Documentație

Programarea Aplicațiilor de Timp Real – Proiect





|  |  |
| --- | --- |
| Îndrumător | Studenți |
| Ș.l.Dr.Ing. Cătălin Brăescu | Boghean Ștefan (1404B) și Paciurcă Andrei-Alexandru (1404A) |

# Cuprins

# Tema proiectului

Să se proiecteze și să se realizeze o aplicație încorporată bazată pe sistemul de operare

FreeRTOS și microcontroller-ul dsPIC33FJ128MC802. Aplicația va comanda o trapă prin

intermediul unui servomotor, pe baza informației de temperatură achiziționate prin intermediul

unui senzor de temperatură digital DS18S20/DS18B20. Aplicația va respecta următoarele detalii de implementare:

1. va fi pornită respectiv oprită prin intermediul butonului SW1 de pe plăcuță. Starea

aplicatiei va fi indicată de un LED conectat la pinul RB0. LED-ul va fi aprins atunci când

aplicația este pornită și va fi aprins intermitent când aplicația este oprită.

1. temperatura va fi achiziționată de senzorul DS18S20 sau DS18B20, deschiderea trapei

fiind controlată prin servomotor. Temperaturii de 25°C îi va corespunde poziția centrală a

servomotorului. Servomotorul se va deplasa între pozițiile extreme pentru limitele de

temperatură (20°C-30°C). Senzorul de temperatură va fi conectat la pinul RB2 al

microcontroller-ului.

1. aplicația va pune la dispoziție pe lângă modul automat de comandă a servomotorului și

un mod manual de control prin intermediul unei tensiuni externe aplicate pe pinul RB3.

Trecerea dintr-un mod de lucru în altul se face prin intermediul meniului de comenzi

implementat pe PC. În modul de lucru manual servomotorul va fi poziționat pe mijloc

pentru o tensiune de 2V și se va deplasa în pozițiile extreme pentru valorile limită de 1V

și 3V. Modul de lucru va fi indicat vizual prin intermediul unui LED conectat la pinul

RB1. LED-ul va fi aprins în modul de lucru automat și stins în modul de lucru manual.

1. aplicația va permite comunicația prin interfața serială cu PC-ul. Placa cu dsPIC va

recepționa comenzi sub forma unor caractere și va transmite drept răspuns șiruri de

caractere.

1. pe un ecran LCD vor fi afișate informații privind temperatura, modul de lucru, tensiunea

achiziționată de la pinul RB3 precum și ultima comandă primită.

1. aplicația va pune la dispoziția utilizatorului un meniu cu următoarele comenzi:

* interogare mod de lucru;
* comutare mod de lucru automat/manual (modul de lucru predefinit este cel

automat);

* interogare temperatură.

# Dezvoltarea Proiectului

* Proiectul este realizat în mediul de dezvoltare MPLAB IDE V8.43.
* Aplicația este implementată în jurul microcontroller-ului dsPIC33FJ128MC802/dsPIC33FJ64MC802, iar pentru programarea acestuia se folosește programatorul PICkit 3.
* Pentru realizarea aplicației este folosit sistemul de opereare FreeRTOS.
* Pentru utilizarea perifericelor, folosim un set de drivere. Pentru comunicarea cu senzorul digital de temperatură se folosesc funcțiile necesare care se regăsesc în fișierele ds18s20.c si ds18s20.h .
* Pentru LCD sunt puse la dispoziție fișierele new\_lcd.c si new\_lcd.h. Pentru comunicarea cu interfața serială se folosesc funcțiile necesare existente în fișierele new\_serial.c și new\_serial.h

# Modul de implementare

Aplicatia noastra nu a fost finalizata iar undele cerinte lipsesc din implementare cum ar fi: comunicarea din exterior prin UART, comanda pwm nu este testata in deplinatate iar scalarea tensiunii adc nu este corecta, insa cu toate acestea citirea datelor de la sensor, afisarea pe display a informatiilor, oprirea si pornirea aplicatiei si schimbarea modului de lucru functioneaza.

In implementarea noatra am folosit 4 taskuri :

|  |  |  |  |
| --- | --- | --- | --- |
| Nume task | Descriere task | Delay | Prioritate |
| StartStop | Acest task are rolul de a porni/opri aplicatia si comutarii unui LED conform apasarii unui buton. | 300 ms | 6 |
| TempRead | Se preia datele de la senzorul de temperatura. | 400 ms | 5 |
| DisplayInfo | Acest task afiseaza pe display informatiile aplicatiei. | 400 | 1 |
| Mode | Comutare mod de lucru si comanda PWM. | 400 | 4 |

1. Taskul StartStop

 Acest task are doua clauze if: prima verifica daca aplicatia este pornita(fie s-a apasat butonul si variabila ucApplicationRunning a trecut din 0 in 1, fie aceasta este deja 1) iar atunci se va face resume la celelalte taskuri ale aplicatiei si se reseteaza o variabila( ucTaskDeleted ) care indica daca taskurile au fost suspendate. In celalalt if se intra doar daca aplicatia este oprita (se apasa butonul si se reseteaza ucApplicationRunning sau daca aplicatia este deja oprita) si se trec celelalte taskuri in suspended.

Desigur la pornirea si oprirea aplicatiei nu se ia in considerare si acest task deoarece acesta se ocupa de comutarea LED-ului coresounzator starii aplicatiei.

1. Taskul Mode



Acest task are doua clauze if care verifica modul de operare curent. Daca modul de operare curent este AUTOMAT ( opMode are valoarea 1) atunci registrul corespunzator factorului de umplere al modulului PWM este calculat in functie de temperatura( Nu am testat vaorile optinute prin acea ecuatie, nu stim daca este in regula sau nu).

Daca modul curent de operare este cel MANUL atunci factorul de umplere se calculeaza conform tensiunii de pe intrarea analogica ( in main aceasta valoare este abstractizata prin variabila tensValue care este declarata si actualizata in fisierul interupts.c )

tensValue = (*float*)((*float*)ADC1BUF0\*3.3)/(*float*)4096;

Datorita faptului ca nu avem comunicatii UART, comutarea intre moduri se face tot prin apasarea butonului, asadar aceste doua functionalitati( start/stop si mod aplicatie) trebuie testate izolate una de cealalta prin comentarea unei linii in rutina de intrerupere a butonului:

*void* \_\_attribute\_\_ ((interrupt, no\_auto\_psv)) \_INT0Interrupt(*void*)

{

    ucApplicationRunning = !ucApplicationRunning;

    //opMode = !opMode;

    \_INT0IF = 0;// Resetam flagul corespunzator intreruperii

}