Programarea Aplicatiilor in Timp Real

Sem. 2, 2022-2023

Tema de casa PATR

-Imprimanta 3D-

Elev: Rizea Radu

Grupa 341B2

Contents

1 Introducere Definire Problema

Problema abordata de echipa noastra este imprimanta 3D.

Performantele urmarite de noi sunt urmatoarele:

- Mentinerea temperaturii contante(Task_A)
- Verificarea filamentului la fiecare strat printat(Task_B)
- Citirea si interpretarea Gcode-ului, motoarele/extrouder-ul se rotesc cu pasii corespunzatori. (Task_C)

2 Analiza problemei

Mentinerea temperaturii:

Pentru aceasta cerinta de performanta, termistorul imi va da o valoare citita de pe hot end-ul imprimantei 3D, iar cu ajutorul unui algoritm PID, voi determina cat voltaj trebuie sa aplic rezistentei care incalzeste ansamblul.

La fiecare 50 de ms voi introduce in sistem o intrerupere pentru a verifica daca mai exista filament valabil pentru printare. Daca nu mai exista, tot ansamblul se va indeparta de obiectul printat si imprimanta va intra in stand by.

Din memoria externa se va decodifica fiecare linie de cod si se va stoca comanda in memoria interna, apoi se va decoda ca mai departe sa fie executata de catre motoare/extrouder.

3 Definirea structurii aplicatiei

Pentru buna functionare a procesului vor fi indeplinite urmatoarele cerinte:

- Intreg procesul de printare va incepe cu Task_A si va ramane pornit pe tot parcursul printarii.
- Dupa ce se atinge temperatura corespunzatoare, va incepe Task_C
- Task_A va fi sincronizat cu Task_C pentru a preveni distrugerea componentelor in timp
- Dupa 50ms, Task_C va fi oprit pentru a porni Task_B, in timpul acesta, Task_A nu se va opri

- Daca se va stinge curentul, se va executa aceeasi intrerupere ca in cazul Task_B
- Dupa ce se va executa tot Gcode-ul, imprimanta va face homing

4 DEFINIREA SOLUTIEI IN VEDEREA IMPLEMENTARII

Vom folosi Arduino si becuri pentru a semnifica functionarea fiecarui task.

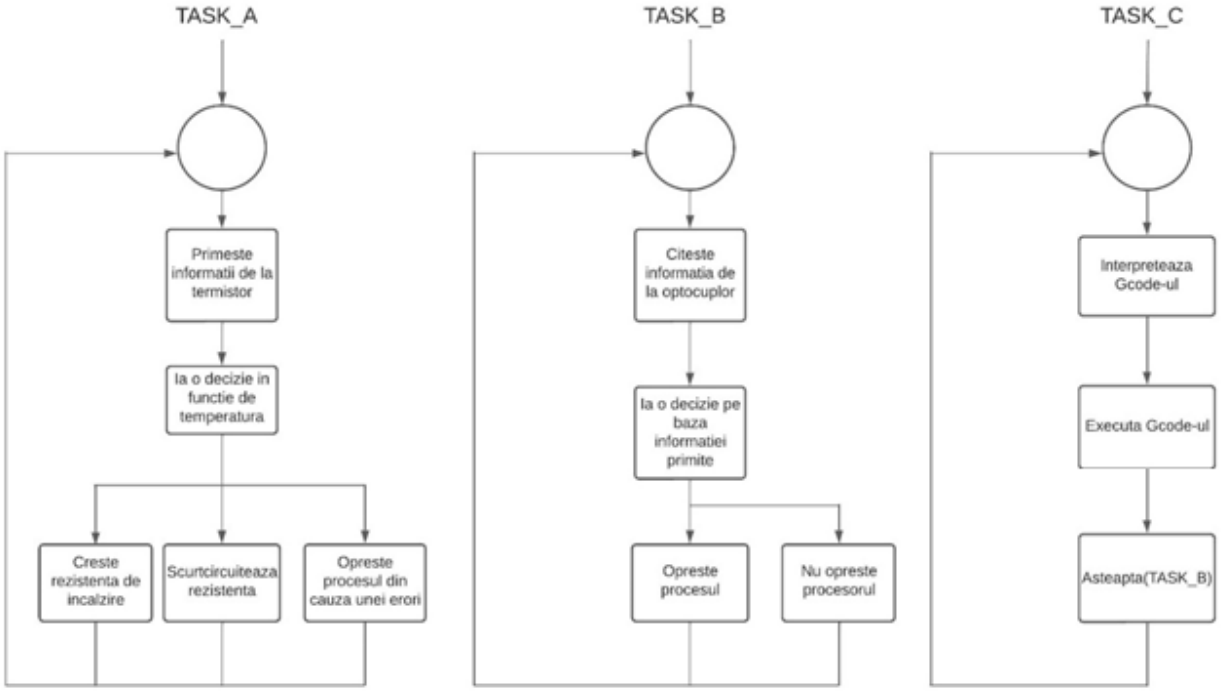
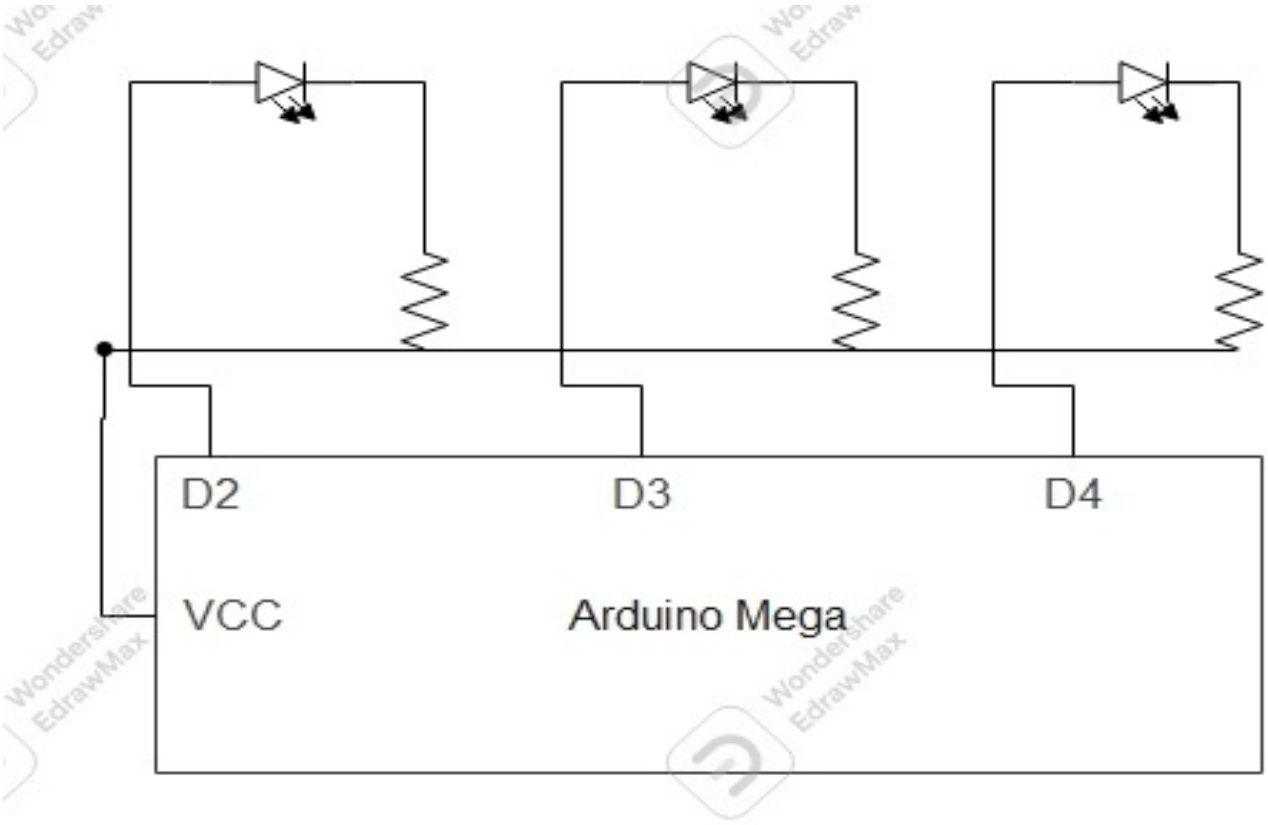
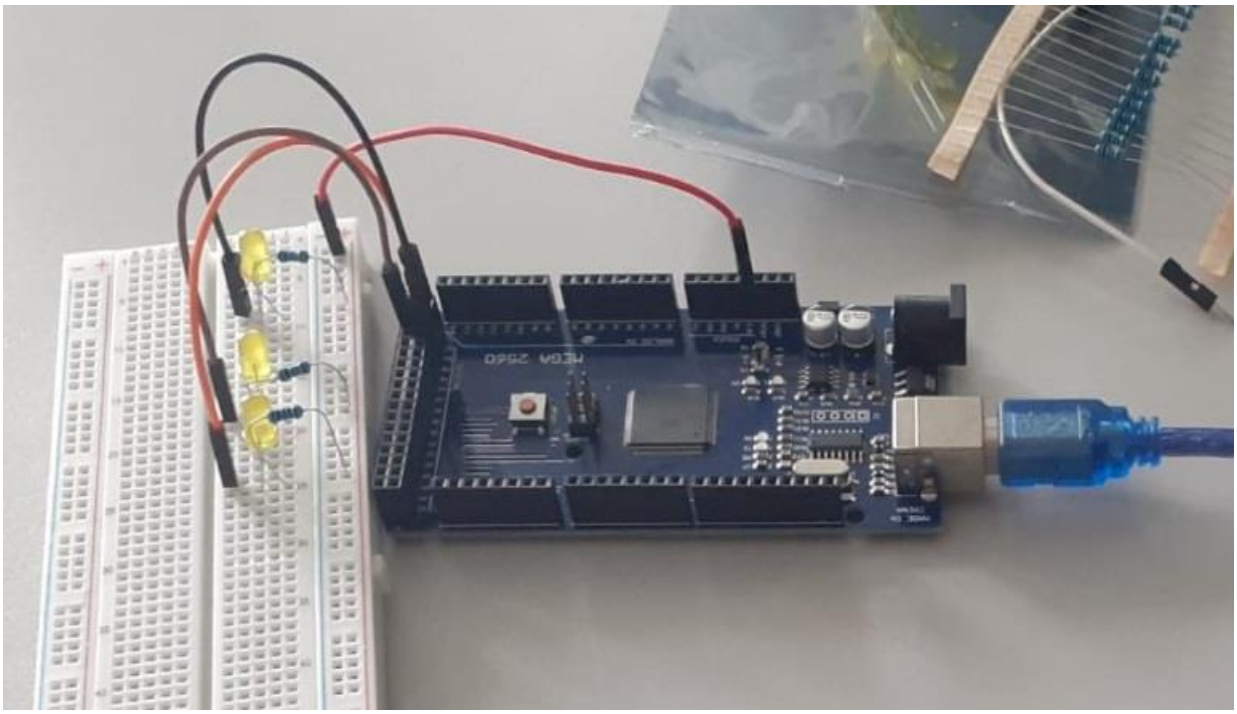


Diagrama elecrica a implementarii nostre este urmatoarea:





5 IMPLEMENTAREA SOLUTIEI

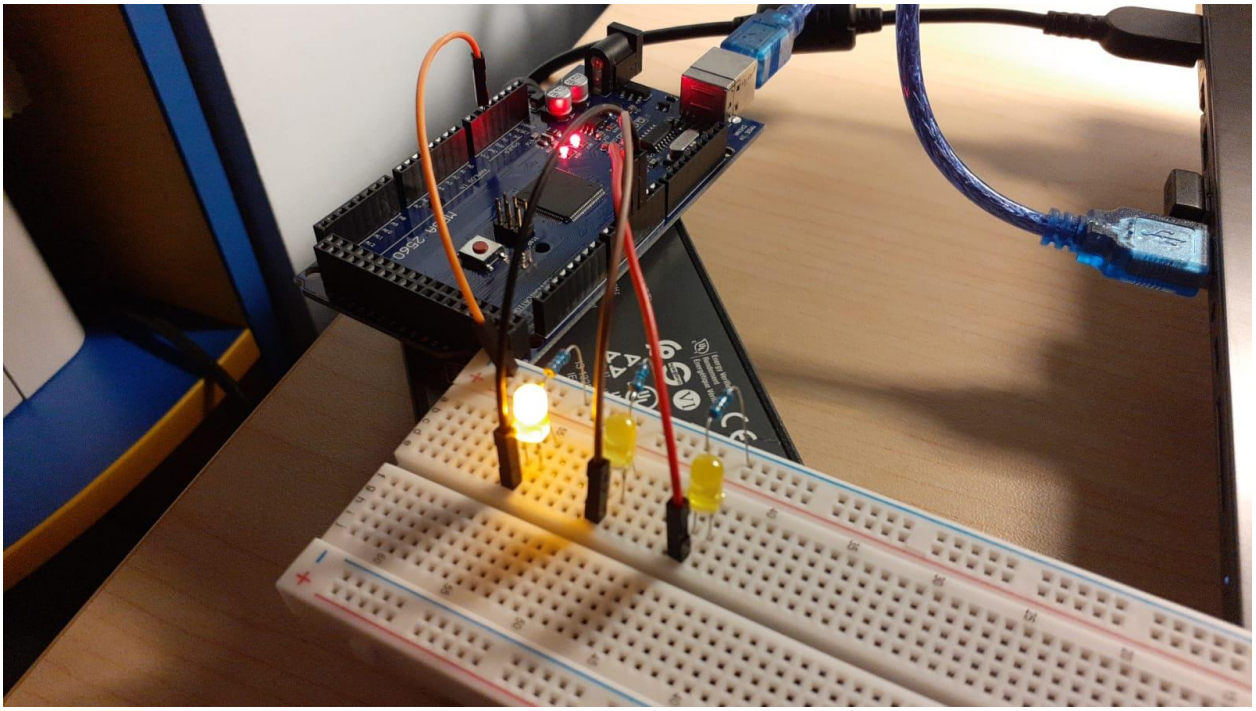
Codul aplicatiei este:

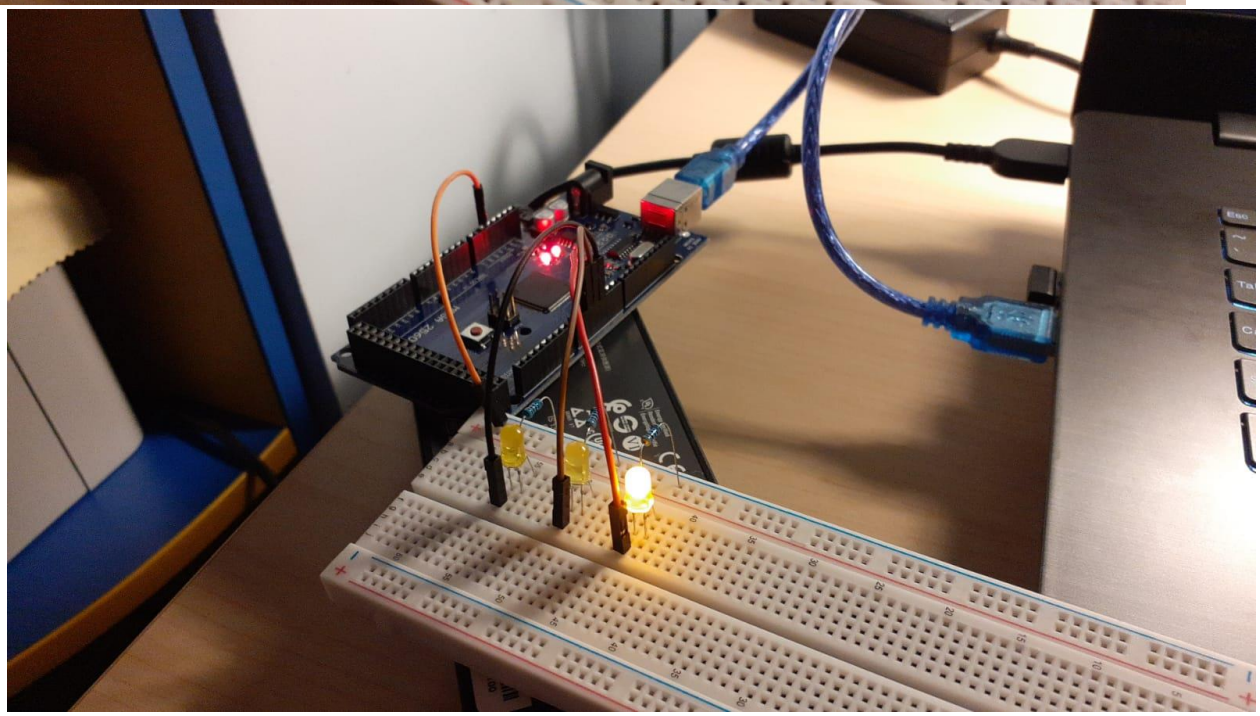
```
#include <Arduino_FreeRTOS.h> // Este inclusa biblioteca FreeRTOS
#include <semphr.h> // Este inclusa biblioteca semhr.h, pentru utilizarea
mutexurilor

SemaphoreHandle_t mutex;
SemaphoreHandle_t mutex2;
int ok;
void setup() {
  // put your setup code here, to run once:
  pinMode(2, OUTPUT); //TASK_A
  pinMode(3, OUTPUT); //TASK_B
  pinMode(4, OUTPUT); //TASK_C
  Serial.begin(9600); // Este initializata comunicatia seriala
  ok = 0;
  Serial.println("OK este 0");
  mutex = xSemaphoreCreateMutex();

  // Sunt create cele 3 taskuri, fiecare cu 128 de cuvinte de memorie rezervate
si prioritate egala cu 1
  xTaskCreate(TASK_A, "task1", 128, NULL, 1, NULL);
  xTaskCreate(TASK_B, "task2", 128, NULL, 1, NULL);
  xTaskCreate(TASK_C, "task3", 128, NULL, 1, NULL);
  vTaskStartScheduler();
}

int i = 20;

void TASK_A(){
  while(1){
    if(i < 200){
      xSemaphoreTake(mutex, portMAX_DELAY);

      for(i = 190; i < 200; i++){
          digitalWrite(2, HIGH);
          vTaskDelay(200 / portTICK_PERIOD_MS);
          Serial.println(i);
          digitalWrite(2, LOW);
          vTaskDelay(200 / portTICK_PERIOD_MS);
      }
      xSemaphoreGive(mutex);
      digitalWrite(2, HIGH);
    }
    else{
      ok = 1;
      Serial.println(i);
      digitalWrite(2, LOW);
    }

    vTaskDelay(200 / portTICK_PERIOD_MS);

  }
}
```

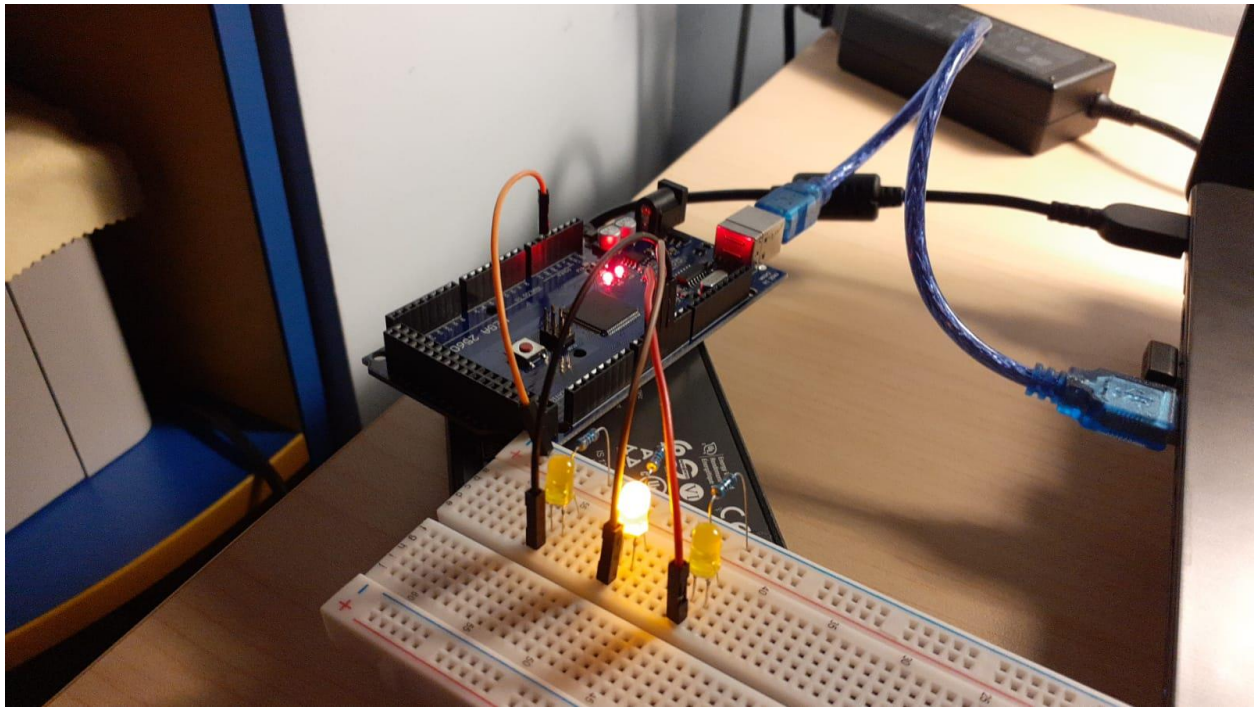
```
void TASK_C(){

    while(1){
        Serial.println("citeste OK");
       if(ok == 1){
        xSemaphoreTake(mutex, portMAX_DELAY);
        Serial.println("Interpreteaza Gcode-ul");
        digitalWrite(4, HIGH);
        delay(800);
        Serial.println(i);
        digitalWrite(4, LOW);
        delay(800);
        ok = 1;
        TASK_B();
    }
  }
}

void TASK_B(){

    while(1){
      if(ok == 1){
        Serial.println("Citeste informatia de la optocuplor");
        digitalWrite(3, HIGH);
        delay(400);
        Serial.println(i);
        digitalWrite(3, LOW);
        delay(400);
        ok = 1;
        xSemaphoreGive(mutex);
        TASK_C();
    }
  }
}


void loop() {}
```

6 TESTAREA APLICATIEI SI VALIDAREA SOLUTIEI PROPUSE

Dupa cum puteti vedea in imaginile de mai jos, am exemplificat diferite stari de functionare ale programului:







Output-ul programului din serial dupa numararea pana la 200 a task-ului A este urmatorul: