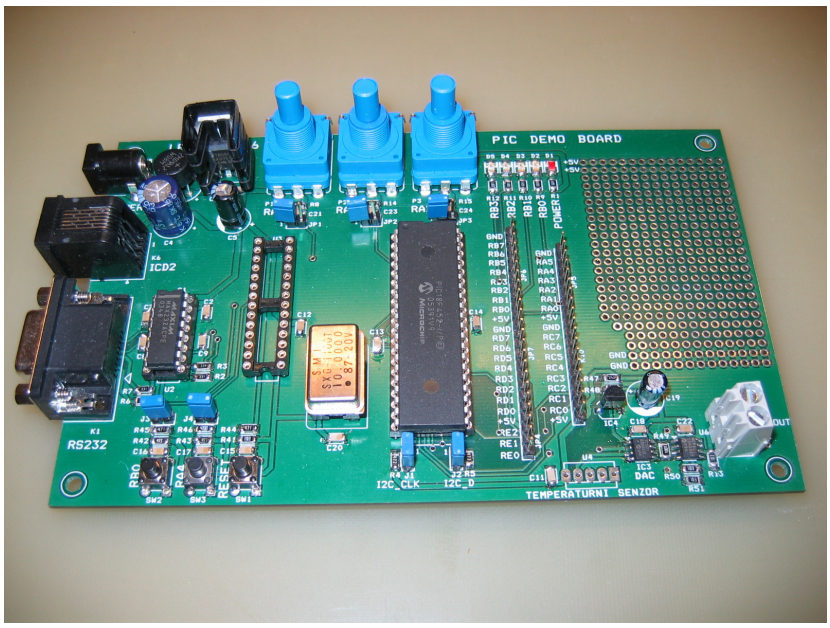
PIC18FXX2 8-bitni MIKROKRMILNIK

Razvojna orodja PIC DEMO, MPLAB in ICD2

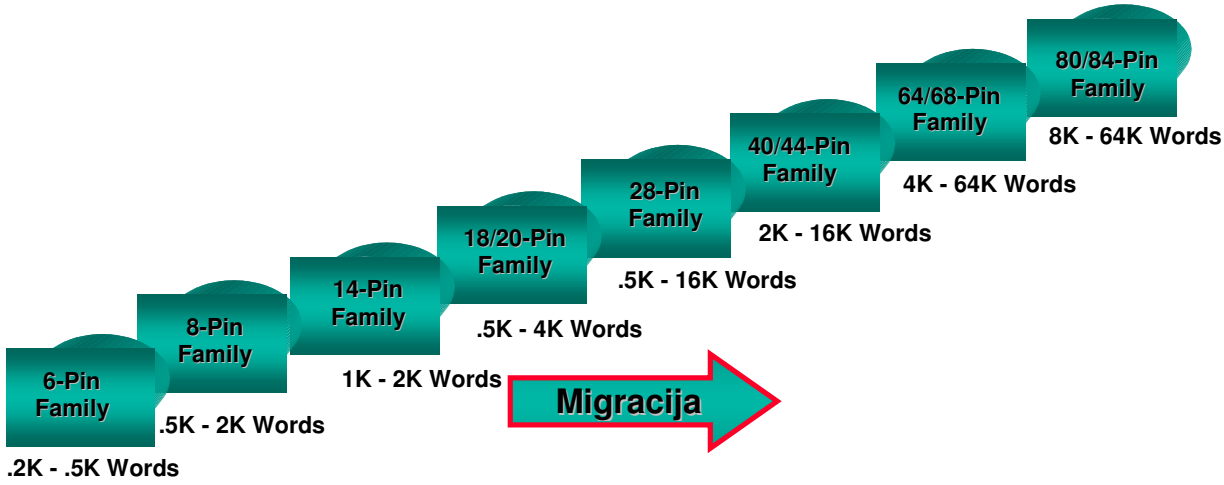


Ljubljana, 2006

PIC 18FXX2 8 bitni mikrokrmilnik – NAVODILA ZA UPORABO

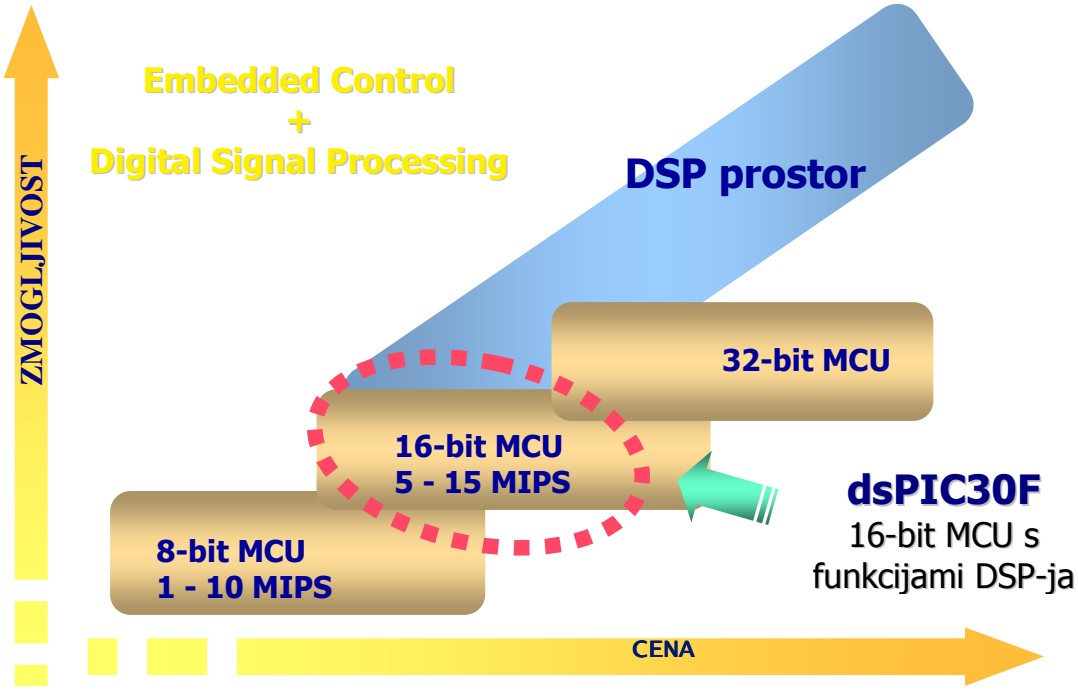
1 UVOD

Microchip ima široko paleto 8-bitnih mikrokrmilnikov in je njihov vodilni proizvajalec, saj ponuja več kot 194 različnih mikrokrmilnikov od najmanjšega na svetu do zmogljivega 80 pinskega. Njihova dobra lastnost je veliko število vgrajenih funkcij (A/D pretvorniki, Capture/Compare/PWM-moduli, SPI, I2C, UART, …) in cenena razvojna orodja (ICD2, MPLAB IDE).



Slika1: 8-bitni mikrokrmilniki

Za zahtevnejše aplikacije nam 8-bitni mikrokrmilniki običajno ne zadoščajo, zato se uporabniki v takih primerih odločijo za uporabo zelo zmogljivih DSP mikrokrmilnikov, ki pa v veliko primerih niso optimalno izkoriščeni. Microchip je v ta namen izdelal posebno skupino mikrokrmilnikov, ki zasedajo območje med 8-bitnimi mikrokrmilniki in DSP-ji (slika 2) imenovane dsPIC ali DSC mikrokrmilnike (*angl. Digital Signal Controllers*). dsPIC so v bistvu 16-bitni mikrokrmilniki z vgrajenimi perifernimi funkcijami in so navzven močno podobni običajnim 8-bitnim mikrokrmilnikom z razliko, da imajo dodano močno DSP jedro.



2. RAZVOJNA ORODJA

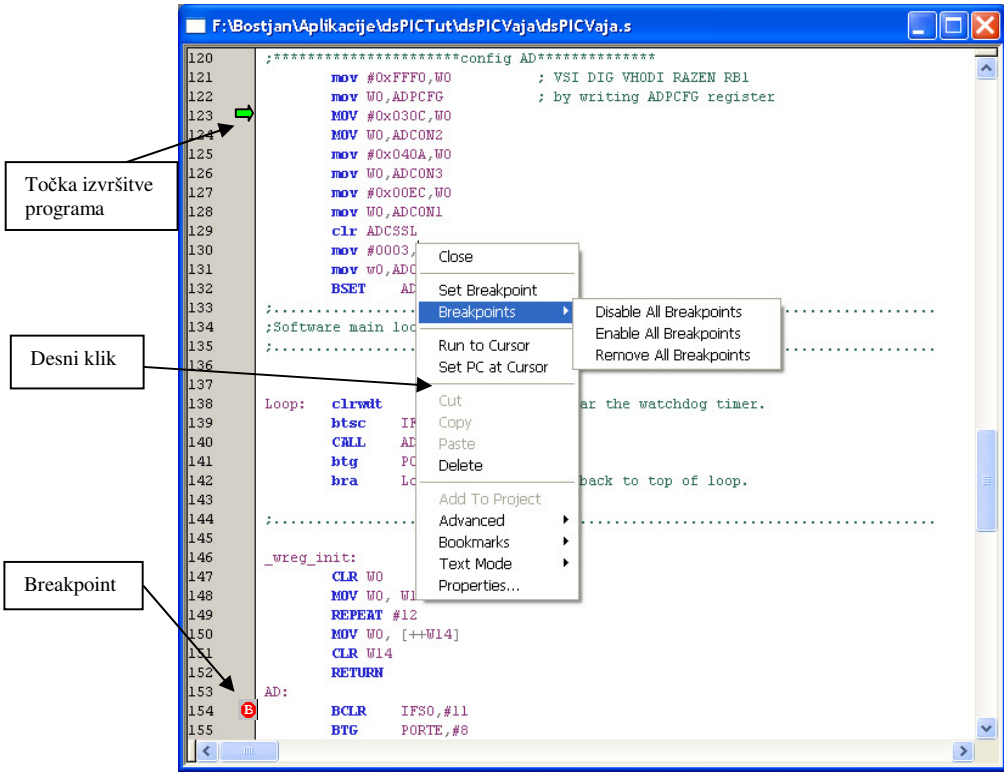
Za izdelavo projekta oz. aplikacije z PIC18Fxx2 mikrokrmilnikom rabimo razvojna orodja s pomočjo katerih enostavno napišemo, uredimo, testiramo naš program in programiramo izbrani mikrokrmilnik. Na voljo je več pripomočkov, ki omogočajo realizacijo zgoraj navedenih zahtev od katerih sta tudi MPLAB IDE in MPLAB ICD2 najpogosteje uporabljena.

2.1 MPLAB IDE

MPLAB IDE (*angl.: Integrated Developement Enviroment*) je obsežen urejevalnik, simulator, upravljalnik projektov in konstruktorski vmesnik za aplikacijo z uporabo MICROCHIP-ovega mikrokrmilnika.

2.1.1 Urejevalnik

Urejevalnik besedila je uporabniku prijazen sestavni element MPLAB IDE-ja, ki omogoča enostavno vnašanje programske kode. Njegove osnovne značilnosti kaže slika 2.



Slika 2: Urejevalnik besedila v MPLAB IDE

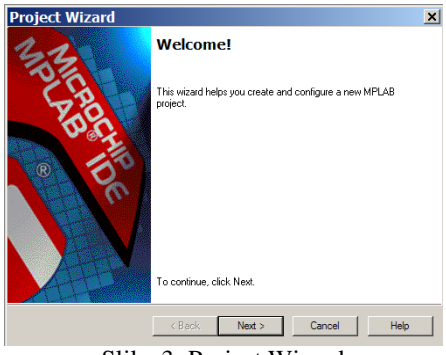
Zelena puščica kaže na katerem mestu se je izvajanje programa ustavilo, rdeča pika s črko B znotraj pa na katerem mestu naj se program ustavi (*angl.: Breakpoint*); program se izvaja do ukaza vključno z označenim.

2.1.2 Simulator

Simulator nam omogoča preverjanje delovanja programa in s tem odpravljanje morebitnih napak pred testiranjem na izbranem mikrokrmilniku. Delovanje simulatorja, ki je vključen v MPLAB IDE, je časovno omejeno z instrukcijskim ciklom. Kar pomeni, da je resolucija simuliranih dogodkov enaka dolžini instrukcijskega cikla. Simulator omogoča spreminjanje podatkovnega in programskega spomina, simuliranje vhodov/izhodov, funkcija »stopwatch« omogoča merjenje časa izvajanja posameznih delov programa. Kljub vsemu pa določene funkcije ne morejo biti popolnoma simulirane; tako lahko simuliramo A/D pretvorbo, med tem, ko same analogne vrednosti na vhodu ne moremo.

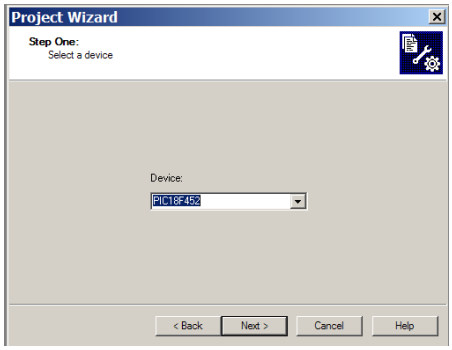
2.1.3 Izdelava projekta

Najlažji način za izdelavo projekta je s pomočjo MPLAB Project Wizard, ki se zažene s klikom na *Project>ProjectWizard.* Odpre se nam »Welcome« okno (slika 3). Kliknemo **Next** za nadaljevanje.



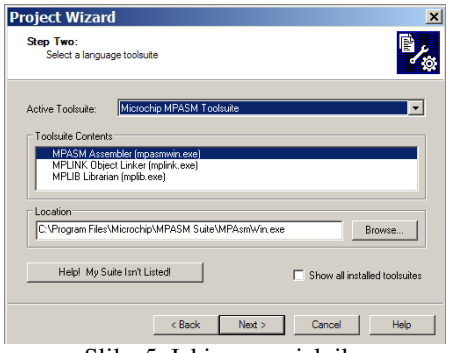
Slika 3: Project Wizard

V naslednjem koraku moramo izbrati mikrokrmilnik, ki ga bomo uporabljali v naši aplikaciji (slika 4).



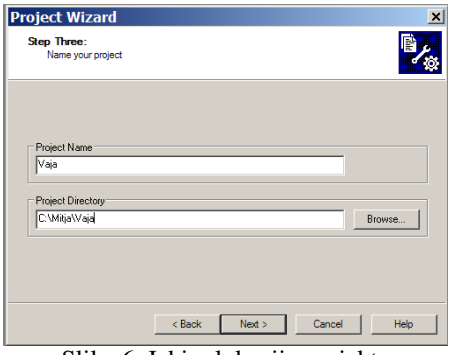
Slika 4: Izbira mikrokrmilnika

Nato je treba določiti orodje s katerim bomo program prevajali v strojno kodo, oz. izbrati moramo ustrezen prevajalnik. Če bomo pisali program v asemblerju potrdimo lokacijo *Microchip MPASM Toolsuite.* Slika 5 kaže izbrano možnost.



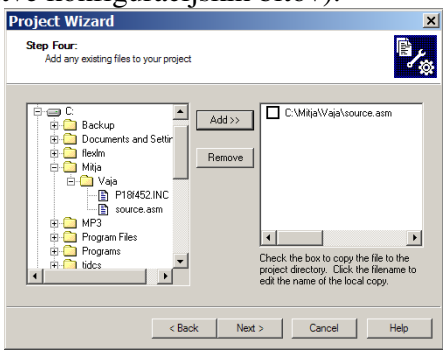
Slika 5: Izbira prevajalnika

Zaradi preglednosti je priporočljivo, da je vsak projekt v svoji mapi. V naslednjem koraku določimo ime projekta in mapo v kateri se nahaja (slika 6).



Slika 6: Izbira lokacije projekta

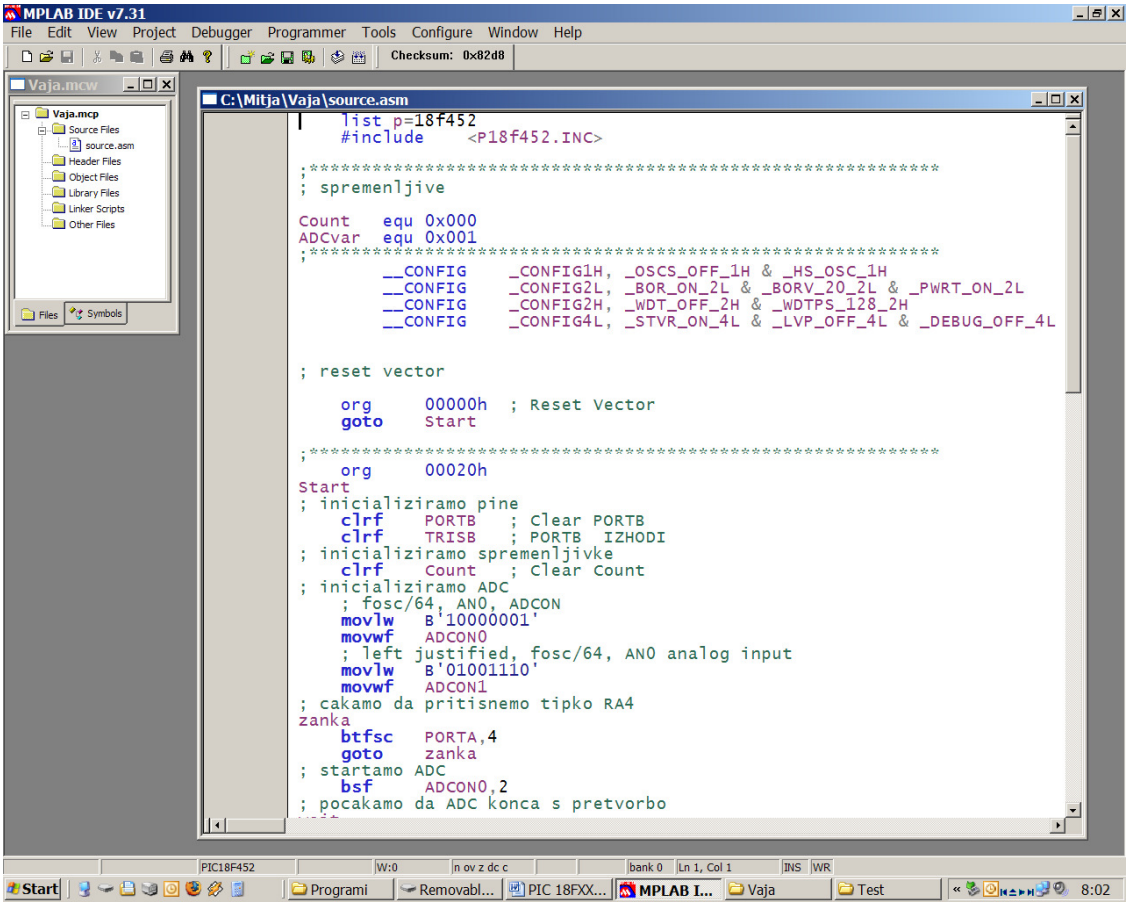
V naslednjem koraku lahko dodamo že obstoječe datoteke v naš projekt. V primeru na sliki 10 dodamo »source« datoteko, v katero bomo pisali naš program. Datoteka že vsebuje del programa, ki poskrbi za osnovne nastavitve (»include« datoteka, izbira oscilatorja in ostale nastavitve konfiguracijskih bitov).



Slika 7: Dodajanje datotek v projekt

Da zaključimo izdelavo projekta kliknemo **Next** in odpre se nam še okno v katerem je zapisan povzetek pravkar kreiranega projekta. Ko kliknemo **Finish** se nam v projektnem oknu (*angl.: Project Window*) pokaže ime datoteke, ki smo jo dodali projektu (*Source Files*).

Z dvojnim klikom na source.asm (»Source files«) se nam odpre datoteka, ki vsebuje programsko kodo potrebno za delovanje programa in prostor za pisanje našega programa (slika 8).

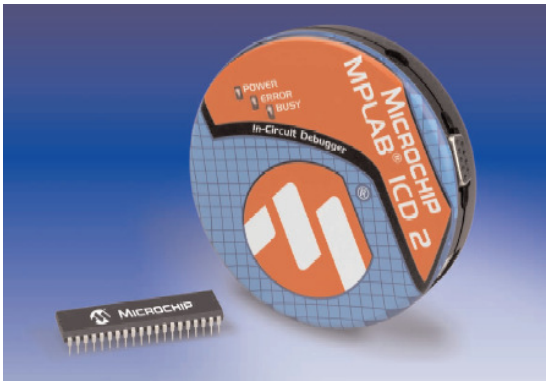


Slika 8: »Source« datoteka

2.1.4 MPLAB ICD2

V prejšnjem podpoglavju smo izvedli vse postopke za izdelavo projekta. Izbrali smo ustrezen mikrokrmilnik in prevajalnik, mu določili ime in mapo v kateri bodo shranjene vse potrebne datoteke za njegovo delovanje, vstavili šablonsko datoteko v katero zapišemo naš program. Vse to nam omogoča izdelavo programske kode, prevajanje programa in testiranje s simulatorjem. Naslednji korak je zapis programa v izbrani mikrokrmilnik. Možnih je več načinov programiranja:
- MPLAB ICD2,
- MPLAB PM3,
- PRO MATE II,
- preko serijske (RS 232) povezave z uporabo »Bootload« programa.

Ogledali si bomo le uporabo MPLAB ICD2 (*angl.: In-Circuit Debugger*), ki je najbolj pogost in cenen »debugger/programer«. Če ga želimo uporabljati kot »debugger«, moramo izvesti dva koraka. Prvi zahteva, da je aplikacija sprogramirana v mikrokrmilnik, drugi pa uporablja del spomina mikrokrmilnika za izvajanje in testiranje. Oba koraka sta neposredno povezana z MPLAB IDE funkcijami: programiranje kode v mikrokrmilnik, vstavljanje prekinitvenih točk (»breakpoints«), izvajanje programa po korakih,…



Slika 9: MPLAB ICD2

5. Primer programa

Na naslednjem programu si bomo ogledali tipičen potek programiranja PIC mikrokrmilnika. Napisali bomo program, ki ob vsakem pritisku tipke na nožici RA4 sproži AD pretvorbo, počaka na rezultat, poveča števec pritiskov ter prižge oziroma ugasne LED diodo na nožici RB2.
Ves program se nahaja samo v eni datoteki. Potek programa pa je sledeč:

Na začetku datoteke je vključena datoteka z deklaracijami registrov PIC mikrokrmilnika.

Zatem sledi deklaracija spremenljivk, ki jih bo program uporabljal.

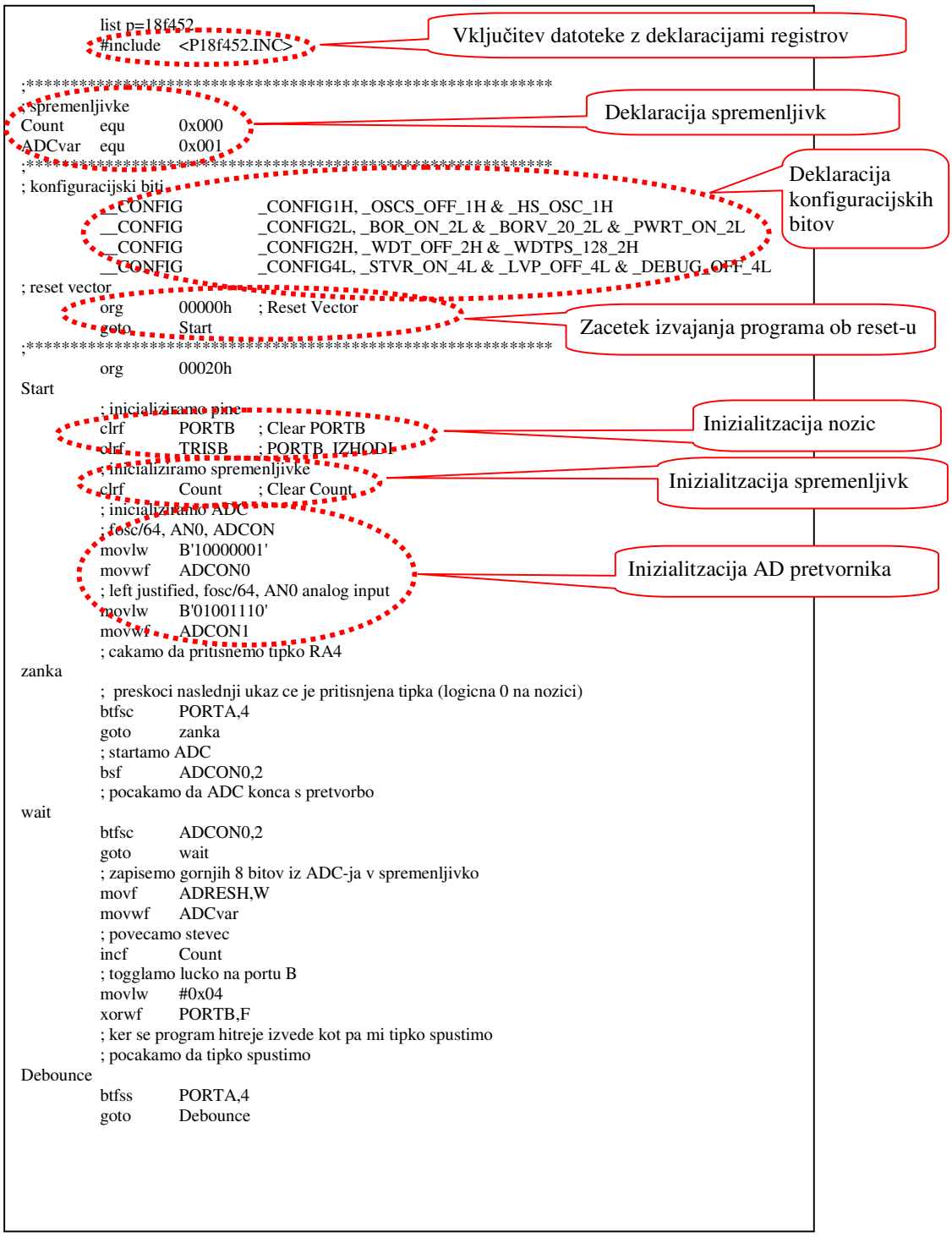
Nato se nahaja deklaracija konfiguracijskih bitov. Ti biti določajo delovanje PIC mikrokrmilnika ob zagonu (hitrost, napetost, …).

Sam program se začne šele za vrstico »*;reset vektor*«. Tu povemo kje naj PIC mikrokrmilnik začne z izvajanjem programa ob reset-u.

Program nato nadaljuje z inicializacijo vhodno izhodnih nožic ter inicializacijo AD pretvornika.

Šele pri labeli *zanka* se začne izvajati glavni del programa:

Najprej počakamo na pritisk tipke RA4.
Nato poženemo AD pretvornik in počakamo na rezultat.
Rezultat shranimo v spremenljivko ADCvar.
Povečamo števec pritiskov na tipko za ena.
Ter spremenimo stanje na nožici RB2.


list p=18f452
#include <P18f452.INC>
Vključitev datoteke z deklaracijami registrov
;***
; spremenljivke
Count equ 0x000
ADCvar equ 0x001
Deklaracija spremenljivk
;***
; konfiguracijski biti
__CONFIG _CONFIG1H, _OSCS_OFF_1H & _HS_OSC_1H
__CONFIG _CONFIG2L, _BOR_ON_2L & _BORV_20_2L & _PWRT_ON_2L
__CONFIG _CONFIG2H, _WDT_OFF_2H & _WDTPS_128_2H
__CONFIG _CONFIG4L, _STVR_ON_4L & _LVP_OFF_4L & _DEBUG_OFF_4L
Deklaracija konfiguracijskih bitov
; reset vector
org 00000h ; Reset Vector
goto Start
Zacetek izvajanja programa ob reset-u
;***
org 00020h
Start
; inicializiramo pine
clrf PORTB ; Clear PORTB
clrf TRISB ; PORTB IZHODI
Inizialitzacija nozic
; inicializiramo spremenljivke
clrf Count ; Clear Count
Inizialitzacija spremenljivk
; inicializiramo ADC
; fosc/64, AN0, ADCON
movlw B'10000001'
movwf ADCON0
; left justified, fosc/64, AN0 analog input
movlw B'01001110'
movwf ADCON1
Inizialitzacija AD pretvornika
; cakamo da pritisnemo tipko RA4
zanka
; preskoci naslednji ukaz ce je pritisnjena tipka (logicna 0 na nozici)
btfsc PORTA,4
goto zanka
; startamo ADC
bsf ADCON0,2
; pocakamo da ADC konca s pretvorbo
wait
btfsc ADCON0,2
goto wait
; zapisemo gornjih 8 bitov iz ADC-ja v spremenljivko
movf ADRESH,W
movwf ADCvar
; povecamo stevec
incf Count
; togglamo lucko na portu B
movlw #0x04
xorwf PORTB,F
; ker se program hitreje izvede kot pa mi tipko spustimo
; pocakamo da tipko spustimo
Debounce
btfss PORTA,4
goto Debounce