Ministerul Educației, Culturii și Cercetării al Republicii Moldova Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică Ingineria Software și Automatică

Raport

## Disciplina: Internetul Lucrurilor Tema: Sisteme de operare

A efectuat: st.gr.TI-194, Zavorot Daniel

A verificat: asist.univ. Litra Dinu

Chișinău 2022

# Definirea problemei :

Realizarea unei aplicații pentru MCU care va rula minim 3 task-uri in doua versiuni - Secvential si cu FreeRTOS

Aplicația va rula minim 3 task-uri printre care

1. Button Led - Schimbare stare LED la detecția unei apăsări pe buton.
2. un al doilea Led Intermitent în faza în care LED-ul de la primul Task e stins
3. Incrementare/decrementare valoare a unei variabile la apăsarea a doua butoane care va reprezenta numărul de recurențe/timp în care ledul de la al doilea task se va afla într-o stare
4. Task-ul de Idle se va utiliza pentru afișarea stărilor din program, cum ar fi, afișare stare LED, și afișare mesaj la detecția apăsării butoanelor, o implementare fiind ca la apăsarea butonului sa se seteze o variabila, iar la afișare mesaj - resetare, implementând mecanismul provider/consumer.

# Recomandări:

* 1. sa se implementeze comunicarea intre Taskuri ca provider consumer, adica:
* task-ul care generează date, provider, stochează rezultatele într-o variabila globala/semnal
* task-ul care utilizează aceste date, consumer, citește aceasta variabila/semnal.

*de ex: task de UI (LCD sau Serial) preia informația din niște variabile-semnale globale și raportează*

* 1. A se urma principiile prezentate la curs Sisteme Secvențiale
* stabilirea rezonabila a recurentei pentru a diminua incarcarea procesorului
* stabilirea ofsetului, intru a activa în ordinea cuvenita task urile

3: Task-ul de raportare pentru Secvențial cu utilizare STDIO printf() catre LCD va fi rulat in bucla infinita/IDLE deoarece este bazat pe un spin lock și ar putea bloca întreruperile deci secvențial clasic - printf & delay in main loop, in FreeRtos - un task separat.

4. Pentru Versiune cu FreeRtos, a se utiliza recomandarile sistemul.

# Obiective:

1. Proiectare aplicații în bază de MCU;
2. Aplicația va rula minim 3 task-uri;
3. Rulare în versiunea - Secvential
4. Rulare în versiunea cu FreeRTOS
5. Efectuarea schemelor logice a aplicațiilor;

# Introducere:

Un sistem de operare este un program de computer care acceptă funcțiile de bază ale unui computer și oferă servicii altor programe (sau *aplicații* ) care rulează pe computer. Aplicațiile oferă funcționalitatea pe care utilizatorul computerului o dorește sau are nevoie. Serviciile furnizate de sistemul de operare fac scrierea aplicațiilor mai rapidă, mai simplă și mai ușor de întreținut. FreeRTOS este o clasă de RTOS care este proiectată să fie suficient de mică pentru a rula pe un microcontroler - deși utilizarea sa nu se limitează la aplicațiile de microcontroler. FreeRTOS oferă funcționalitatea principală de planificare în timp real, comunicarea între sarcini, sincronizarea și primitivele numai. Aceasta înseamnă că este descris mai exact ca un nucleu în timp real sau un executiv în timp real. Funcționalități suplimentare, cum ar fi o interfață a consolei de comandă sau stive de rețea, pot fi apoi incluse cu componentele de completare.

**Problema nr.1:** versiunea - Secvențial

## Materiale și metode:

* 1. Trei LED;
  2. Un Microcontroler;
  3. 3 Rezistență;
  4. Trei butoane;

## Rezultate:

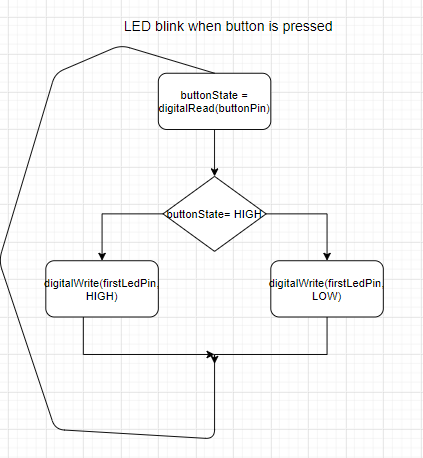


Figura 1.1 Schema bloc Task1 versiunea Secvențial (Task – LED blink - buton)

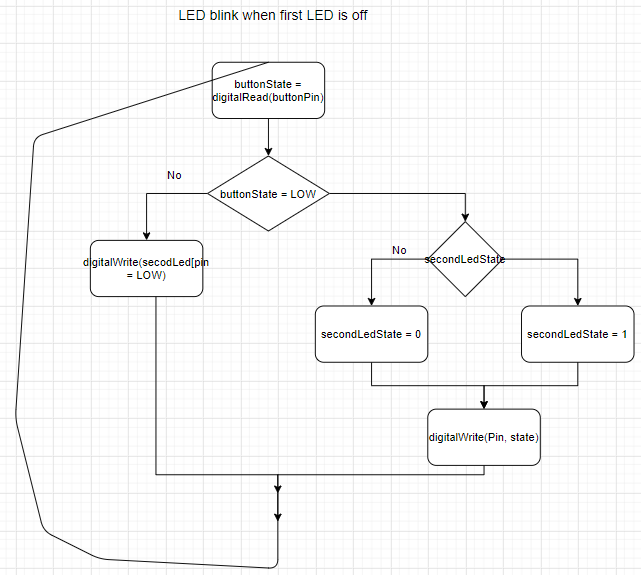


Figura 1.2 Schema bloc Task2 versiunea Secvențial (Task – LED blink – LED2)

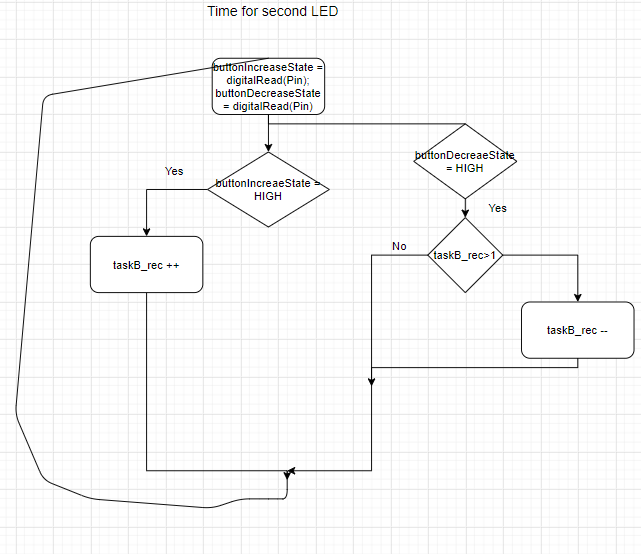


Figura 1.3 Schema bloc Task3 versiunea Secvențial (Task – LED – 2 butoane)

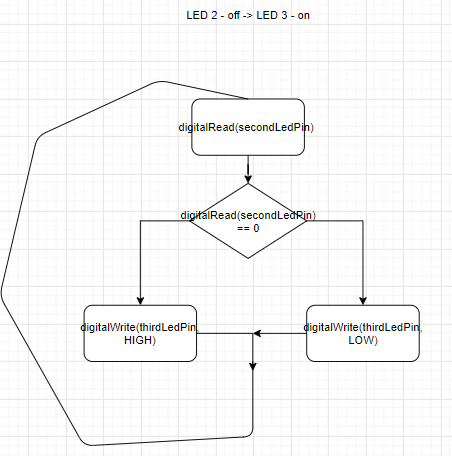


Figura 1.4 Schema bloc Task4 versiunea Secvențial (Task – LED2off –> 2 LED3on)

**Problema nr.2:** versiunea cu FreeRTOS

## Materiale și metode:

1. Trei LED;
2. Un Microcontroler;
3. 3 Rezistență;
4. Trei butoane;

## Rezultate:

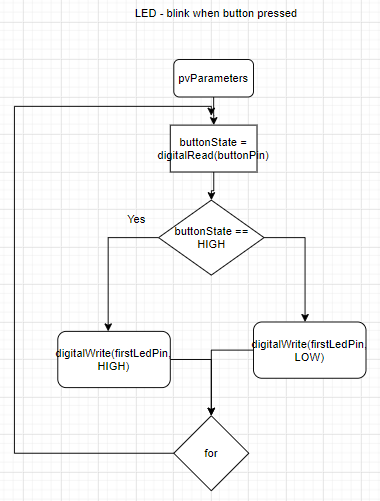


Figura 1.5 Schema bloc Task1 versiunea FreeRTOS (Task – LED blink - buton)

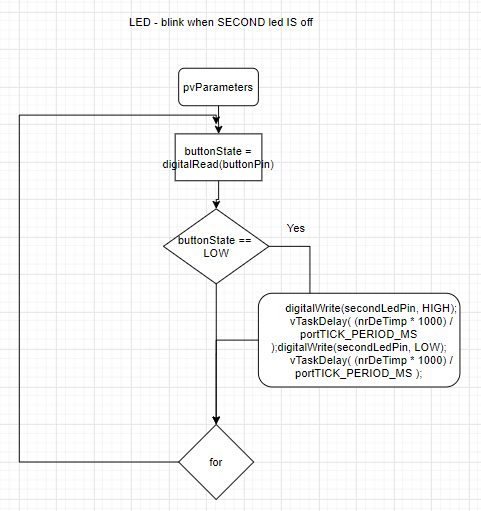


Figura 1.6 Schema bloc Task2 versiunea FreeRTOS (Task – LED blink – LED2)

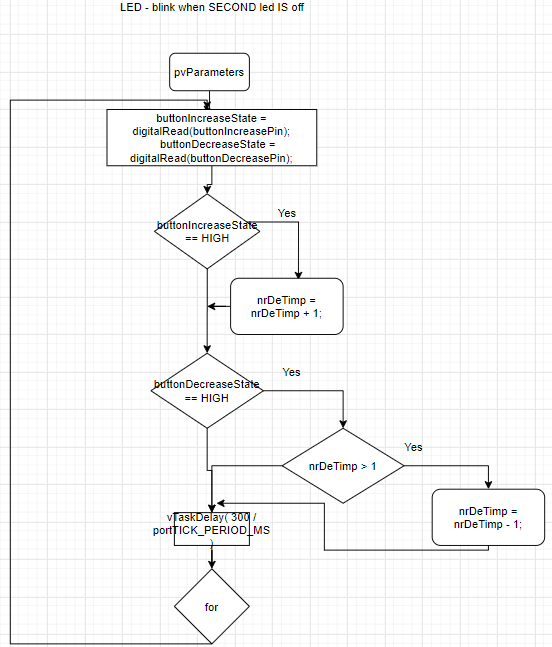


Figura 1.7 Schema bloc Task3 versiunea FreeRTOS (Task – LED – 2 butoane)

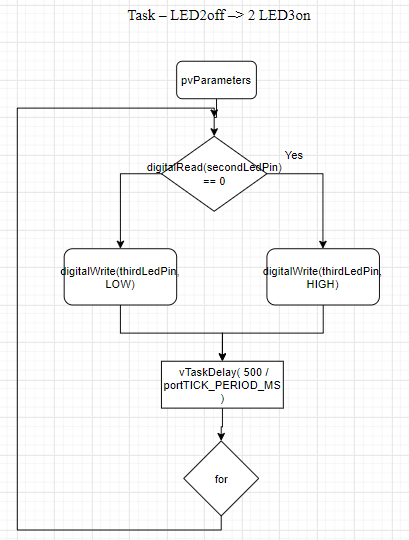


Figura 1.8 Schema bloc Task4 versiunea FreeRTOS (Task – LED2off –> 2 LED3on)

Schema Problema:

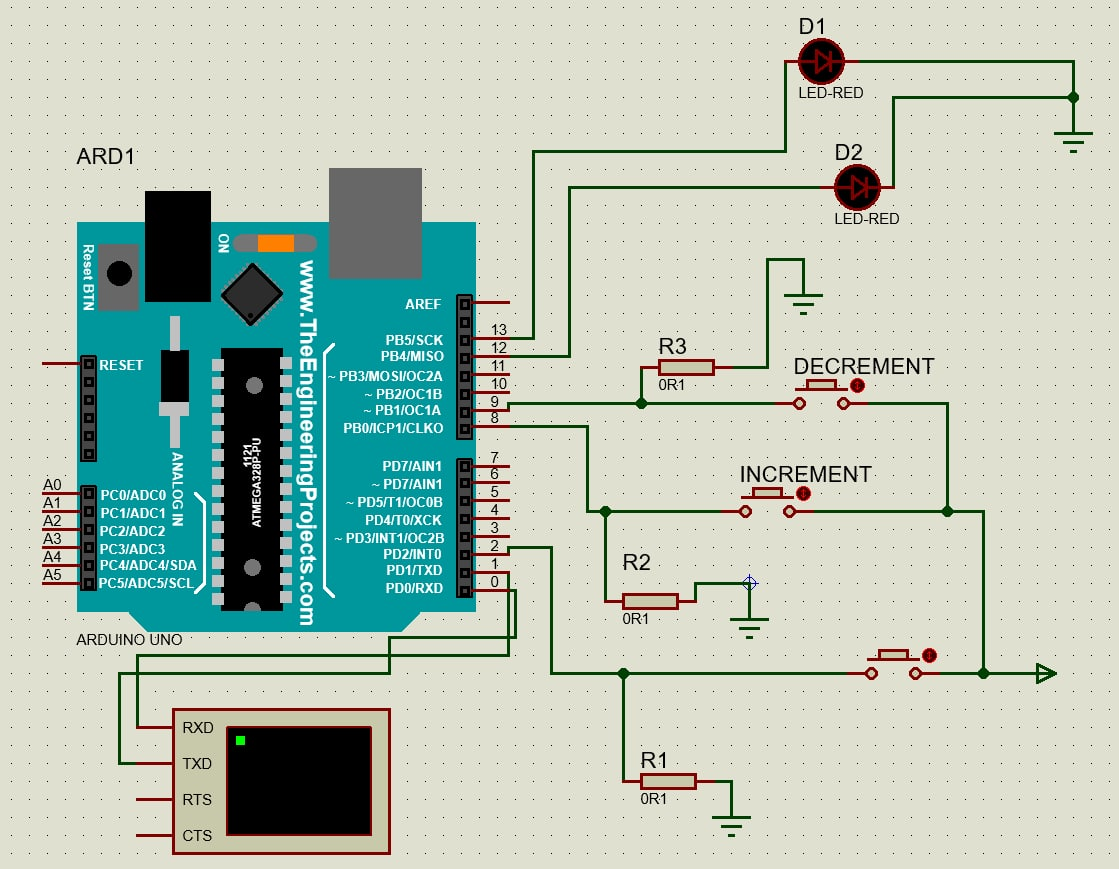


Figura 1.9 Circuit Aplicație

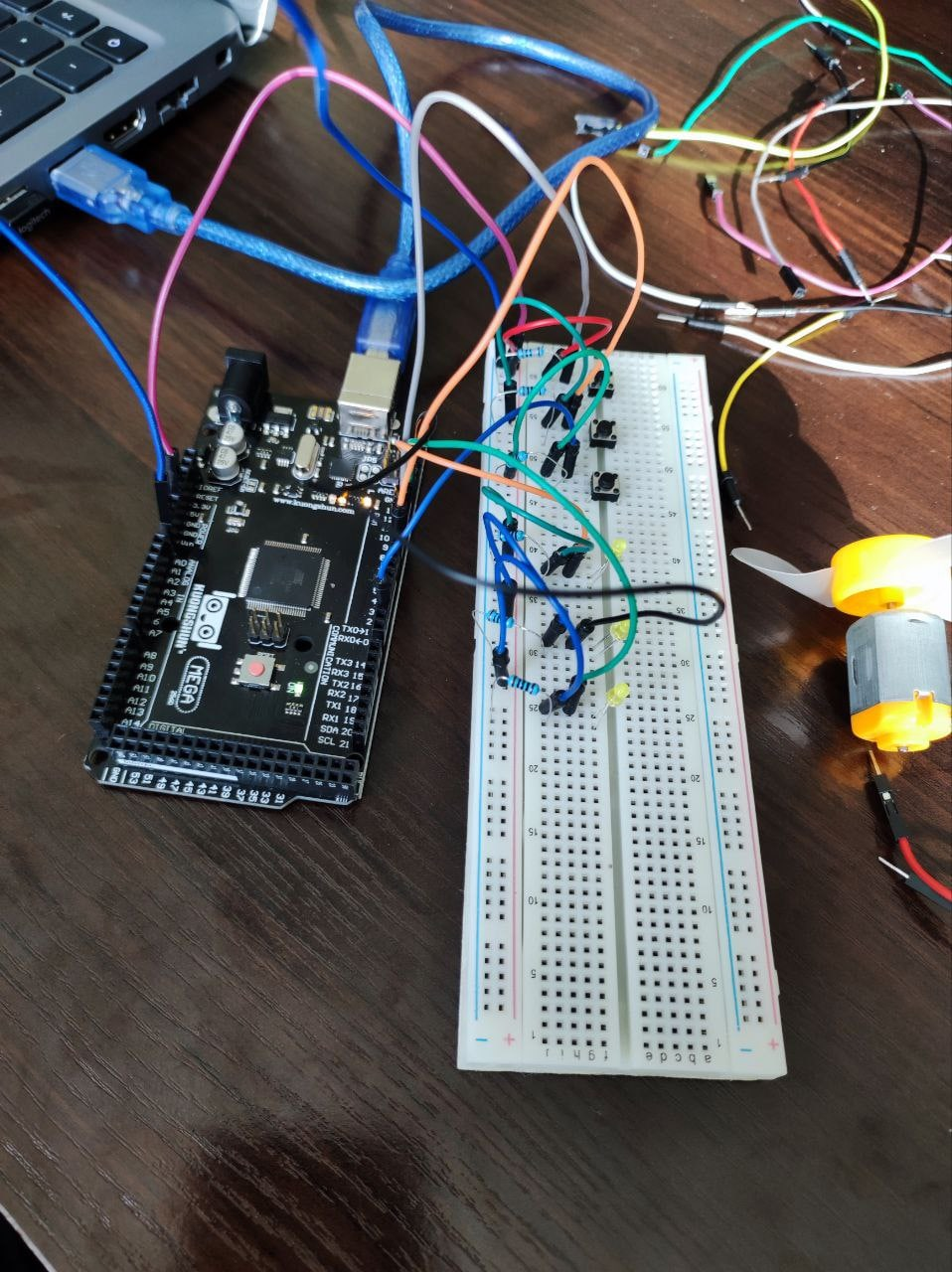


Figura 2.0 Schema asamblata

# Discuții:

Pentru simularea aplicației a fost utilizat Proteus, iar pentru scrierea codului IDE-ul Arduino. A fost utilizate elemente precum LED-uri, butoane, MCU Arduino și rezinstențe. Aplicația a fost executată în mod secvențial cât și FreeRTOS.

# Concluzii:

În cadrul lucrării date au fost efectuate 2 programe pentru simularea situațiilor : aplicație – mod Secvențial, aplicație – mod FreeRTOS. Fiecare din aceste 2 programe conțin câte 4 task-uri. Pentru simularea aplicațiilor a fost modelat sistemul în Proteus. Componentele sistemului au reprezentat 3 LED-uri, 2 btoane și un MCU.

# Bibliografia:

1. <https://www.arduino.cc/en/Tutorial/HomePage?from=Main.Tutorials>
2. <https://www.freertos.org/about-RTOS.html>
3. https://else.fcim.utm.md/mod/assign/view.php?id=8