLAB* (*Start*  $\rightarrow$  *Programs*  $\rightarrow$  *Microchip MPLAB*  $\rightarrow$  *MPLAB* ) in prikaže se osnovno okno MPLAB programirnega okolja.

#### 2.1. Konfiguriranje MPLAB programskega okolja

Odprite (Slika 2-1) novi projekt (*Project*  $\rightarrow$  *New Project* ...); projekt imenujte poljubno; obvezna je končnica **.pjt**, pot (direktorij) odprete v svojem delovnem imeniku:

File <u>N</u> ame:	Directories:	OK
vaja1.pjt	d:\vaje\mplab	Cancel
deb.pjt	🔄 🔄 d:\	A
test.pjt	🔄 🔄 vaje	Help
	mplab	
	prog	
		7
	<u> </u>	_
ist Files of <u>T</u> ype:	Dri <u>v</u> es:	
Project Files (*.pjt)	▼ 🗐 d:	-

Slika 2-1: Okno za »nov projekt«

Pritisnite OK, odpre se (Slika 2-2) okno *Edit Project*:

Edit Project Project Target Filename vaja1.hex Include Path Library Path Linker Script Path	Cancel Help	izbereš lahko tudi: MPLAB-ICD16F876
Development Mode: MPLAB-ICD16F877 Language Tool Suite: Microchip Project Files vaja1 [.hex]	Change Change Copy Node Delete Node Build Node de Properties	pomeni izbiro progr. orodja: MPLAB - Assembler

Slika 2-2: Okno »urejanje lastnosti projekta«

Ustvari se datoteka s končnico **.hex**, z imenom projekta npr.: vaja1.hex (tovrstna datoteka predstavlja preveden program).

Kliknite *Change* ..., odpre se (Slika 2-3) okno *Development Mode*, kjer nastavite opcije, kot so prikazane na spodnji sliki:



Slika 2-3: Okno za izbiro razvojnega okolja

Kliknete **OK** in prikaže se (Slika 2-4) okno **MPLAB-ICD**:

#### 2. Uporaba programskega orodja Microchip MPLAB

Uvod v programiranje mikrokrmilnikov, učni vodnik in navodila za vaje



Slika 2-4: Okno za izbiro COM vrat ICD modula

To okno naj bo ves čas prisotno na zaslonu, prestavimo ga lahko npr.: desno zgoraj.

Po potrebi izberemo ustrezna COMx vrata. Če komunikacija med **ICD** modulom in **MPLAB** okoljem ni vzpostavljena, utripa LED indikator na **ICD**. Komunikacijo poskušamo vzpostaviti s klikom na gumb **Reconnect**. Če LED indikator še vedno utripa, je potrebno izbrati druga COMx vrata.

<u>Opomba:</u> Na namiznih osebnih računalnikih so na voljo običajno vrata COM1 ali COM2; na prenosnih računalnikih (brez vgrajenih serijskih vmesnikov) z **dodanim USB vmesnim pretvornikom so to običajno vrata COM4.** 

#### 2.2. Nalaganje programa v ciljni sistem in testiranje

V oknu *Edit Project* kliknete še *Add Node* in vključite datoteko (Slika 2-5), v kateri se nahaja program s končnico **.asm** (izvorni program v zbirnem jeziku, npr.: vaja1.asm, le-to lahko kasneje urejate v vključenem urejevalniku **MPLAB Editor**):

Add Node		? ×
File <u>n</u> ame: vaja1  asm deb.asm example.asm example2.asm led.asm light.asm test.asm test1.asm test2.asm	Eolders: d:\vaje\mplab @ d:\ @ vaje @ mplab @ prog	OK Cancel <u>H</u> elp N <u>e</u> twork
List files of <u>type</u> : Source files (*.c;*.asm)	Folders: OK   d:\vaje\mplab Cancel   d:\ Image:	

Slika 2-5: Okno za izbiro/vpis imena izvorne datoteke

Nato v oknu *Edit Project* kliknete OK (s tem se zapre).

V oknu **MPLAB-ICD** kliknete **Options** ... in v oknu **ICD Options** nastavite opcije (v desni polovici), kot to kaže (Slika 2-6).

R ICD Options		_ 🗆 ×		
Configuration Bits				
Device:				
Oscillator:	HS	•		
Watchdog Timer:	Off/Disable	•		
Power Up Timer:	Off/Disable	•		
Brown out Detect:	Off/Disable	•		
Low Voltage Program:	Disable	•		
Code Protect Data EE:	code protect	ion Off 📃 🗾		
Flash Memory Write:	Memory writte	en to by EECON 📃		
Code Protect:	Code Protect	ion Off 📃 🗾		
ID's and Checksum         Checksum:       0x1B39<3F39>         ID Code:       010B0309         IV Use Checksum as ID         Voltages         VDD:       Update         VPP:       Update	Program Start Add End Add V Prog V Confi I D Lo te EEPI V Eraso	Options dress: 0x0000 dress: 0x1EFF ram Memory iguration Bits bocations ROM Data e All before Program ble Debug Mode	možnost	iz
Blank Read Progra	am Verify	Erase Def. Addr	Debug N	Ло
Download ICD Operating	g System	Self Test		

Slika 2-6: Okno za nastavitev možnosti delovanja ICD

Okno ICD Options lahko zaprete (desni kot zgoraj).

Zatem odprete novo datoteko (*File*  $\rightarrow$  *New*), napišete (ali modificirate) izvorni program v zbirnem jeziku (Assembler) ter ga shranite s *File*  $\rightarrow$  *Save as* ... (v našem primeru vaja1.asm).

Program bo tako napisan v zbirnem jeziku (Assembler) in vsebuje obvezno končnico **\*.asm**.

- Prevajanje (aktiviranje zbirnika) izvedete tako, da v osnovnem oknu MPLAB menuju Project izberete opcijo Build All.
- 2. Program naložite tako, da v oknu *MPLAB-ICD* kliknete *Reconnect* in nato *Program*.
- 3. **Program poženete** (sprostite RESET) tako, da **še enkrat** sprožite prevajanje (*Project* → *Build All*).
- 4. Delovanje programa prekinete tako, da v oknu *MPLAB-ICD* kliknete *Reconnect*.

<u>Opomba:</u> Če izberete možnost *Enable Debug Mode* (v oknu *ICD Options*), ne izvajate točk 3 in 4, kajti MPLAB omogoča kontrolirano izvajanje programa preko ikon ali menijev v MPLAB okolju (*Debug*  $\rightarrow$  *Run*  $\rightarrow$  ...).

Možnosti: start programa (Run F9), ponovni start (Reset F6), ustavitev (Halt F5),

izvajanje po korakih (**Step** F7), vstavljanje prekinitvenih točk (**Break Settings** F2), sledenje registrov (**SFR**) in spremenljivk (**Watch Windows**).

DE - D:	\ARHIV\PICMPLAB\DEMOTEST.PJ1							
Edit	Debug PICSTART Plus Options Tools	<u>W</u> indow <u>H</u> elp	😭 MPLAB ICD	Vetsion 1	44.00			_ 🗆 🗙
	<u>B</u> un )	<u>R</u> un F9	Status: Step					
	Execute	Reset F6	COM1 - 576	OC 🔹 🖬 All I	Register	rs .	▼ 10MH	1z-20MI
	Simulator Stimulus	<u>H</u> alt F5	Reconnect	Program	n 0	otions F/w	: Ver 2.04.0	
	Center Debug Location	Halt Trace Shift+F5 Animate Ctrl+F9	Special Fu	nction Be	aister V	lindow		
liv\pic	Break Settings F2	Step F7	SER Name	Hov	Dec	Binaru	Char	
10000	Trace Settings	Step <u>O</u> ver F8	W	DE	223	11011111	onar	-
120018	Trigger In/Out Settings	Update All Registers	tmr0	61	97	01100001	a	
	Trigger Output Points	Change Program Counter	option	FF	255	11111111		
; I	Clear All Points	1	pcl	AB	160	10100000		
, hsf		•hanka 1	pclath	00	0	00000000		
mou	Complex Trigger Settings	u delouni register n	status	10	28	00011100	×	
mov	Code Coyerage	urednost Ax6 unise u	fsr	FF	255	11111111	1	
	Class Disease Manager CMI Chill F2	:(PORTA pini so digit	porta	00	6	00000000	1	
mov	Liear Program Memory Ltri+Shirt+F2	:0 - izhod. 1 - vhod	trisa	1F	31	00011111		
mov	Bystem Reser Lin+Shitt+F3	vrednost 0x10 voise	portb	00	0	00000000	*	
clr	Power-On-Reset Ctrl+Shift+F5	PORTR v celoti izhod	trisb	CO	192	11000000		
clrf	TRISC	PORTC v celoti izhod	portc	FØ	240	11110000		
	(2.2.2.7.7.7.7.)		trisc	00	0	00000000		
hcf	STATUS RP0	∙hanka Ø	intcon	01	1	00000001	1	
	7.83		pir1	00	0	00000000	1	
clrf	PORTA	izhris PORTA	pie1	00	0	00000000	14	
clrf	PORTB	izbris PORTB	pir2	00	0	00000000		
mov1	w 0xf0		pie2	00	0	00000000	×	

Slika 2-7: Okno za »razhroščanje« (kontrolirano izvajanje programa

<u>Opozorilo:</u> »Razhroščevalni način« izvajanja programa (*Debug Mode*) ima določene pomanjkljivosti: zasede se del RAM in ROM pomnilnika, za komunikacijo med ICD1 modulom in MPU-PIC16F876 se uporabljata liniji RB6 in RB7 (pike in črtice na LED prikazovalnikih MPU-PIC16F876 modula se naključno spreminjajo, vsebina prikazanih registrov ne ustreza vedno dejanskemu stanju, program ne teče v realnem času!).

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov programiranja v zbirnem jeziku, kar je na voljo med drugim v [2][4][6][9]. **Pomembna lastnost** Microchip MPLAB programskega okolja je, da lahko **programe izvajamo** (z določenimi omejitvami) tudi v **simulacijskem načinu** (brez uporabe ciljnega sistema) zgolj na osebnem računalniku povprečnih zmogljivosti [10].

#### Vprašanja za utrjevanje:

- 1. Kakšno končnico ima datoteka izvorni program v zbirnem jeziku?
- 2. Kakšno končnico ima datoteka strojni program v zbirnem jeziku?
- 3. S katerim orodjem pretvarjamo izvorne programe v zbirnem jeziku v strojne programe ?
- 4. Kam se naloži program pri uporabi MPLAB-ICD modula?
- 5. Kakšne možnosti pri testiranju programa imamo, če izberemo «ICD Debug Mode«?
- 6. Kakšne omejitve pri testiranju programa imamo, če izberemo «ICD Debug Mode«?

# 3. Uporaba programskega jezika C v MPLAB okolju

Spoznali boste možnost uporabe C-prevajalnika v programskem okolju Microchip MPLAB. Naučili se boste kreirati projekt, konfigurirati ICD ali Simulator ter sprožiti postopek generiranja izvršljivega programa za PIC mikrokrmilnik.

Na voljo je **več različic C prevajalnikov** različnih proizvajalcev. Za nekomercialno uporabo je zelo primeren **HI-TEC C** [11] prevajalnik **PICC Lite** (v nadaljevanju bo opisana različica: [13][14] *HI-TECH PICC Lite Compiler Release notes* v8.02PL1).

To je **popolnoma delujoč C prevajalnik z nekaj** (manj pomembnimi) **omejitvami**: omogoča le 16F877 ali 16F84, dolžina programa največ 2 kW, možnost uporabe pomnilnika za spremenljivke samo v prvih dveh segmentih ter izpisi števil v *float* in *long* formatu niso možni.

### 3.1. Konfiguriranje C-prevajalnika

Program se po instalaciji naloži (privzeto) na disk C: v imenik C:\PICCLITE. Samodejno se integrira v IDE okolje MPLAB. To preverimo (Slika 3-1) v oknu: (*Project*  $\rightarrow$  *Install Language Tool*).

Install Language Tool X						
Language Suite:	HI-TECH PICC Lite					
Tool Name:	IAR IAR PIC18					
Executable:	microEngineering Labs Inc HI-TECH PICC Lite					
Command-line C Windowed						
OK Cancel Help						

Slika 3-1: Okno za vključitev HI-TECH PICC Lite prevajalnika

V meniju (*Project*  $\rightarrow$  *New... ali Edit Project...*) odpremo (Slika 3-2) ali modificiramo projekt \*.pjt, pri čemer najprej izberemo Language Tool Suite: HI-TECH PICC Lite. Zatem obvezno izberemo PIC16F877. V okviru razvojnega okolja (Slika 3-3), lahko izbiramo med ICD razvojnim okoljem in med MPLAB SIM simulatorjem).

Edit Project	×
Project Target Filename	ОК
Include Path	Cancel
Library Path	Help
Linker Script Path	
Development Mode: MPLAB ICD PI	C16F877 Change
Language Tool Suite: HI-TECH PICC	Lite 🗾
Project Files	
tst_Id2 [.c]	Add Node
	Copy Node
	Delete Node
	Build Node
	Node Properties

Slika 3-2: Okno za pripravo projekta

Development Mode		2	×	
Configuration Power Tools Ports	Pins Clock	Break Options		
<ul> <li>None (Editor Only)</li> <li>MPLAB SIM Simulator</li> <li>MPLAB ICE Emulator</li> </ul>	Processor: PIC16F877 Requires OnChip reso debugging. Only one t may be set.	urces for preakpoint	na n P	amesto PIC16F876 noramo izbrati: IC16F877
<ul> <li>PICMASTER Emulator</li> <li>ICEPIC Emulator</li> <li>MPLAB ICD Debugger</li> </ul>	ICD supports 16Fxxx p only. ICD 2 supports 1 Inquire	Details		
OK	Cancel Ar	pply <u>H</u> elp		

Slika 3-3: Okno za izbiro ICD orodja in čip PIC16F877

Označimo datoteko s končnico [.hex] in odpremo okno (Slika 3-4) Node Properties.

de Properties				
Node: TST_LD2	HEX	▼ Lange	uage Tool: PICC Lite (	Compiler 🗨
Description	I	1		Data 🔺
Informational messages	🗐 Quiet	Verbose 🗐		
Warning level	🔳 On			
Generate debug info	🗹 On			
Assembler Optimization	🔳 On			
Global Optimizations	🔳 On			
Include search path	🔳 On			
Floating point for double	24-bit 🔳	🔳 32-bit		
Chars Are Signed	🔳 On			
Strict ANSI Conformance	e 🔲 On			
Define Macro	🔲 On			<b>_</b>
•				•
Command Line				
-FAKELOCAL -G -EpicC.err	-ASMLIST -16	F877		
Additional Command Line O	ptions			
		-		
	OK	Canc	el Help	

Slika 3-4: Okno za nastavitev opcij C-prevajalnika (1. del)

V oknu lahko izberemo **Generate Debug Info** (koristno pri testiranju z **ICD** v **Debug Mode** ali z **MPLAB SIM**), prav tako lahko izberemo: **Produce assembler listening** (izpis prevedenega C – programa v različici za zbirni jezik).

de Properties							
Node: TST_LD2.	HEX	✓ Lang	uage Tool: PIC	C Lite Comp	iler 🗾		
Description						Data	
Undefine Macro	🔳 On						
Error file	🗹 On				picC.err		
Append Errors to file	🔳 On						
Produce assembler list	🗹 On						
Generate prototypes	🔳 On						
Produce Pre-processed	🔳 On						
Compile to assembler c	🔳 On						
Specify identifier length	🔲 On						
Strip Local Symbols	🔲 On						
						Þ	-
Command Line -FAKELOCAL -G -EpicC.err	-ASMLIST -16F8	377					
Additional Command Line A	otions						_
[	OK	Cano	el	Help			

Slika 3-5: Okno za nastavitev opcij C-prevajalnika (2. del)

**Obvezno** izberemo rubriko **Error File** (Slika 3-5) in v koloni **Data** vpišemo ime datoteke (npr.: *picC.err*), kamor se bodo po prevajanju vpisovala sporočila o morebitnih napakah.

Zatem v oknu **Edit project** izberemo **Add node...**, ter vključimo ime izvornega programa v C jeziku npr.: *tst\_ld2.c*. Temu se takoj prilagodi tudi ime datoteke z izvršljivo kodo: *tst\_ld2.hex*.

#### 3.2. Prevajanje in testiranje programa

**Primer izpisa** (Slika 3-6) izvornega programa **\*.c** (levo), okno vsebine programskega pomnilnika s programom v strojni kodi (inverzni zbirnik) in okno s prikazom vsebine **SFR** registrov (desno).

MPLAB IDE - C:\ARHIV\PICMPLAB\TEST_LED\CC2\T5_LD2C.PJT										- 8 ×				
Elle Project Edit Debug PICSTART Plus Options	<u>T</u> ools <u>W</u> indo	w <u>H</u> elp						MPLAB I	CD Versio				- 🗆 ×	
	8 2 2		OM RAM	SFR 😎	1 🕎 1			Status: Reset						
						1	_	COM 🚽 576	0 🔽 All F	Registers	;	▼ 10M	Hz-20M	<b>_</b>
🖺 c:\arhiv\picmplab\test_led\cc2\tst_ld2.c					_ 🗆 ×			Reconnect	Program	i Opl	tions F/W:	Ver 2.31.	00	
/* Demo program za utripanje LED diod:	LD1 na RC@	) v taktu	ı 200m:	5 */		1								
#include <pic1687x.h> /* vkljucitev dat</pic1687x.h>	oteke z de	finicija	ami si	mbolov *,	, 🗌	]		Special	Function	Regist	er Window		- 🗆 ×	
unsigned int i;								SFR Name	Hex	Dec	Binary	Char		
void main(void)								w tmr0	00	0	000000000	1		
(								option	00 FF	255	00000000 11111111			
/* Inicializacija renistrov v	uhodno/izho	dnih vme	sniko	u */				pclath	00	0	00000000	1.1		
ADCON1=0x06; /* ne uporabljamo analo	ognih vhodo	v (0000	3110)	×/				status	00	8	00000000			
TRISA=0x20; /* vsi pini porta A so	izhodni, r	azen RAS	5 (001	00000) *	/			porta	00	8	00000000	- 1		
TRISC=0; /* USI pini porta c so	izhodni */	,						trisa	00	8	00000000			
PORTC=0; /* izhode porta C posta	avi na Ø							portb	00	0	00000000	-		
INTCON-0; /* prekinitev ne bomo u	ıporabljali	. */						trisb	00	0	00000000			
								trisc	66	6	00000000			
/* jedro programa */								portd	00	ő	00000000	- C - C		
{								trisd	00	8	00000000	-		
RC0=!RC0; /* izmenicni vklop l	.ED diode L	.D1 na RC	CØ */					porte	00	0	00000000			
for(i=0; i<58820; i++); /* zaka	asnitev: 58	820x3,4u	ıs=~20	Ons */				trise	00		00000000	-		
asm("test nop"); /* moznost nas	stav. Break	point na	a simb	. "test"	*/			nir1	66	6	88888888			
}								ie1	00	0	00000000	1.1		
}							>	x ir2	00	8	00000000	-		
					-	0x3		ie2	00	0	00000000			
						0×0	-	= mr11	00	8	000000000	-		
	3	0002	008A		novwf	0xA	-	-Icon	66	ñ	00000000	1		
	4	0003	2804		goto	exit		mr2	00	0	00000000	1.		
	5	0004	3020	exit	moviw	0x20		r2	00	8	00000000			
	7	0005	3822		nover	0x4 0x22		2con	00	0	00000000			
	8	0007	2000		call	clear ram		SPDUF	00	8	000000000			
	9	0008	0183		clrf	0x3 _		spcon2	66	ñ	00000000	1		
	10	0009	2FDB		goto	main		spadd	00	0	00000000	1.		
	11	808A	0604		xorw <del>f</del>	0x4,\		spstat	00	8	00000000	-		
	13	000B 000C	0180 0A84		incf	0x0 0x4		cpr11	00	0	00000000	-		
	14	000D	06 04	clear	xorwf	0x4,W		CPP 1n	00		00000000			
	15	000E	1D 03	-	btfss	0x3,0x2		csta	00	0	00000000	- 1		
	16	000F	280A		goto	0xA		xreq	00	0	00000000	1.1		
	1/	0010	3400		retiw	0x 0 0x 0		creg	00	8	00000000	-		
	19	8812	3FFF		wroos wfbhs	Axff		xsta	00	0	00000000			
	20	0013	3FFF		addlw	Øxff		porg	00	9	000000000	-		
	21	0014	<b>3FFF</b>		addlw	Øxff		cpr2h	66	6	00000000	- 1		
	22	0015	3FFF		addlw	Øxff		cp2con	00	õ	00000000	- 2		
	23	0016	3FFF		addlw	UXFF		dresh	00	8	00000000	-		
	24	0017	SEFE		addlw addlw	0xff Øyff		dresl	00	0	00000000	-		
	26	0019	3FFF		addlw	Øxff		dcon@	00	8	000000000			
	27	001A	<b>3FFF</b>		addlw	Øxff		con	66	6	00000000			
	28	001B	3FFF		addlw	Øxff		edata	00	ŏ	00000000	- C - C		
🕙 Build R 🗗 🗖 🗙	29	0010	3FFF 3FFF		addlw addlw	0xff 0vff		eadr	00	0	00000000	1.0		-
Lp 1 Col 1 26 W/B No Wrap INS PIC165977	nc:Dvff		dele	Bk On ICD		UNTT		ledath	88	ß	нааааааа			
	potown j	1.0/00 11 2 0		or on hop h	1.112 038									

Slika 3-6: Okno, kjer testno izvajamo program

Program v C jeziku prevedemo z ukazom: (*Project* → *Build all*).

V kolikor ni sintaktičnih napak, se izpiše sporočilo o lastnostih prevedenega programa (Slika 3-7) z zaključno vrstico: »Build completed successfully«. Nadaljujemo z izvajanjem programa (simulator) ali z nalaganjem (Download) v Flash ROM ciljnega mikrokrmilnika (enako kot pri programiranju z Microchip zbirnikom) s pomočjo ICD (*Reconnect, Program*). Če v oknu Options... nimamo izbran Debug Mode, še enkrat sprožimo *Build all*, kar povzroči start programa (sprostitev Reseta).

Če se izpiše sporočilo »Build failed«, je potrebno odpreti in proučiti datoteko (npr.: *picC.err*) s sporočili o napakah, le-te odpraviti in ponavljati zgornji postopek, dokler nismo odpravili vseh sintaktičnih napak v izvorni kodi.

1	MPLAB IDE - C:\ARHIV\PICMPLAB\TEST_LED\CC2\TS_LD2C.PJT				
Eile	e <u>P</u> roject <u>E</u> dit <u>D</u> ebug PIC <u>S</u> TIART Plus <u>O</u> ptions <u>T</u> ools <u>W</u> indow <u>H</u> elp	MPLAB I	CD Version 1		
•	a 🕫 🖬 🎗 🖇 🖺 🖬 👩 😭 🖫 🌠 🚛 🕅 🏧 🗺 🍸	Status: Reset			
		COM 🔽 576	0 💌 All Regis	sters	
,		Reconnect	Program	Options	. F/₩
1	Build Results			- 🗆 ×	:
#	<sup>i</sup> Building TST_LD2.HEX			-	low
u	<sup>IN</sup> Compiling TST LD2.C:				
	_Command line:""C:\PICCLITE\BIN\PICL.EXE -FAKELOCAL -G -EpicC.err -ASMLIST -16F877 C:\ARHIV\PICMF	LAB\TEST_L	ED\CC2\TST_	LD2.C	0000
ł	"HI-TECH PICC Lite COMPILER (Microchip PIC) V8.02 PL1 Conwright (C) 100h-2000 HI-IECH SOFENDORE				0000
	Copyright (C) 1904-2003 ni-icon Suriwine				1111
6	"Memory Usage Map:				0000
T	ν Πρεσακάτο DOM \$8888 - \$8818 \$8811 / 17) μοντές				0000
T	Program ROM S07DB - \$07FF \$0025 ( 37) words				0000
P	R \$0036 ( 54) words total Program ROM				0000
I	u Nashu a dami (1887a - (18871 (18882 ( - 2) butos total Bank a dami				9999
					0000
1	34 				0000
	Program statistics:				0000
	Total ROM used 54 words (2.6%)				0000
	Total RAM used 2 bytes (1.1%)				0000
					0000
	Build completed successfully.				0000
}	4			► /	/0000
	2 0001 3000 movlw 0x0	prez tmr11	00 88	0 000	ាចមមមម ពេតពតត

Slika 3-7: Okno z izpisom statusa C-prevajalnika

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov programiranja v ANSI C-jeziku, kar je na voljo med drugim v [11][5][13][18]. **Pomembna lastnost** Microchip MPLAB programskega okolja je, da lahko **programe izvajamo** (z določenimi omejitvami) tudi v **simulacijskem načinu** (brez uporabe ciljnega sistema) zgolj na osebnem računalniku povprečnih zmogljivosti [10].

#### Vprašanja za utrjevanje:

- 1. Kakšno končnico ima datoteka izvorni program v C-jeziku?
- 2. Kakšno končnico ima datoteka strojni program (preveden iz C-jezika)?
- 3. S katerim orodjem pretvarjamo izvorne programe iz C-jezika v strojne programe ?
- 4. Kam se naloži program pri uporabi MPLAB-ICD modula?
- 5. Kaj predstavlja (pri uporabi HI-TEC PICC Lite) izbira opcije »Generate Debug Info«?
- 6. Kateri čip moramo izbrati (pri uporabi HI-TEC PICC Lite), če imamo v ciljnem sistemu mikrokrmilnik »16F876«?

# 4. Vprašanja za utrjevanje in učni vodnik

Vsebina poglavja predstavlja dopolnitev »zbranega gradiva za predavanje« z naslovom **Uvod v programiranje mikrokrmilnikov** in sicer kot učni vodnik z vprašanji in odgovori za utrjevanje, razdeljenimi po poglavjih ter napotki za potrebna znanja s sugestijo za iskanje v dodatnih virih.

### 4.1. Predstavitev MPU-PIC16F876 učnega kompleta

Spoznali boste učni komplet MPU-PIC16F876 in osnovne lastnosti mikrokrmilnika PIC 16F876. Naučili se boste identificiranja vhodnih oziroma izhodnih priključkov mikrokrmilnika.

**Za popolno razumevanje tematike je potrebno tudi predznanje** iz osnov digitalne elektronike in gradnikov mikroprocesorskih sistemov, kar je na voljo med drugim v [3]. Dobri opisi zgradbe in uporabe sistemov s PIC mikrokrmilniki so tudi v [2] in [6].

Vprašanja za utrjevanje:

- 1. Kateri moduli tvorijo MPU-PIC16F876 komplet ?
- 2. Kakšen tip procesne enote je vsebovan v mikrokrmilniku PIC16F876?
- 3. Kakšni pomnilniški elementi so vsebovani v mikrokrmilniku PIC16F876 ?
- 4. Koliko vhodno/izhodnih priključkov ima mikrokrmilnik PIC16F876 ?
- 5. Katere funkcije opravlja ICD modul ?
- 6. Katere vhodne in izhodne signale vsebuje modul MPU-GREL ?

## 4.2. Programiranje PIC16F87x v zbirnem jeziku

Naučili se boste oblikovanja programa v zbirnem jeziku (splošni format) ter osnovne programske ukaze in njihov pomen. Ko boste obvladali nekatere podrobnosti, boste v nadaljevanju lahko uporabljali zgolj zgoščeno tabelo nabora vseh 35 ukazov.

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov digitalne elektronike (številski sistemi, logične operacije) [3] in osnov programiranja v zbirnem jeziku: [2] in [8]. Dobri opisi programskih ukazov PIC mikrokrmilnikov srednje kategorije so na voljo v [2] in [6] ter na spletnem portalu: <u>http://www.interq.or.jp/japan/se-inoue/e\_pic.htm</u>.

<u>Vprašanja za utrjevanje:</u>

S katerim ukazom postavimo (na 1) bite 0, 1 in 2 v delovnem registru W ?
Rešitev:

IORLW b'00000111' ;'ALI' log. oper. med istoleznimi biti, visjih 5 bitov v W se ohrani, ;spodnji trije biti v W se postavijo na 1

Koliko različnih ukazov je na voljo ?
Kakšna polja vsebuje programska vrstica v zbirnem jeziku ?
S katerim ukazom zbrišemo bit 3 v registru f na naslovu 0x2F ?
S katerim ukazom opravimo logični IN (AND) med registrom W in f na naslovu 0x22 ?
S katerim ukazom zaključimo podprogram ?

6. S katerim ukazom zbrišem bite 5, 6, in 7 v registru W?

### 4.3. Organizacija pomnilnika

Spoznali boste model in organizacijo podatkovnega pomnilnika (segmenti) in model programskega pomnilnika ter v zvezi s tem najpomembnejši datotečni register **STATUS**. Naučili se boste ločevati med podatkovnim pomnilnikom (RAM) in programskim pomnilnikom (FLASH ROM).

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov digitalne elektronike in gradnikov mikroprocesorskih sistemov (pomnilniški elementi), kar je na voljo med drugim v [3]. Dobri opisi zgradbe in opisa pomnilniškega modela PIC mikrokrmilnikov so tudi v [2] in [6] ter na spletnem portalu: <u>http://www.interq.or.jp/japan/se-inoue/e\_pic.htm</u>.

<u>Vprašanja za utrjevanje:</u>

Napišite zaporedje ukazov za izbiro segmenta Bank1 **Rešitev:** BCF STATUS,RP1 ; 0 → RP1 (brisi bit 6 v reg. STATUS) BSF STATUS,RP0 ; 1 → RP0 (postavi bit 5 v reg. STATUS)

- 1. Napišite zaporedje ukazov za izbiro segmenta Bank2
- 2. Kakšen pomen imajo zastavice v registru STATUS ?
- 3. Kateri pomnilniški segment (Bank0, 1, 2, 3) se največ uporablja?
- 4. Koliko bitne so celice v programskem pomnilniku ?
- 5. Koliko nivojev ima sklad (stack) ?
- 6. S katerega naslova v programskem pomnilniku se začne izvajati program ?
- 7. Kateri datotečni registri se pojavijo v vseh štirih segmentih ?

### 4.4. Osnovne vhodno/izhodne enote

Spoznali boste osnovne vhodno/izhodne enote – digitalne ali logične vhode/izhode. Naučili se boste konfigurirati posamezne priključke vrat A, B in C ter programsko krmiliti izhode in ugotavljati stanja vhodov.

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov programiranja. Dodatna znanja iz programiranja v zbirnem jeziku so v: [2][4][6][7]. Za razumevanje programov, ki so kodirani v C-jeziku, je potrebno poznavanje osnov ANSI C strukturnega programiranja: [15][11][13][5]. Veliko rešitev praktičnih nalog z uporabo PIC mikrokrmilnikov je na spletnem portalu <u>http://www.interg.or.jp/japan/se-inoue/e pic6.htm</u>.

```
<u>Vprašanja za utrjevanje:</u>
Napišite podprogram za inicializacijo vseh linij vrat PORTB kot izhodi.
Rešitev:
PBizh bsf STATUS,RP0 ; 1 -> RP0 , izbira Bankl za dostop do TRISB
clrf TRISB ; 0 -> TRISB, vsi izhodi
bcf STATUS,RP0 ; 0 -> RP0 , izbira Bank0 za dostop do PORTB
return ; konec podprograma
1. Napišite podprogram za inicializacijo vseh linij vrat PORTC kot vhodi.
2. Kakšen pomen imajo linije vrat PORTA ?
3. Napišite programček za ugotavljanje stanj na vhodih: RA0, RA1 in RA3 ?
4. Stanje stikala na vhodu RA4 prenesi (programsko) na izhod RC3.
```

5. Izhode na RC0 in RC1 preklapljaj izmenoma vsakih (pribl.) 100 ms.

### 4.5. Analogni vhodi

Spoznali boste vmesnik mikrokrmilnika, ki omogoča analogne vhode mikrokrmilnika. Naučili se boste konfigurirati posamezne priključke vrat PORTA za izbiro analognih vhodov. Naučili se boste uporabljati 10 ali 8-bitni rezultat in ločevati med desno oz. levo poravnavo ter interpretirati številčni rezultat AD pretvorbe.

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov programiranja in številskih sistemov. Dodatna znanja iz programiranja v zbirnem jeziku so v: [2][4][6][7]. Za razumevanje programov, ki so kodirani v C-jeziku, je potrebno poznavanje osnov ANSI C strukturnega programiranja: [15][11][13][5]. Veliko rešitev praktičnih nalog z uporabo PIC mikrokrmilnikov, je na spletnem portalu <u>http://www.interq.or.jp/japan/se-inoue/e pic6.htm</u>. Podrobnejši opis delovanja A/D pretvornika s primeri uporabe v PIC mikrokrmilnikih najdemo tudi v: [8][16][17][18].

<u>Vprašanja za utrjevanje:</u>

Napišite podprogram z imenom InAD1 za inicializacijo AD pretvornika (vhod AN1) z desno poravnavo rezultata.

Rešitev:

; InAD1	bsf	icializacija re STATUS,RPO	gistrov vhodno/izhodnih vmesnikov ; 1 -> RPO, Bankl za dostop do TRISA in ADCONI ; 1 -> TRISE 1 B21/ANI zbod
	DBL	IKIOA	/ I -> IRISA.I, RAI/ANI- VIIOU
; i	niciali	zacija A/D — AN	1
	movlw	b'10000100'	; priprava vredn. za ADCON1, vhod AN1-analogni, tudi AN3 in AN0
	movw£	ADCON1	;bit7=1 -> desna poravnava 10-bitnega rezultata
	bcf	STATUS, RPO	;0 ->RPO, nazaj v banko O
	movlw	b'10001001'	;takt (f/32,Fosc=20MHz), izbira anal. kan. AN1, AD omogocen
	movw£	ADCON0	;konec inicializacije
	return		;konec podprograma

- 1. Napišite podprogram z imenom InAD3 za inicializacijo AD pretvornika (vhod AN3) z levo poravnavo rezultata.
- Napišite podprogram (ki se navezuje na točko 1) za Start AD pretvorbe vhoda AN3 in shranjevanje 8-bitnega rezultata v register W.
- 3. Kolikšna napetost je na analognem vhodu, če smo pri 8-bitni pretvorbi dobili rezultat 7Fh ?
- 4. Kolikšna napetost je na analognem vhodu, če smo pri 10-bitni pretvorbi dobili rezultat 300h in je V<sub>ref+</sub> = 4,00 V ?
- 5. Kolikšen je čas AD pretvorbe (približno) in po kolikem času lahko sprožimo ponovno AD pretvorbo ?

## 4.6. Časovnik Timer0 in periodično generiranje prekinitev

Spoznali boste osnovni 8-bitni časovnik. Naučili se ga boste konfigurirati in generirati natančne časovne intervale s pomočjo prekinitev.

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov programiranja in številskih sistemov. Dodatna znanja iz programiranja v zbirnem jeziku so v: [2][4][6][7]. Za razumevanje programov, ki so kodirani v C-jeziku, je potrebno poznavanje osnov ANSI C strukturnega programiranja: [15][11][13][5]. Veliko rešitev praktičnih nalog z uporabo PIC mikrokrmilnikov je na spletnem portalu <u>http://www.interq.or.jp/japan/se-inoue/e\_pic6.htm</u>. Podrobnejši opis delovanja časovnika Timer0 s primeri uporabe v PIC mikrokrmilnikih je na voljo v: [8][16][17][18].

<u>Vprašanja za utrjevanje:</u>

Napišite podprogram z imenom InTMR0 za inicializacijo časovnika Timer0, ki bo periodično prožil prekinitve na 0,2 ms.

Rešitev:

InTMR0	bsf movlw movwf	STATUS, RPO b'00000001' OPTION_REG	; 1 -> RPO , izbira Bankl za dostop do OPTION_REG ;priprava vrednosti 0x01 za OPTION_REG ;preddelilnik casovnika TimerO nastavimo na 4:1
	bcf	STATUS, RPO	; 0 -> RPO, nazaj v BankO
	movlw	b'10100000'	;priprava vrednosti 0xA0 za INICON, GIE=1, TOIE=1
	movw£	INTCON	;Omogocimo prekinitve, ki jih sproza casovnik TimerO
	movlw	d'6'	;priprava vrednosti (desetisko) 6 za TMRO
	movw£	TMR0	;vrednost intervala: 0,2*(256-6)*4=200us=0,2ms
	return		;konec podprograma

- 1. Napišite podprogram z imenom InTMR0\_2 za inicializacijo časovnika Timer0, ki bo periodično prožil prekinitve na 2,0 ms.
- 2. Koliko bitni je števec TMR0 in kakšne vrednosti preddelilnika lahko izbiramo ?
- 3. Kakšna je funkcija časovnega stražnika (Watch Dog Timer)?
- 4. Ali lahko Timer0 uporabljamo, ne da bi prožil prekinitve ?
- 5. Kakšen interval bi dosegli v zgornjem rešenem zgledu, če ne bi vpisali nobene vrednosti v register TMR0 ?

## 4.7. Generiranje pulzno-širinskih signalov (PWM)

Spoznali boste vmesnik mikrokrmilnika, ki omogoča generiranje pulznoširinskih signalov (PWM). Naučili se boste konfigurirati registre za izbiro periode (frekvence) in širine pulza vmesnikov PWM1 in PWM2.

**Za razumevanje tematike je potrebno tudi predznanje** iz osnov programiranja in številskih sistemov. Dodatna znanja iz programiranja v zbirnem jeziku so v: [2][4][6][7]. Za razumevanje programov, ki so kodirani v C-jeziku, je potrebno poznavanje osnov ANSI C strukturnega programiranja: [15][11][13][5]. Veliko rešitev praktičnih nalog z uporabo PIC mikrokrmilnikov je na spletnem portalu <u>http://www.interq.or.jp/japan/se-inoue/e pic6.htm</u>. Podrobnejši opis delovanja časovnika Timer2 in PWM vmesnika s primeri uporabe v PIC mikrokrmilnikih je v: [16][17].

<u>Vprašanja za utrjevanje:</u>

Napišite podprogram z imenom InPWM1 za inicializacijo izhoda PWM1 s frekvenco 2,00 kHz.

#### Rešitev:

InPWMI   	osf novlw novwf ocf clrf osf osf novlw novwf	STATUS, RP0       ;         TRISC, 2       ;         d'155'       ;         PR2       ;         STATUS, RP0       ;         CCPR1L       ;         CCP1CON, 3       ;         CCP1CON, 2       ;         b'00000111'       ;         T2CON       ;	<pre>: 1 -&gt; RP0 , izbira Bankl za dostop do TRISC in PR2 : RC2/CCP1 port za PWM1 definiramo kot izhod : desetisko 155 -&gt; Za nastavitev frekvence signala 2kHz : Perioda=(155+1)0.2us*16 = 499,2us -&gt; f=1/499,2 = 2,003KHz : Nazaj v banko0 : zacetna vrednost sirine pulza: CCPR1L = 0 : 1 -&gt; CCP1CON.3 : 1 -&gt; CCP1CON.2, bit3 in 2 na 1 v CCP1CON-PWM nacin delovanja : vrednost za T2CON, bit1,0 -&gt; 1, preddelilnik je 1:4 : 1 -&gt; bit2, start casovnika TMR2 : koneg pedprograma</pre>
---	--	--	---

Napišite podprogram z imenom InPWM\_2 za inicializacijo časovnika izhoda PWM2 s frekvenco 5,00 kHz.

- 1. Koliko bitno vrednost lahko vpišemo kot vrednost za širino pulza ?
- 2. Kakšne vrednosti preddelilnika lahko izbiramo?
- 3. Kakšna je ločljivost nastavitve širine pulza (najkrajša in najdaljša širina pulza v ns) pri frekvenci 20 kHz ?
- 4. Ali PWM vmesnik proži prekinitve ?
- 5. Kakšna je najmanjša širina pulza (razen 0) v zgornjem (rešenem) primeru ?

# 5. Dodatek

## 5.1. Primer popolnega zgleda programa v C-jeziku za MPU-16F876

```
// Naziv programa: Regulacija temperature na MPU-PIC16F876
11
// Avtor(ji), verzija, datum: Darjan Leskovar, Sašo Jesenicnik, Janez Pogorelc, 3.6.2004
11
// Opis programa: Program dvotockovno regulira temperaturo grelnega telesa.
//----
#include <pic1687x.h>
#include <stdlib.h>
                                //stdlib.h potrebujemo, da lahko vkljucimo funkcijo abs()
#include <stdio.h>
        const unsigned char stevila[13]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F,0x63,0x39,0x00};
        const unsigned char linija[4]={0x80, 0x40, 0x20, 0x10};
        unsigned int histereza, stevec, test1, test2, t1, t2, w, x, y, z;
        unsigned int temperatura, zeljena_temp, meja, zg_meja, sp_meja;
        unsigned char i=0;
        unsigned char toleranca;
        int razlika;
        int podatki[7];
static void interrupt prekinitev()
                                        //podprogram, ki se izvede ob prekinitvi
{
        stevec++;
                                 //spremenljivka za dolocanje zakasnitev;
        w++;
                                 //spremenljivka za hitrost izpisovanja na display
        if(w>=100)
                                 //na vsako stoto prekinitev se vrednosti na zaslonu osvezijo
        {
                podatki[0]=podatki[4];
                podatki[1]=podatki[5];
                podatki[2]=podatki[6];
                w=0;
        }
        PORIC=(PORIC&0x0F)|linija[i]; //Izbira i-tega LED displaya
        PORTB=stevila[podatki[i]];
                                         //zapis stevila v izbran display
        if(i==1)
                PORTB=(PORTB|0x80);
                                        //pri drugi stevilki vkljucimo decimalno piko (RB7=1)
        i++;
                                         //izbira naslednjega displaya in stevilke
        if(i>3)
                i=0;
        TOIF=0;
}
//-
void zapis(x)
                        //podprogram za razstavitev neke vrednosti na cifre in njihov vpis v spremenljivke
za izpis
                        // na display.
{
        podatki[4]=x/100;
                             //desetice
        z=x%100;
                        //v spremenljivko si shranimo ostanek deljenja z sto
        podatki[5]=z/10; //enice
        podatki[6]=z%10; //prva decimalka
        if(podatki[4]==0)
                podatki[4]=12; //ce je desetica nic, je ne izpišemo, display ostane prazen
        return;
}
//-
unsigned int branje_senz()
                                 //podprogram za branje senzorja(dejanske temperature)
{
        ADCON0=0x81;
                                 //(Ob1000001)izberemo kanal ANO (senzor)
        ADRESH=0;
        ADREST =0;
        temperatura=0;
                                         //spremenljivko postavimo na nic, da bo vsota pravilna
        for(t1=0; t1<16; t1++)
                                         //sestnajstkrat izmerimo temperaturo in sestejemo vse vrednosti
```

```
{
                ADGO=1;
                while(ADGO==1);
                temperatura=(temperatura+(ADRESL|(ADRESH<<8))); //merimo v 10-bitni locljivosti
        }
        temperatura=temperatura/16;
                                         //izracunamo povprecje izmerjenih vrednosti
        return temperatura;
                                          //vrnemo vrednost v spremenljivki temperatura
}
void branje_potenc()
                                //podprogram za branje iz potenciometra
{
        ADCON0=0x99;
                                 //(Ob10011001) izbira kanala AN3 (potenciometer)
        ADRESH=0x00;
        ADRESL=0x00;
        ADGO=1;
                                 //start AD pretvorbe
        while(ADGO==1);
                                 //cakanje na izvedbo AD pretvorbe
        return;
}
11-
void nastavitev()
                        //podprogram za nastavljanje željene temperature
{
        RC0=1;
                        //vklop signalizacijske diode
        while(1)
        {
                do
                         //zanka za nastavitev temperature, izvedla se bo vsaj enkrat
                 {
                         branje_potenc();
                         zeljena_temp=(ADRESL (ADRESH<<8));
                         zeljena_temp=(zeljena_temp*42)/43;
                                                                //vrednosti 0-1023 pretvorimo v
vrednosti od 0-999
                         zapis(zeljena_temp);
                         meja=((histereza*zeljena_temp)/100);
                         sp_meja=(zeljena_temp-meja);
                         zg_meja=(zeljena_temp+meja);
                         branje_potenc();
                         test1=ADRESL (ADRESH<<8);
                                                          //prva vrednost AD pretvorbe
                         stevec=0;
                         while(stevec<31);
                                               //zakasnitev 0.1 sekunde
                         branje_potenc();
                         test2=ADRESL (ADRESH<<8);
                                                          //druga vrednost AD pretvorbe
                         razlika=abs(test1-test2);
                }
                while(razlika>toleranca);
                                                 //Zanko ponavljamo, dokler premikamo potenciometer
                branje_potenc();
                test1=ADRESL (ADRESH<<8);
                y=0;
                for (t1=0; t1<=200; t1++)//zakasnitev dveh sekund, med katero 200 krat preverimo
                                                  // spremembo položaja potenciometra
                 {
                         for(t2=0; t2<3100; t2++)
                                asm("nop");
                         branje potenc();
                         test2=ADRESL (ADRESH<<8);
                         razlika=0;
                         razlika=abs(test1-test2);
                         if(razlika>toleranca)
                                 break;
                         else
                                 y++;
                if(y>=200)
                        break;
                                         //prekinemo zanko while(1)
        }
        RC0=0;
                        //izklop signalizacijske diode
        return;
}
```

void mair {	n()		//zacetek glavnega	programa				
<pre>podatki[3]=11; toleranca=5; histereza=10; TRISA=0x19; TRISB=0x00;</pre>			//Konstanta -> crka 'C' //Za toleranco izberemo neko majhno vrednost, recimo 5. //zacetna vrednost histereze (%) //(ObO0011001) RAO(ANO) in RA3(AN3) ter RA4 definirani kot vhodi, RA5 izhod //PORTB in PORTC definiramo kot izhode					
	PORTA=0x00 PORTB=0x00	); );	//Resetir	amo vse i	zhode			
	PORIC=0x00 OPTION=0x0 INICON=0xA ADCON1=0x8	); )5; ,0; 94;	<pre>//(0b0000101 )Preddelilnik casovnika Timer0 nastavljen na 1:64 //(0b1010000)Prekinitve casovnika Timer0 omogocene //(0b10000100)AN3 in AN0 analogna vhoda, desna razporeditev bitov // rezultata A/D pretvorbe //Funkcija za nastavljanje željene temperature //neskoncna zanka</pre>					
	// nastavitev while(1)	r();						
	{	001 1.						
	r t t	RCI=1; temperatu temperatu	ra=branje_senz(); ra=(temperatura*42	//tempera )/43;	atura je vrednost, ki jo vrne funkcija branje_senz() //skaliranje vrednosti v obmocje 0-999			
	L	rr ( cenper	RA5=0;		//temperatura visja od zgornje meje -> izklopimo grelec			
	i	if(temper	atura < sp_meja) RA5=1;		//temperatura nizia od spodnje meje -> vklopimo greleg			
	2	zapis(tem	peratura);		//temperaturo vpisemo v spremenljivke za izpis na display			
	// if(((RA4		 &&(RC5))==1)	 ۵(RC5))==1) //tipka T3 -> prikaz in vecanje l				
	1	l	RC1=0; histereza++; if(histereza>10) histereza zapis(histereza); stevec=0; while(stevec<610);	a=0; //za dve //stevec //zakasni	//histereza je lahko najvec 10%. sekundi prikažemo trenutno vrednost histereze zakasnitev postavimo na nic, da lahko štejemo prekinitve. .tev 2 sekundi			
	}	}	nastavitev();		//histereza se je spremenila, zato izracunamo nove meje			
	i } {	lf(RC7==1 {	&RA4==1) RC1=0; nastavitev();	//tipka 1	1 -> prikaz nastavljene temperature			
	; ; {	if(RC4==1 {	&RA4==1) zapis(sp_meja); RC1=0; stevec=0;		//tipka T4 -> prikaz spodnje meje temperature			
	١	ı	while(stevec<610);	//2 sec z	rakasnitev,vrednost ostane na displayu izpisana 2 sekundi			
	( ב }	} Lf(RC6==1 {	&RA4==1)	//tipka 1	2 -> prikaz zgornje meje temperature			
			<pre>RCl=0; stevec=0; while(stevec&lt;610);</pre>					
	}	}						
}	}							

Program 5-1: Popoln zgled programa v C-jeziku – dvotočkovna histerezna regulacija temperature

## 6. Literatura

- [1] James M. Sibigtroth: Understanding small microcontrollers; Prentice Hall, 1993; ISBN 0-13-089129-0, 250 strani
- [2] Nebojša Matić: PIC mikrokontroleri; Mikroelektronika, Beograd 2002; ISBN 86-902189-4-7; 276 strani
- [3] Matjaž Colnarič: Osnove digitalne tehnike v računalništvu; UM-FERI, Maribor, 2002; ISBN 86-435-0435-8; 138 strani
- [4] Miran Rodič, Janez Pogorelc: Navodila za delo z modulom MPU-PIC16F876 (interno gradivo), [http://www.ro.feri.uni-mb.si/predmeti/mikro\_el/nav\_mpu\_pic.pdf]; UM-FERI, Maribor, 2002
- [5] Darko Dužanec: Programiranje PIC-ev v C-ju (serija člankov), Svet elektronike, maj 2004 do julij 2004, številke 109-111, ISSN 1318-4679
- [6] Jernej Škvarč: PIC od začetka (serija 9 člankov), Svet elektronike, marec 2003 do januar 2004, številke 96-105, ISSN 1318-4679
- [7] Microchip: spletna stran proizvajalca PIC mikrokrmilnikov, [http://www.microchip.com/]
- [8] Microchip: Priročnik za mikrokrmilnike PIC16F87x (angl.), [http://www.ro.feri.uni-mb.si/predmeti/mikro\_el/FTP/PIC16F876\_30292b.pdf], 2000, 200 strani
- [9] Microchip: Priročnik za MPLAB ICD (angl.), [http://www.ro.feri.uni-mb.si/predmeti/mikro\_el/FTP/ICD\_51184d.pdf], 2000, 104 strani
- [10] Microchip: MPLAB V5.70 integrirano programsko okolje (program), [http://www.ro.feri.uni-mb.si/predmeti/mikro\_el/FTP/mp57full.zip], 2003, 13 MB
- [11] Brian W. Kernicham, Dennis M. Ritchie: Programski jezik C (slovenski prevod), 1990, UM-FERI Ljubljana, 240 strani
- [12] HI-TECH Software: spletna stran proizvajalca C-prevajalnika, [http://www.htsoft.com/]
- [13] HI-TECH Software: priročnik C-prevajalnika HI-TECH PICC Lite (angl.), [http://www.htsoft.com/files/demo/piccliteman.zip], 2002, 358 strani, 1,4 MB
- [14] HI-TECH Software: C-prevajalnik HI-TECH PICC Lite (program), [http://www.htsoft.com/files/demo/picclite-setup.exe], 2,4 MB
- [15] Miran Rodič: Uvod v programski jezik C, (interno gradivo) [<u>http://www.ro.feri.uni-mb.si/predmeti/sis\_meh/vaje/UvodC.pdf</u>], UM-FERI, 2001, 14 strani
- [16] Hobby Electronics: spletna navodila za uporabo PIC mikrokrmilnikov (angl.), [http://www.interq.or.jp/japan/se-inoue/e\_pic.htm]
- [17] Hobby Electronics: spletni primeri uporabe PIC mikrokrmilnikov (angl.), [http://www.interq.or.jp/japan/se-inoue/e\_pic6.htm]
- [18] MicrochipC.com: spletni primeri C-programov za PIC-e (angl.), [http://www.microchipc.com/]
- [19] MicrochipC.com: Bootloader najpreprostejši programator PICev (angl.), [http://www.microchipc.com/PIC16bootload/]
- [20] A. Tetičkovič, B. Brečko, B. Gračner: Mobilni robot za sledenje črti, Uvodni seminar skupine študentov 3.I. Mehatronika, [http://www.ro.feri.unimb.si/predmeti/skup\_sem/projektu/MobRob\_Tet\_Bre\_Gra.pdf], UM-FERI, 2003
- [21] J. Horvat, I. Kodrič: Poročilo o izdelavi mobilnega robota, Uvodni seminar skupine študentov 3.I. Mehatronika, [<u>http://www.ro.feri.uni-mb.si/predmeti/skup\_sem/projektu/protokol Kodric\_meha-</u> <u>rudi.pdf</u>], UM-FERI, 2003, 14 strani
- [22] Tekmovanje RoboT200X: spletni članki z opisi mini mobilnih robotov na osnovi PIC mikrokrmilnikov, [http://www.ro.feri.uni-mb.si/tekma/dodatne\_informacije\_nasveti.htm]