MINISTERUL EDUCAŢIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA
Universitatea Tehnică a Moldovei
Facultatea Calculatoare, Informatică și Microelectronică
Departamentul Ingineria Software și Automatică
Programul de studii: Tehnologia informației

RAPORT

Disciplina „IoT – Internetul Lucrurilor”
Tema: Interacțiunea cu utilizatorul

Student(ă): _____ **Vlașițchi Ștefan , TI-212**

Coordonator universitate: _____ **Lupan Cristian, asist.univ.**

Chișinău, 2024

1. **Button + LED**

Obiective:

1. Crearea unei aplicații MCU care să comunice cu utilizatorul prin intermediul unei interfețe seriale utilizând biblioteca STDIO.
2. Implementarea butonului pentru controlul unui LED, incluzând aprinderea și stingerea acestuia.
3. Validarea funcționării aplicației prin simulare utilizând un mediu adecvat, precum ArduinoIDE, și separarea funcționalităților în fișiere pentru reutilizare ulterioară.

Introducere

Internetul Lucrurilor (IoT) reprezintă interconectarea dispozitivelor fizice prin intermediul internetului pentru a permite schimbul de date și interacțiunea automatizată între ele. În contextul acestui laborator, utilizarea microcontrolerelor (MCU) pentru a implementa interacțiunea cu un utilizator printr-o interfață serială este esențială pentru prototiparea și testarea aplicațiilor IoT. O problemă actuală abordată este controlul simplu al echipamentelor periferice, precum LED-uri, prin comenzi trimise prin terminal, permițând dezvoltatorilor să testeze rapid funcționalitatea fără a depinde de o interfață grafică. Referințe studiate includ documentația STDIO pentru microcontrolere și tutorialele de configurare IDE pentru dezvoltare și simulare.

Materiale si Metode utilizate

Materiale utilizate:

- Microcontroler cu suport pentru interfața serială(Arduino).
- LED, buton și rezistor.
- IDE cu suport pentru Arduino și multiple fișiere, precum ArduinoIDE.
- Simulator Wokwi pentru validarea proiectului.

Metodologie:

- Configurarea MCU pentru a primi comenzi seriale utilizând biblioteca STDIO.
- Dezvoltarea unei logici pentru aprinderea și stingerea LED-ului pe baza butonului
- Implementarea mesajelor de confirmare a comenzilor în terminal.
- Separarea funcționalităților într-un fișier dedicat pentru LED, astfel încât să fie reutilizabil în alte proiecte.

Rezultate

Schema bloc structurală:

Un bloc principal (Arduino UNO) pentru interacțiunea serială care primește comenzi de la buton și decide asupra stării LED-ului (ON/OFF).

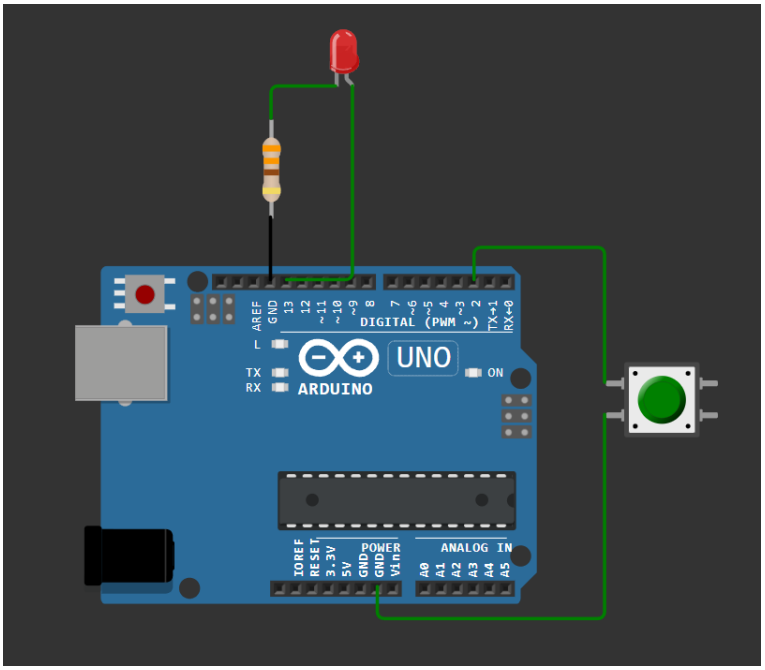

Figura 1.1 – Schema bloc structurala

Cod Cheie:

```
void loop() {
  buttonState = digitalRead(buttonPin); // Citirea stării butonului

  // Verifică dacă starea butonului a fost modificată
  if (buttonState != lastButtonState) {
    // Dacă butonul a fost apăsat
    if (buttonState == LOW) {
      // Schimbă starea LED-ului
      ledState = !ledState;
      digitalWrite(ledPin, ledState);
    }
    // Salvează starea curentă a butonului
    lastButtonState = buttonState;
  }
}
```

Această secvență de cod demonstreaza funcționalitatea esențială pentru aprinderea și stingerea LED-ului prin apasarea butonului.

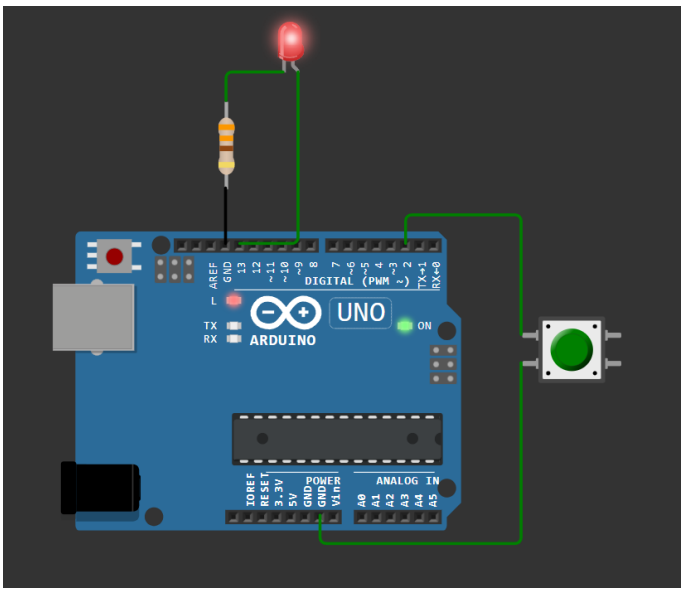

Figura 1.2 – Led-ul pornit

Pentru a aprinde ledul este necesar sa apasam butonul pentru a aprinde led-ul cand va fi apasat din nou led-ul se va stinge

Concluzie

Rezultatele obținute au demonstrat o interacțiune corectă între utilizator și sistem prin intermediul unui buton.Comparativ cu alte soluții de control, acest proiect utilizează o metodă simplă și directă, fără utilizarea unor biblioteci complexe externe. Prin implementarea funcționalităților periferice în fișiere separate, codul este mai modular și ușor de reutilizat în proiecte viitoare. Utilizarea simulatorului Wokwi a oferit o modalitate eficientă de testare fără a avea nevoie de hardware fizic.

Anexa A

stdinout.h

```
#ifndef _STDINOUT_H
#define _STDINOUT_H

// no need to make an instance of this yourself
class initializeSTDINOUT
{
        static size_t initnum;
public:
        // Constructor
        initializeSTDINOUT();
};

// Call the constructor in each compiled file this header is included in
// static means the names won't collide
static initializeSTDINOUT initializeSTDINOUT_obj;

#endif
```

stdinout.cpp

```
#if ARDUINO >= 100
#include "Arduino.h"
#else
#include "WProgram.h"
#endif
#include <stdio.h>
#include "stdinout.h"

// Function that printf and related will use to print
static int serial_putchar(char c, FILE *f)
{
  if (c == '\n') {
    serial_putchar('\r', f);
  }

  return Serial.write(c) == 1 ? 0 : 1;
}
// Function that scanf and related will use to read
static int serial_getchar(FILE *)
{
  // Wait until character is avilable
  while (Serial.available() <= 0) { ; }

  return Serial.read();
}

static FILE serial_stdinout;

static void setup_stdin_stdout()
{
  // Set up stdout and stdin
  fdev_setup_stream(&serial_stdinout, serial_putchar, serial_getchar, _FDEV_SETUP_RW);
  stdout = &serial_stdinout;
  stdin  = &serial_stdinout;
  stderr = &serial_stdinout;
}

// Initialize the static variable to 0
size_t initializeSTDINOUT::initnum = 0;

// Constructor that calls the function to set up stdin and stdout
```

```
initializeSTDINOUT::initializeSTDINOUT()
{
  if (initnum++ == 0) {
    setup_stdin_stdout();
  }
}
```

Anexa B

main.ino

```
const int buttonPin = 2;   // Pinul la care este conectat butonul
const int ledPin = 13;     // Pinul la care este conectat LED-ul

int ledState = LOW;         // Starea inițială a LED-ului (stins)
int buttonState;            // Starea curentă a butonului
int lastButtonState = LOW;  // Starea precedentă a butonului

void setup() {
  pinMode(ledPin, OUTPUT);      // Configurarea pinului LED ca ieșire
  pinMode(buttonPin, INPUT_PULLUP); // Configurarea pinului butonului ca intrare cu pull-up
intern
}

void loop() {
  buttonState = digitalRead(buttonPin); // Citirea stării butonului

  // Verifică dacă starea butonului a fost modificată
  if (buttonState != lastButtonState) {
    // Dacă butonul a fost apăsat
    if (buttonState == LOW) {
      // Schimbă starea LED-ului
      ledState = !ledState;
      digitalWrite(ledPin, ledState);
    }
    // Salvează starea curentă a butonului
    lastButtonState = buttonState;
  }
}
```