Mpp Cvičenie 8

Vypracoval Adam Petro, Martin Pullmann

#include <msp430.h>

#include "evb1.h"

long int volty;

unsigned int pomoc;

void main(void)

{

unsigned char display[4] = {0,0,0,0};

evb1\_io\_init();

ADC10CTL0 = ADC10SHT\_2 + ADC10ON + ADC10IE;

// interval ustalenia vstup. napatia,

/\* ADC10SHT\_2 znaci 16 x period hodin ADC10CLK \*/

// zapnutie prevodnika - bit ADC10ON,

// povolenie prerusenia - bit ADC10IE

//VR+ = VCC, VR- = VSS, MSC = 0 po skonceni prevodu sa nespusti hned novy prevod

//REFON=0 vypni interny zdroj referencneho napatia

//ADC10IFG = 0 nulovanie priznaku, ENC = 0 prevod zakazany

//SHSx = 0x00 - spustenie prevodu od bitu ADC10SC

//ADC10DF =0 - hodnota napatia vyjadrena vo formate: priamy kod (straight binary code)

//ADC10DIVx = 0x00 - delicka hodin signalu 1:1

//ADC10SSELx = 0x00 - volba hodinoveho signalu pre prevodnik - oscilator ADC10OSC

//3.57V=1022 MAX namerana

//3570/1022=3.493 => koeficient

ADC10CTL1 = INCH\_0; // vstup prevodnika pripojime na kanal A0,

// ktory je na pine, kde bol povodne P1.0

// t.j. nastavime alternativnu funkciu pinu

ADC10AE0 |= 0x01; // odpojenie digitalnych casti pinu portu P1.0

while(1)

{

ADC10CTL0 |= ENC + ADC10SC;

// povolenie prevodu - bit ENC a

// start prevodu - bit ADC10SC

\_\_bis\_SR\_register(CPUOFF + GIE);

// vstup do LPM0, global povolenie preruseni

// ale v ISR sa zmeni bit CPUOFF na 0 a preto

// program po navrate z ISR nezaspi, ale pokracuje

// v hlavnom programe. Tato technika programovania

//nesuvisi s AD prevodnikom, je vseobecne pouzitelna

volty=ADC10MEM;

volty=(volty\*57229)>>14;

//kedze MSP430 nepodporuje float cisla, vynasobil som

//najvyssiu meranu hodnotu 10000 aby som sa zbavil desatinnej

//casti (nie 10000 ale najblizsiou mocninou 2,

//teda 2^14=16384) a potom urobil bitovy posun o 14 bitov

int\_to\_bcd(display, (int) volty);

// prevod jedneho cisla na styri dekadicke cislicovky

send\_data\_to\_disp(display, 0); // desatinna bodka => [V], 4

delay(15000); // rychlost opakovania prevodu

}

}

// ADC10 interrupt service routine

#pragma vector=ADC10\_VECTOR

\_\_interrupt void ADC10\_ISR(void)

{

\_\_bic\_SR\_register\_on\_exit(CPUOFF); // Clear CPUOFF bit from 0(SR)

// Nastav bit CPUOFF nulu v odlozenej kopii

// stavoveho registra SR.

// kopia SR je odlozena v zásobníku

//po RETI sa tato (modifikovana) kopia zapise do SR.

}

# Záver:

* Na cvičení sme sa naučili nastavovať DAC prevodníky pomocou matematických operácii a bitových posunov
* Pomocou prevodu DAC sme vyrobili ukážku voltmetra
* Na zobrazenie hodnoty sme použili segmentový LED-Displej a funkciu „send\_data\_to\_disp“, ktorá je zahrnutá v súbore evb1.c