**MINISTERUL EDUCAŢIEI, CULTURII ȘI CERCETĂRII**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare Informatică şi Microelectronică**

 **Departamentul Ingineria Software și Automatică**

Raport

**la disciplina Internetul lucrurilor IOT**

**Tema: Automate finite**

A efectuat: st.gr. TI-204 Agatiev Dumitru

A verificat: asist.univ. Litra Dinu

Chișinău 2023

**Definirea problemei 1**

Realizarea unei aplicatie in baza de MCU care reprezinta un Automat Finit aplicatie Button-Led.

**Obiective**

* configurarea aplicației pentru automat finit Button-led;
* crearea schemei și codului conform sarcinii propuse;
* înțelegerea sistemului creat și procesele ce se întâmplă.

**Metode și materiale**

Materiale Necesare:

* Arduino Board (de exemplu, Arduino Uno);
* breadboard și fire de conexiune;
* trei leduri;
* o telecomanda;
* un senzor IR.

Metoda de Implementare:

* conexiuni hardware:
* conectarea senzorului;
* conectarea LED.
* scrierea codului în Arduino IDE:
* definirea variabilelor și pinilor corespunzători pentru senzor si leduri;
* testarea și debugging:
* încărcare cod pe Arduino folosind Arduino IDE;
* monitorizare comportament senzor, led;
* identificare și rezolvare eventualelor erori în cod.

**Rezultate**

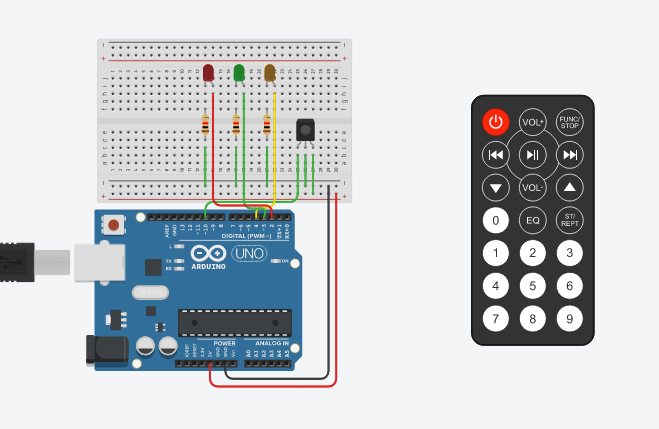
**

Figura 1 – Asamblarea circuitului pentru sarcina 1

**Definirea problemei 2**

Realizarea unei aplicații in baza de MCU care va reprezenta un Automat Finit aplicatie Semafor.

**Obiective**

* configurarea aplicației pentru un semafor automat finit;
* crearea schemei și codului conform sarcinii propuse;
* înțelegerea sistemului creat și procesele ce se întâmplă.

**Metode și materiale**

Materiale Necesare:

* Arduino Board (de exemplu, Arduino Uno);
* breadboard și fire de conexiune;
* editor Arduino IDE instalat pe un calculator;
* trei leduri.

Metoda de Implementare:

* conexiuni hardware:
* conectare Led.
* scrierea codului în Arduino IDE:
* definirea variabilelor și pinilor corespunzători pentru leduri.
* testarea și debugging:
* încărcare cod pe Arduino folosind Arduino IDE;
* monitorizare comportament LED;
* identificare și rezolvare eventualelor erori în cod.

**Rezultate**

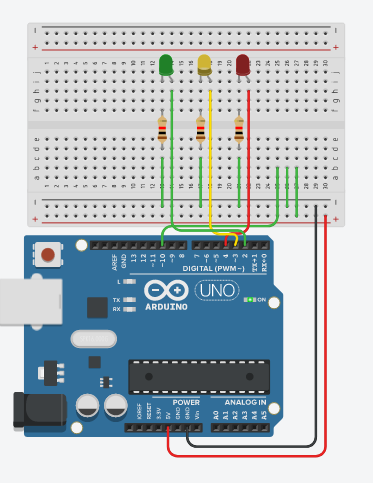
**

Figura 2 – Asamblarea circuitului pentru sarcina 2

**Concluzii**

Laboratorul s-a concentrat pe explorarea și aplicarea conceptului de automat finit în două scenarii distincte. În primul scenariu, am utilizat un receptor IR și butoanele unei telecomenzi pentru a controla LED-urile, demonstrând astfel capacitatea unui automat finit de a gestiona dispozitive externe în mod corespunzător. În al doilea scenariu, am simulat funcționarea unui semafor rutier, evidențiind utilitatea automatelor finite în proiectarea și controlul sistemelor. Laboratorul a subliniat capacitatea acestor automate de a modela comportamente și de a oferi o bază solidă în proiectarea sistemelor de control și în simularea funcționării sistemelor complexe, reprezentând astfel o unealtă valoroasă în dezvoltarea tehnologică.

**Anexa 1 – Codul sursă sarcina 1**

#include <IRremote.h>

#define IR\_RECEIVE\_PIN 10

#define BUTTON\_LEFT 4

#define BUTTON\_RIGHT 6

#define BUTTON\_VOLPLUS 1

#define BUTTON\_VOLMINUS 9

int yellowLed = 3;

int greenLed = 2;

int redLed = 4;

enum State { OFF, RED, GREEN, YELLOW };

State currentState = OFF;

State previousState = OFF;

void transition(State newState);

void setup() {

Serial.begin(9600);

IrReceiver.begin(IR\_RECEIVE\_PIN);

pinMode(redLed, OUTPUT);

pinMode(greenLed, OUTPUT);

pinMode(yellowLed, OUTPUT);

}

void loop() {

if (IrReceiver.decode()) {

IrReceiver.resume();

int command = IrReceiver.decodedIRData.command;

Serial.println(command);

switch (command) {

case BUTTON\_LEFT:

transition(GREEN);

break;

case BUTTON\_RIGHT:

transition(YELLOW);

break;

case BUTTON\_VOLPLUS:

transition(RED);

break;

case BUTTON\_VOLMINUS:

transition(OFF);

break;

default:

break;

} }}

void transition(State newState) {

switch (previousState) {

case RED:

digitalWrite(redLed, LOW);

break;

case GREEN:

digitalWrite(greenLed, LOW);

break;

case YELLOW:

digitalWrite(yellowLed, LOW);

break;

case OFF:

digitalWrite(yellowLed, LOW);

digitalWrite(greenLed, LOW);

digitalWrite(redLed, LOW);

break;

}

switch (newState) {

case RED:

digitalWrite(redLed, HIGH);

Serial.println("Red");

break;

case GREEN:

digitalWrite(greenLed, HIGH);

Serial.println("Green");

break;

case YELLOW:

digitalWrite(yellowLed, HIGH);

Serial.println("Yellow");

break;

case OFF:

digitalWrite(yellowLed, LOW);

digitalWrite(greenLed, LOW);

digitalWrite(redLed, LOW);

Serial.println("off");

break;

}

previousState = currentState;

currentState = newState;

}

**Anexa 2 – Codul sursă sarcina 2**

int yellowLed = 3;

int greenLed = 2;

int redLed = 4;

enum State { RED, GREEN, YELLOW };

State currentState = GREEN;

State previousState = GREEN;

void setup() {

Serial.begin(9600);

pinMode(redLed, OUTPUT);

pinMode(greenLed, OUTPUT);

pinMode(yellowLed, OUTPUT);

digitalWrite(greenLed, HIGH);

digitalWrite(yellowLed, LOW);

digitalWrite(redLed, LOW);

}

void loop() {

switch (currentState) {

case RED:

digitalWrite(redLed, HIGH);

digitalWrite(yellowLed, LOW);

digitalWrite(greenLed, LOW);

delay(3000);

currentState = YELLOW;

previousState = RED;

break;

case GREEN:

digitalWrite(redLed, LOW);

digitalWrite(yellowLed, LOW);

digitalWrite(greenLed, HIGH);

delay(3000);

currentState = YELLOW;

previousState = GREEN;

break;

case YELLOW:

digitalWrite(redLed, LOW);

digitalWrite(yellowLed, HIGH);

digitalWrite(greenLed, LOW);

delay(3000);

if (previousState == RED)

currentState = GREEN;

else

currentState = RED;

break;

}

}