Міністерство освіти та науки України

Національний університет “Львівська політехніка”



Лабораторна робота №3  
з курсу «Мікроконтролери Ч.1»

РОБОТА З ТАЙМЕРАМИ, LCD-ДИСПЛЕЄМ НА БАЗІ HD44780

ТА КЛАВІАТУРОЮ 4х4

Варіант 9

**Виконав:**

студент групи ІР-21

Касараба В.В.

**Прийняла:**

ст.викладач кафедри КСА

Лагун І. І.

Львів 2021

## Завдання

1. Згідно варіанту завдання (таблиця 1) зібрати у пакеті симуляції

Proteus схему на основі МК ATmega2560 та написати програму

мовою C++ в Arduino IDE для реалізації вказаного завдання.

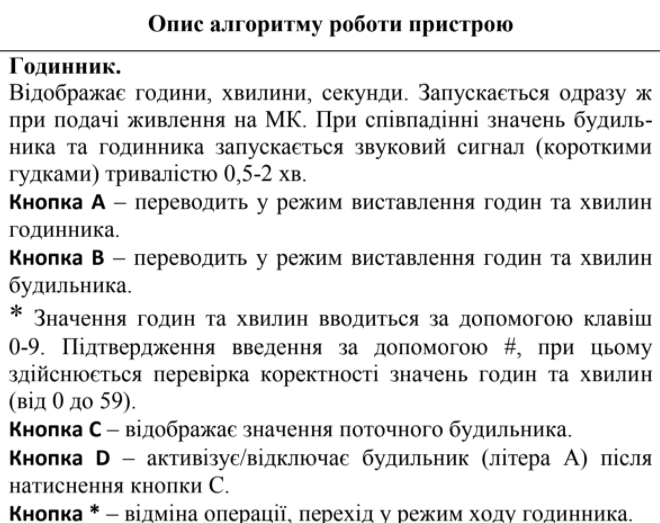
2. Розробити клієнтську програму для керування МК.

3. Протестувати роботу клієнтської програми зі схемою в Proteus.

4. Залити програму в Arduino-Kit.

5. Отримані результати представити викладачу.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | **Завдання**  1. Непарні номери варіантів використовують 8-ми розрядний  інтерфейс LCD для обміну даними, Парні номери варіантів  – 4-ри розрядний.  2. У завданні вказується таймер, за допомогою якого потрібно  реалізувати відлік інтервалів часу для зазначеного  пристрою.  3. Тривалість звукових сигналів вибирається студентом  самостійно, дотримуючись вказаних меж.  4. Натиск кожної кнопки супроводжується коротким  одинарним сигналом, успішне підтвердження введення  даних за допомогою клавіші # супроводжується коротким  подвійним сигналом, введення невірних даних за  допомогою клавіші # та відміна усієї операції за допомогою  клавіші \* супроводжується довгим сигналом. | | | |
|  | **Таймер**  **відліку часу** | **Клавіатура** | **Порт даних**  **LCD** | **Алгоритм** |
| 9 | T1 | port-K | port-L | 1 |

****

**Відеопрезентація**

<https://drive.google.com/file/d/1aj9XqQ7-XEliGR1CEuVzu9U_ucUovmUR/view?usp=sharing>

## Робочий процес

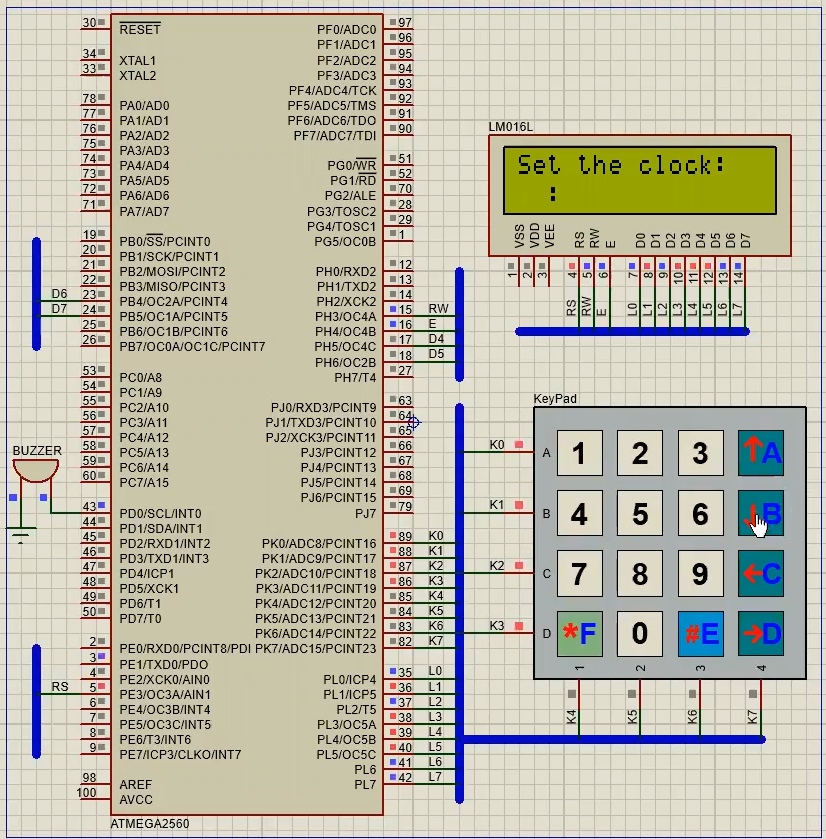
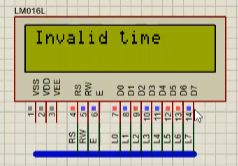
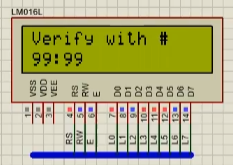
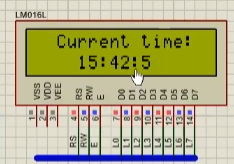


Рис. 1. Схема у пакеті модуляції Proteus



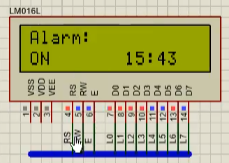
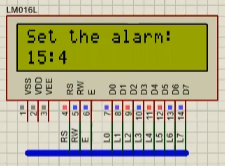


Рис. 2. Демонстрація LCD при різних операціях

**Код програми мікроконтролера:**

#include <Keypad.h>

#include <Wire.h>

#include <LiquidCrystal.h>

const int rs = 5, rw = 6, en = 7, d0 = 49, d1 = 48, d2 = 47, d3 = 46, d4 = 45, d5 = 44, d6 = 43, d7 = 42, buzzerPin = 21;

LiquidCrystal lcd(rs, rw, en, d0, d1, d2, d3, d4, d5, d6, d7);

String hrSet,hrSetTimer, minSetTimer, countNumberStringTimer, countNumberString, minSet;

char hrCharSet[50], minCharSet[50], hrCharSetTimer[50], minCharSetTimer[50], key;

int countNumberHour, countNumberMinute, countNumberHourTimer, countNumberMinuteTimer, errorFlag, sec;

bool flagTimer, timerOn, flagExit;

void arrangeClock(void), verifyClock(void), arrangeClockTime(void), goToArrangeClock(void), goToArrangeClockTime(void), goToTimerShow(void);

char hexaKeys[4][4] = {

{ '1', '2', '3', 'A' },

{ '4', '5', '6', 'B' },

{ '7', '8', '9', 'C' },

{ '\*', '0', '#', 'D' }

};

byte rowPins[4] = {62, 63, 64, 65}; // Port K

byte colPins[4] = {66, 67, 68, 69}; // Port K

Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, 4, 4);

void setup()

{

pinMode(buzzerPin, OUTPUT);

noInterrupts();

TCCR1A = 0;

TCCR1B = 0;

OCR1A = 15624;

TCCR1B |= (1 << WGM12);

TCCR1B |= (1 << CS10);

TCCR1B |= (1 << CS12);

TIMSK1 |= (1 << OCIE1A);

interrupts();

lcd.begin(16, 2);

Serial.begin(9600);

arrangeClockTime();

}

void loop()

{

lcd.setCursor(2, 0);

lcd.print("Current time:");

lcd.setCursor(4, 1);

lcd.print(countNumberHour);

lcd.setCursor(6, 1);

lcd.print(":");

lcd.setCursor(7, 1);

lcd.print(countNumberMinute);

lcd.setCursor(9, 1);

lcd.print(":");

lcd.setCursor(10, 1);

lcd.print(sec);

key = customKeypad.getKey();

goToArrangeClock();

goToArrangeClockTime();

// ------------ //

switch(key){

case('C'):

lcd.clear();

key = customKeypad.getKey();

while (key != '\*' ||key != '#') {

key = customKeypad.getKey();

switch(key){

case('#'):

digitalWrite(buzzerPin, 1);

delay(100);

digitalWrite(buzzerPin, 0);

return;

case('D'):

lcd.clear();

timerOn = !timerOn;

case('\*'):

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

return;

default:

if (timerOn == 0) {

lcd.setCursor(0, 0);

lcd.print("Alarm:");

lcd.setCursor(0, 1);

lcd.print("OFF");

}

if (timerOn == 1) {

lcd.setCursor(0, 0);

lcd.print("Alarm:");

lcd.setCursor(0, 1);

lcd.print("ON");

}

lcd.setCursor(9, 1);

lcd.print(countNumberHourTimer);

lcd.setCursor(11, 1);

lcd.print(":");

lcd.setCursor(12, 1);

lcd.print(countNumberMinuteTimer);

}

}

default:

lcd.clear();

delay(500);

}

if (timerOn == 1) {

while (countNumberHour != countNumberHourTimer && countNumberMinute != countNumberMinuteTimer) {

break;

}

lcd.clear();

lcd.print("!ALARM! !ALARM!");

lcd.setCursor(0, 1);

lcd.print("!ALARM! !ALARM!");

digitalWrite(buzzerPin, 1);

delay(1000);

digitalWrite(buzzerPin, 0);

}

key = customKeypad.getKey();

delay(100);

lcd.clear();

}

// дана функція є процедурою обробки переривання за збігом таймера Т1

ISR(TIMER1\_COMPA\_vect){

if (sec == 59) {

sec = 0;

countNumberMinute += 1;

}

if (countNumberMinute == 60) {

countNumberHour += 1;

countNumberMinute = 0;

}

if (countNumberHour == 24) {

countNumberHour = 0;

countNumberMinute = 0;

sec = 0;

}

sec += 1;

}

// --------------- goToTimerShow ------------

void goToTimerShow()

{

lcd.clear();

flagExit = !flagExit;

while (flagExit == 1) {

key = customKeypad.getKey();

if (key == '\*') {

flagExit = !flagExit;

loop();

}

delay(100);

if (timerOn == 0) {

lcd.setCursor(0, 0);

lcd.print("Alarm:");

lcd.setCursor(0, 1);

lcd.print("OFF");

}

if (timerOn == 1) {

lcd.setCursor(0, 0);

lcd.print("Alarm:");

lcd.setCursor(0, 1);

lcd.print("ON");

}

lcd.setCursor(0, 1);

lcd.print(countNumberHourTimer);

lcd.setCursor(2, 1);

lcd.print(":");

lcd.setCursor(1, 1);

lcd.print(countNumberMinuteTimer);

}

}

// --------------- arrangeClockTime ------------

void arrangeClockTime()

{

lcd.clear();

lcd.print("Set the clock:");

while (countNumberString.length() != 5) {

key = customKeypad.getKey();

if(key == '\*'){

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

loop();

}

if (countNumberString.length() < 3) {

hrSet += String(key);

if (key) {

digitalWrite(buzzerPin, 1);

delay(50);

digitalWrite(buzzerPin, 0);

}

lcd.setCursor(0, 1);

lcd.print(hrSet);

lcd.setCursor(2, 1);

lcd.print(":");

}

if (countNumberString.length() >= 3 && countNumberString.length() < 5) {

minSet += String(key);

if (key) {

digitalWrite(buzzerPin, 1);

delay(50);

digitalWrite(buzzerPin, 0);

}

if (key == '\*') {

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

loop();

}

lcd.setCursor(3, 1);

lcd.print(minSet);

}

Serial.print(String(key));

Serial.println(countNumberString.length());

countNumberString = hrSet + ":" + minSet;

}

lcd.clear();

while (key != '#') {

key = customKeypad.getKey();

if(key == '#'){

lcd.clear();

lcd.print("Clock seted");

lcd.setCursor(0, 1);

hrSet.toCharArray(hrCharSet, 50);

countNumberHour = atoi(hrCharSet);

lcd.print(countNumberHour);

lcd.setCursor(2, 1);

lcd.print(":");

lcd.setCursor(3, 1);

minSet.toCharArray(minCharSet, 50);

countNumberMinute = atoi(minCharSet);

if (countNumberHour > 24 || countNumberMinute > 59) {

hrSet = "";

sec = 0;

minSet = "";

countNumberHour = 0;

countNumberMinute = 0;

countNumberString = "";

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Invalid time");

delay(1500);

arrangeClockTime();

}

lcd.print(countNumberMinute);

lcd.clear();

digitalWrite(buzzerPin, 1);

delay(100);

digitalWrite(buzzerPin, 0);

delay(50);

digitalWrite(buzzerPin, 1);

delay(100);

digitalWrite(buzzerPin, 0);

return;

}

if(key == '\*'){

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

return;

}

lcd.setCursor(0, 0);

lcd.print("Confirm with #");

lcd.setCursor(0, 1);

lcd.print(countNumberString);

}

}

// --------------- verifyClock ------------

void verifyClock()

{

lcd.clear();

lcd.print("Clock setted");

lcd.setCursor(0, 1);

hrSetTimer.toCharArray(hrCharSetTimer, 50);

countNumberHourTimer = atoi(hrCharSetTimer);

lcd.print(countNumberHourTimer);

lcd.setCursor(2, 1);

lcd.print(":");

lcd.setCursor(3, 1);

minSetTimer.toCharArray(minCharSetTimer, 50);

countNumberMinuteTimer = atoi(minCharSetTimer);

if (countNumberHourTimer > 24 || countNumberMinuteTimer > 59) {

hrSetTimer = "";

minSetTimer = "";

countNumberHourTimer = 0;

countNumberMinuteTimer = 0;

countNumberStringTimer = "";

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Invalid time");

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

errorFlag = !errorFlag;

delay(1500);

arrangeClockTime();

}

lcd.print(countNumberMinuteTimer);

delay(500);

}

// --------------- goToArrangeClockTime ------------

void goToArrangeClockTime(){

switch(key){

case('A'):

lcd.clear();

key = customKeypad.getKey();

if (key) {

digitalWrite(buzzerPin, 1);

delay(50);

digitalWrite(buzzerPin, 0);

}

while (key != '\*' ||key != '#') {

if(key == '\*'){

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

return;

}

hrSet = "";

sec = 0;

minSet = "";

countNumberHour = 0;

countNumberMinute = 0;

countNumberString = "";

arrangeClockTime();

}

}

}

// --------------- goToArrangeClock ------------

void goToArrangeClock(){

Serial.print(String(key));

switch (key) {

case('B'):

lcd.clear();

key = customKeypad.getKey();

Serial.print(String(key));

if (key) {

digitalWrite(buzzerPin, 1);

delay(50);

digitalWrite(buzzerPin, 0);

}

hrSetTimer = "";

minSetTimer = "";

countNumberHourTimer = 0;

countNumberMinuteTimer = 0;

countNumberStringTimer = "";

Serial.print(String(key));

arrangeClock();

Serial.print(String(key));

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Verify with #");

lcd.setCursor(0, 1);

lcd.print(countNumberStringTimer);

while (key != '#' ||key != '\*') {

key = customKeypad.getKey();

if(key == '#'){

digitalWrite(buzzerPin, 1);

delay(100);

digitalWrite(buzzerPin, 0);

delay(50);

digitalWrite(buzzerPin, 1);

delay(100);

digitalWrite(buzzerPin, 0);

verifyClock();

return;

}

if(key == '\*'){

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

return;

}

Serial.print(String(key));

}

}

}

// --------------- arrangeClock ------------

void arrangeClock()

{

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Clock setup:");

Serial.begin(9600);

lcd.clear();

lcd.print("Set the alarm:");

while (countNumberStringTimer.length() != 5) {

key = customKeypad.getKey();

if (countNumberStringTimer.length() < 3) {

if (key == '\*') {

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

hrSetTimer = "";

countNumberStringTimer = "";

arrangeClock();

}

hrSetTimer += String(key);

if (key) {

digitalWrite(buzzerPin, 1);

delay(50);

digitalWrite(buzzerPin, 0);

}

lcd.setCursor(0, 1);

lcd.print(hrSetTimer);

lcd.setCursor(2, 1);

lcd.print(":");

}

if (countNumberStringTimer.length() >= 3 && countNumberStringTimer.length() < 5) {

if (key == '\*') {

digitalWrite(buzzerPin, 1);

delay(1500);

digitalWrite(buzzerPin, 0);

countNumberStringTimer = "";

minSetTimer = "";

arrangeClock();

}

minSetTimer += String(key);

if (key) {

digitalWrite(buzzerPin, 1); // Запустили звучание

delay(50);

digitalWrite(buzzerPin, 0);

}

lcd.setCursor(3, 1);

lcd.print(minSetTimer);

}

Serial.print(String(key));

Serial.println(countNumberStringTimer.length());

countNumberStringTimer = hrSetTimer + ":" + minSetTimer;

}

lcd.clear();

}

## Висновок

На цій лабораторній роботі я дослідив та навчився працювати з таймерами, LCD-дисплеєм на базі HD44780 та клавіатурою 4x4. Також, для реалізації будильника, я під’єднав Buzzer та дослідив принцип його роботи.   
 Для виконання роботи, згідно мого варіанту я використав 8-ми розрядний інтерфейс LCD, який подів на порт-L. Клавіатуру ж я під’єднав до порта-К. Використовуючи таймер відліку часу Т1 я реалізував алгоритм 1, тобто годинник, на якому можна встановити поточний час та будильник, у будь-який момент змінювати значення часу та будильника, па переводити будильник в стан ON/OFF. Значення часу вводяться з клавіатури, а потім валідуються. Натиск кожної кнопки супроводжується коротким одинарним сигналом, успішне підтвердження введення даних за допомогою клавіші # супроводжується коротким подвійним сигналом, введення невірних даних за допомогою клавіші # та відміна усієї операції за допомогою клавіші \* супроводжується довгим сигналом.