міністерство освіти і науки України

**Прикарпатський національний університет**

**імені В.Стефаника**

*Факультет математики та інформатики*

*Кафедра інформаційних технологій*

*Програмування вбудованих систем*

Лабораторна робота № 4

Тема: «Взаємодія Arduino із пристроями через послідовні інтерфейси. Проект "Таймер"»

Варіант : 6

Виконала: Петрів В. В.

Група ІПЗ-23

Дата: 10 листопада 2023р.

Викладач: Лазарович І.М.

Івано-Франківськ - 2023

**Завдання**

Розробити проект цифрового таймера. Роботу виконувати в емуляторі за URL-адресою https://wokwi.com

Вимоги до схеми. Схема в Proteus повинна містити: Arduino UNO, РКІ (згідно варіанту в таблиці 1), RTC типу DS1307, три кнопки (K1, K2, K3), і модуль реле (Relay Module). Для свого типу РКІ використати необхідну бібліотеку для Arduino.

Пристрій повинен реалізовувати наступні алгоритми:

1. На РКІ в звичайному режимі роботи повинен відображатись поточний час у форматі ГГ:ХХ:СС (години, хвилини і секунди, по 2 розряди на кожне число) – це "основний екран".

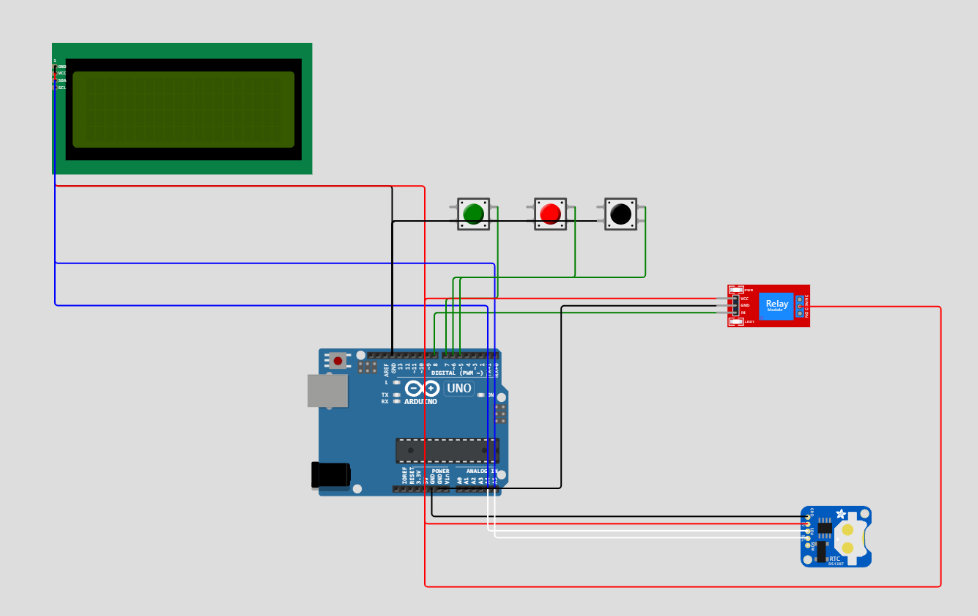
2. При короткому натисненні (до 1 сек) на К1 на РКІ повинні відображатись час ввімкнення і час вимкнення реле у форматі T.ON= ГГ:ХХ:СС, T.OFF= ГГ:ХХ:СС – це "екран таймера". Якщо протягом 10сек не натиснуто жодної кнопки, то таймер повинен перейти в основний екран.

3. Якщо активний "екран таймера", при тривалому натисненні К1 (>1сек) таймер повинен перейти в режим встановлення часу спрацювання T.ON і T.OFF. При цьому повинні блимати цифри, які відображають години T.ON – це режим "зміни часу таймера". Коротке натиснення К2 має збільшувати години, а коротке натиснення К3 – їх зменшувати. Далі коротке натиснення К1 переходить до зміни хвилин T.ON, і так далі по всіх параметрах T.ON і T.OFF. Тривале натиснення К1 в режимі "зміни часу таймера" – перехід в режим "екран таймера".

4. Якщо поточний час знаходиться в проміжку від T.ON до T.OFF, повинно спрацювати реле Р1. Якщо час поза межами цього інтервалу – то реле повинно розімкнути контакти.

Рекомендовано завдання виконувати групою із 2 студентів, де один реалізує схему, та пункти 1 і 4 алгоритму роботи таймера, а інший студент – пункти 2 і 3.

**Схема**

****

**Посилання на проект:**

<https://wokwi.com/projects/381236339728497665>

**Програмний код**

#include <Wire.h>

#include <RTClib.h>

#include <LiquidCrystal\_I2C.h>

RTC\_DS1307 realTimeClock;

LiquidCrystal\_I2C lcd(0x27, 20, 4);

DateTime startTime (2023, 11, 14, 13, 18, 00);

DateTime endTime (2023, 11, 14, 16, 28, 59);

int relayPin = 8;

int buttonPinA = 7, buttonPinB = 6, buttonPinC = 5;

int buttonStateA, buttonStateB, buttonStateC;

int lastButtonStateA = LOW, lastButtonStateB = LOW, lastButtonStateC = LOW;

unsigned long lastDebounceTimeA = 0, lastDebounceTimeB = 0, lastDebounceTimeC = 0;

unsigned long debounce = 50;

unsigned long shortInterval = 1000;

unsigned long outInterval = 10000;

int previousMillis = 0;

int pressDurationA = 0, pressDurationB = 0, pressDurationC = 0;

int state = 0;

bool timeSettingState = false;

int timeSwitchingState = 0;

int blinkPosition = 0;

bool blinkState = false;

void setup() {

**Wire**.begin();

  realTimeClock.begin();

  lcd.init();

  lcd.backlight();

  pinMode(buttonPinA, INPUT\_PULLUP);

  pinMode(buttonPinB, INPUT\_PULLUP);

  pinMode(buttonPinC, INPUT\_PULLUP);

  pinMode(relayPin, OUTPUT);

}

void displayTime(int time, bool notLast = true) {

  if (time < 10) lcd.print("0");

  lcd.print(time);

  if (notLast)

    lcd.print(":");

}

DateTime addTime(DateTime dt, int timeSwitchingState) {

  switch (timeSwitchingState) {

    case 0:

      return dt + TimeSpan(0, 1, 0, 0);

    case 1:

      return dt + TimeSpan(0, 0, 1, 0);

    case 2:

      return dt + TimeSpan(0, 0, 0, 1);

  }

}

DateTime subTime(DateTime dt, int timeSwitchingState) {

  switch (timeSwitchingState) {

    case 0:

      return dt - TimeSpan(0, 1, 0, 0);

    case 1:

      return dt - TimeSpan(0, 0, 1, 0);

    case 2:

      return dt - TimeSpan(0, 0, 0, 1);

  }

}

void switchRelay() {

  DateTime now = realTimeClock.now();

  if (now >= startTime && now <= endTime) {

    digitalWrite(relayPin, HIGH);

  } else {

    digitalWrite(relayPin, LOW);

  }

}

void loop() {

  switchRelay();

  DateTime now = realTimeClock.now();

  int currentMillis = millis();

  int readingA = digitalRead(buttonPinA);

  int readingB = digitalRead(buttonPinB);

  int readingC = digitalRead(buttonPinC);

  if (readingA != lastButtonStateA) {

    lastDebounceTimeA = currentMillis;

  }

  if (readingB != lastButtonStateB) {

    lastDebounceTimeB = currentMillis;

  }

  if (readingC != lastButtonStateC) {

    lastDebounceTimeC = currentMillis;

  }

  if ((currentMillis - lastDebounceTimeA) > debounce) {

    if (readingA != buttonStateA) {

      buttonStateA = readingA;

      if ((state == 2) && readingA && (pressDurationA > 0) && (pressDurationA < shortInterval)) {

        timeSwitchingState++;

        blinkPosition++;

        if (timeSwitchingState > 2) {

          timeSettingState = !timeSettingState;

          timeSwitchingState = 0;

        }

        if (blinkPosition > 5) {

          blinkPosition = 0;

        }

      }

    }

    if (buttonStateA == LOW) {

      pressDurationA = currentMillis - lastDebounceTimeA;

    }

  }

  if ((currentMillis - lastDebounceTimeB) > debounce) {

    if (readingB != buttonStateB) {

      buttonStateB = readingB;

      if ((pressDurationB < shortInterval) && (pressDurationB > 0) && (state == 2) && buttonStateB) {

        if (!timeSettingState) {

          startTime = addTime(startTime, timeSwitchingState);

        }

        else endTime = addTime(endTime, timeSwitchingState);

      }

    }

    if (buttonStateB == LOW) {

      pressDurationB = currentMillis - lastDebounceTimeB;

    }

  }

  if ((currentMillis - lastDebounceTimeC) > debounce) {

    if (readingC != buttonStateC) {

      buttonStateC = readingC;

      if ((pressDurationC < shortInterval) && (pressDurationC > 0) && (state == 2) && buttonStateC) {

        if (!timeSettingState) {

          startTime = subTime(startTime, timeSwitchingState);

        }

        else endTime = subTime(endTime, timeSwitchingState);

      }

    }

    if (buttonStateC == LOW) {

      pressDurationC = currentMillis - lastDebounceTimeC;

    }

  }

  if (state == 0) {

    if ((pressDurationA < shortInterval) && (pressDurationA > 0) && readingA) {

      state = 1;

    }

  }

  else if (state == 1) {

    if (endTime < startTime) {

      startTime = endTime;

    }

    if ((currentMillis - lastDebounceTimeA > outInterval) && (currentMillis - lastDebounceTimeB > outInterval) && (currentMillis - lastDebounceTimeC > outInterval)) {

      state = 0;

      pressDurationA = 0;

    }

    if ((pressDurationA > shortInterval) && readingA) {

      state = 2;

      pressDurationA = 0;

    }

  }

  else if (state == 2) {

    if ((pressDurationA > shortInterval) && readingA) {

      state = 1;

      pressDurationA = 0;

    }

  }

  lastButtonStateA = readingA;

  lastButtonStateB = readingB;

  lastButtonStateC = readingC;

  if (currentMillis - previousMillis > shortInterval) {

    previousMillis = currentMillis;

    lcd.clear();

    displayTime(now.hour());

    displayTime(now.minute());

    displayTime(now.second(), false);

    if (state == 1) {

      lcd.setCursor(0, 1);

      lcd.print("T.ON: ");

      displayTime(startTime.hour());

      displayTime(startTime.minute());

      displayTime(startTime.second(), false);

      lcd.setCursor(0, 2);

      lcd.print("T.OFF: ");

      displayTime(endTime.hour());

      displayTime(endTime.minute());

      displayTime(endTime.second(), false);

    }

    else if (state == 2) {

      lcd.setCursor(0, 1);

      lcd.print("T.ON: ");

      if (blinkPosition != 0 || blinkState) {

          displayTime(startTime.hour());

      } else {

          lcd.print("  :");

      }

      if (blinkPosition != 1 || blinkState) {

          displayTime(startTime.minute());

      } else {

          lcd.print("  :");

      }

      if (blinkPosition != 2 || blinkState) {

          displayTime(startTime.second(), false);

      } else {

          lcd.print("  ");

      }

      lcd.setCursor(0, 2);

      lcd.print("T.OFF: ");

      if (blinkPosition != 3 || blinkState) {

          displayTime(endTime.hour());

      } else {

          lcd.print("  :");

      }

      if (blinkPosition != 4 || blinkState) {

          displayTime(endTime.minute());

      } else {

          lcd.print("  :");

      }

      if (blinkPosition != 5 || blinkState) {

          displayTime(endTime.second(), false);

      } else {

          lcd.print("  ");

      }

      blinkState = !blinkState;

    }

  }

}