МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ БОГДАНА ХМЕЛЬНИЦЬКОГО

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ФІЗИКИ, МАТЕМАТИКИ
ТА КОМП'ЮТЕРНО-ІНФОРМАЦІЙНИХ СИСТЕМ

КАФЕДРА АВТОМАТИЗАЦІЇ ТА КОМП'ЮТЕРНО-ІНТЕГРОВАНИХ ТЕХНОЛОГІЙ

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт з дисципліни

"ЦИФРОВА СХЕМОТЕХНІКА"

для студентів напрямку підготовки 6.050202

" Автоматизація та комп'ютерно-інтегровані технології "

всіх форм навчання

ЧЕРКАСИ – 2013

ББК 32.973-04
УДК 004.312

Методичні вказівки до виконання лабораторних робіт з курсу «Цифрова схемотехніка» для студентів напрямку підготовки 6.050202 “Автоматизація та комп’ютерно-інтегровані технології ”
Укл. В.А.Дідук – Черкаси, ЧНУ ім. Б.Хмельницького, 2013 –42с.

Навчальне видання

Методичні вказівки до виконання лабораторних робіт з курсу «Цифрова схемотехніка»
для студентів напрямку підготовки:
6.050202 “ Автоматизація та комп’ютерно-інтегровані технології ”

Укладач: Дідук Віталій Андрійович, к.т.н.

Рецензенти: Мусієнко Максим Павлович, д.т.н., професор
Осауленко Ігор Анатолійович, к.т.н., доцент

Затверджено на засіданні кафедри «Автоматизації та комп’ютерно-інтегрованих технологій» (протокол №8 від 26.03.2013р.)

Затверджено на засіданні вченої ради Черкаського національного університету імені Богдана Хмельницького (протокол №4 від 29.04.2013р.)

ЗМІСТ

ВСТУП

Основною метою лабораторних робіт з курсу “Цифрова схемотехніка” є вивчення та дослідження студентами засобів мікропоцесорної техніки, алгоритмів програмування, принципів синтезу пристроїв, що реалізують алгоритми керування, обробку та відображення вимірювальної інформації.

В ході лабораторних робіт досліджуються принципи мікропрограмного керування дискретними пристроями, п’єзовипромінювачами, рідкокристалічними індикаторами та цифровими датчиками.

Виконання лабораторних робіт базується на знаннях, отриманих студентами з основ теорії інформаційних систем, операційних середовищ, систем і оболонок, програмного забезпечення інформаційних систем, електроніки та мікросхемотехніки вимірювальних приладів, інформатики.

Загальний порядок виконання лабораторних робіт.

1. У процесі підготовки лабораторної роботи необхідно ознайомитися з її описом за даними методичними вказівками, вивчити теоретичні питання за додатково рекомендованими джерелами, вияснити мету і завдання досліджень.
2. Проведення вимірювань та досліджень на зібраній схемі установки починається тільки з дозволу викладача.
3. Під час проведення експериментів не дозволяється залишати робоче місце, збирати та розбирати електричні схеми під напругою, розміщувати на робочому місці сторонні предмети.
4. Під час проведення експериментів категорично забороняється торкатись руками чи іншими предметами елементів схеми, що знаходяться під напругою.
5. Забороняється подавати напругу на електричні схеми, які не використовуються на даному етапі роботи.
6. По закінченні експериментів необхідно вимкнути лабораторну установку, розібрати схему дослідження, прибрати робоче місце, здати інструменти, прилади, літературу.
7. Кожен студент повинен скласти звіт про лабораторну роботу і захистити його до виконання наступної роботи.

Лабораторна робота № 1
СИСТЕМИ ЧИСЛЕННЯ

Мета роботи: Навчитись представляти числа в двійковій, вісімковій, шістнадцятковій та двійково-десятковій системах числення; навчитись переводити числа із однієї системи числення в іншу.

Завдання до лабораторної роботи

Доповнити в таблиці 1 рядки з десятковими номерами від 17 до 31 відповідними значеннями в двійковій, вісімковій, шістнадцятковій і двійково-десятковій системах числення.

Таблиця 1

Представлення чисел в різних системах числення

Системи числення				
Десяткова	Двійкова	Вісімкова	Шістнад-цяткова	Двійково-десяткова
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101
16	10000	20	10	0001 0110
17	10001			
32	100000	40	20	0011 0010

1.2 Числа, які наведені в таблиці 2 в заданій системі числення, кожне представити в різних системах числення (десятковій, двійковій, вісімковій, шістнадцятковій і двійково-десятковій). Вибрати завдання із таблиці 2 у відповідності з порядковим номером в журналі групи. Результати представити у вигляді, вказаному в таблиці 3.

Таблиця 2

Індивідуальні завдання

Варіант	Системи числення				
	Десяткова	Двійкова	Вісімкова	Шістнад-цяткова	Двійково-десяткова
1	234	1110111	234	A34	
2	432	10010111	432	B32	
3	567	11001011	567	C67	
4	543	10110011	543	543	
5	786	10101011	756	7FF	
6	469	1111001	463	46A	
7	897	10001110	736	A197	
8	438	11011010	437	4A38	
9	291	10010010	261	2B11	
10	658	11101110	657	6E8	
11	386	10000111	376	2B86	
12	987	10111001	765	3E87	
13	876	11100010	654	A76	
14	564	01101100	564	B64	
15	368	10010011	366	3A8	
16	269	11001110	262	2C9	
17	805	11100011	705	80B	
18	675	11000110	675	2F5	
19	931	10011001	731	9E1	
20	873	11011011	773	87A	
21	764	11001001	764	764	
22	231	10010111	207	F31	
23	345	11011101	345	34B	
24	456	10101111	456	4F6	
25	678	11101110	671	678	
26	765	10111011	765	1A65	
27	891	11111101	561	A91	
28	588	10000111	555	B88	
29	677	10011111	677	A177	
30	483	10011000	443	1B83	
31	1052	1011011011	635	B2B6C	
32	340	01011111	777	56CA	
33	712	1010101011	410	F12C57	
34	543	11100110	367	B4B9FF	
35	390	110111111	202	1245BB	
36	666	11001100	312	B5EE	
37	999	1011000001	516	808C16B	
38	397	1010111011	456	34B2	

39	612	1110001011	303	F6AC	
40	555	1111100001	365	23BC4	
41	914	0100101101	212	45F6	
42	825	0001111101	106	7CCB	
43	789	1111011111	321	3CCB	
44	777	1011001100	456	78BF	
45	490	1000010000	404	58AA	
46	521	1101011000	555	47BA	
47	980	110000010	821	ABC9	
48	311	111101100	156	6F32B	

Таблиця 3

Приклад представлення отриманих результатів

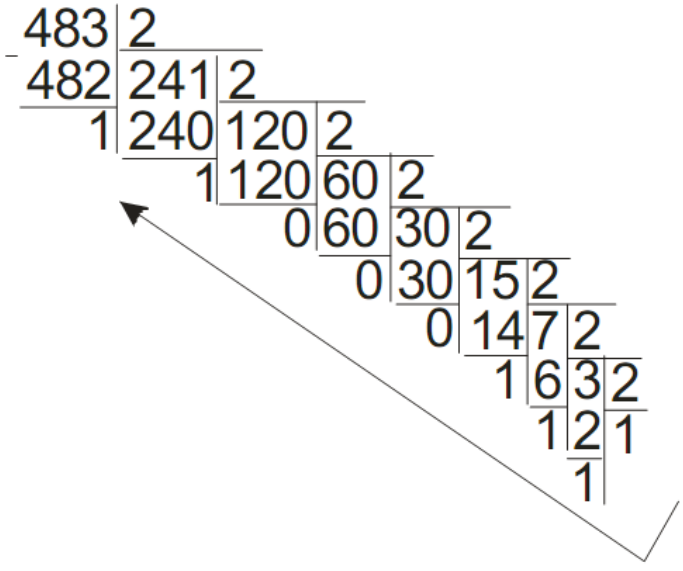
	Системи числення				
№ п/ч	Десяткова	Двійкова	Вісімкова	Шістнадцяткова	Двійково-десяткова
1	**483**	?	?	?	?
2	?	**10011000**	?	?	?
3	?	?	**443**	?	?
4	?	?	?	**1B83**	?

Приклад виконання завдання

Число 483 (в десятковій системі числення) перевести в двійкову систему числення.

$$483_{10} \rightarrow ?_2$$

Виконання.



Відповідь. $483_{10} \rightarrow 111100011_2$ – цей результат необхідно вписати в таблицю 3.

Перевірка.

Ваговий коефіцієнт	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Число	1	1	1	1	0	0	0	1	1

$1*256+1*128+1*64+1*32+1*2+1=483_{10}$

Число 483 (в десятковій системі числення) перевести в вісімкову систему числення.

$483_{10} \rightarrow ?_8$

Виконання. Розбиваємо двійкове число на тріади справа наліво і зображаємо кожну тріаду відповідною вісімковою цифрою (див. таблицю 1):

$483_{10} \rightarrow 111100011_2$
$111.100.011_2 \rightarrow 743_8$

Перевірка.

$743_8 \rightarrow 7*8^2+4*8^1+3*8^0=7*64+4*8+3=483_{10}$

Відповідь. $483_{10} \rightarrow 743_8$ – цей результат необхідно вписати в таблицю.

Число 483 (в десятковій системі числення) перевести в шістнадцяткову систему числення.

$483_{10} \rightarrow ?_{16}$

Виконання. Розбиваємо двійкове число на тетради (четвірки) справа наліво і кожну тетраду зображаємо відповідною шістнадцятковою цифрою (див. таблицю 1):

$483_{10} \rightarrow 111100011_2$
$0001.1110.0011 — 1E3_{16}$

Перевірка.

$1E3_{16} \rightarrow 1*16^2 + 14*16^1 + 3*16^0 = 256+ 224 +3 = 483_{10}$

Відповідь. $483_{10} \rightarrow 1E3_{16}$ – цей результат необхідно вписати в таблицю 3.

Число 483 (в десятковій системі числення) перевести в двійково-десяткову систему числення.

$483_{10} \rightarrow ?_{2\text{-}10}$

Виконання. Кожну десяткову цифру зображаємо відповідною двійковою тетрадою (див. таблицю 1):

4 8 3
0100 1000 0011

Відповідь. $483_{10} \rightarrow 10010000011_{2\text{-}10}$ – цей результат необхідно вписати в таблицю.

Хід роботи

Лабораторна робота виконується аркушах паперу формату А4 в рукописному вигляді.

1. Виконати роботу у відповідності з завданням.
2. Результатом виконання лабораторної роботи є представлення числових значень, записаних в різних системах числення з повним поясненням ходу отримання результатів.

Зміст звіту

1. Титульна сторінка та тема роботи.
2. Мета роботи.
3. Результати роботи.
4. Висновки.

Контрольні запитання

1. Що таке система числення?
2. Які типи систем числення ви знаєте?
3. Що таке основа позиційної системи числення?
4. У чому полягає проблема вибору системи числення для подання чисел у пам'яті комп'ютера, мікроконтролера?
5. Яка система числення використовується для подання чисел у пам'яті комп'ютера, мікроконтролера? Чому?
6. Яким чином здійснюється перевід чисел, якщо основа нової системи числення дорівнює деякому степеню старої системи числення?
7. За яким правилом переводяться числа з десяткової системи числення?
8. За яким правилом переводяться числа в десяткову систему числення?

Лабораторна робота № 2
ЗНАЙОМСТВО З ОБЛАДНАННЯМ ДЛЯ РОБОТИ З МІКРОКОНТРОЛЕРАМИ AVR. ПРОШИВКА МІКРОКОНТРОЛЕРА

Мета роботи: ознайомитись з базовими типами мікроконтролерів AVR, поняттями “прошивка” та “програматор”, автономними та внутрішньосхемними програматорами та відповідним програмним забезпеченням. Набути навиків прошивки управляючих програм в контролер.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; універсальний автономний програматор; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

Поняття про прошивку та програматор

Мікроконтролер (МК), як і комп'ютер, є апаратною частиною пристрою. Без управляючої програми він працювати не буде. Цю програму пишуть і компілюють на комп'ютері. Результат компіляції — HEX-файл, що містить двійковий (бінарний) код. Цей код записується у мікроконтролер і називається прошивкою.

Прошивка — це двійковий код, який виконується мікроконтролером і записаний (“зашитий”) у його постійну пам'ять програм (Flash-пам'ять).

Програматор — це пристрій для запису прошивки у мікроконтролер.

Програматор під'єднується до мікроконтролера і до комп'ютера. На комп'ютері працює програма, яка зчитує файл з прошивкою і передає її у програматор. Програматор, в свою чергу, записує прошивку у постійну пам'ять програм (Flash-пам'ять) мікроконтролера.

Два принципи програмування МК

- Автономний програматор має панельку, куди вставляється мікроконтролер. МК прошивають, потім виймають з панельки і вставляють у пристрій, в якому він буде працювати.
- Внутрішньосхемний програматор програмує МК прямо в схемі. Він підключається до спеціальних виводів МК. Після прошивання програматор відключається і схема одразу починає працювати. Внутрішньосхемний програматор значно зручніший для відладки програм, оскільки не треба переставляти МК зі схеми у програматор і навпаки.

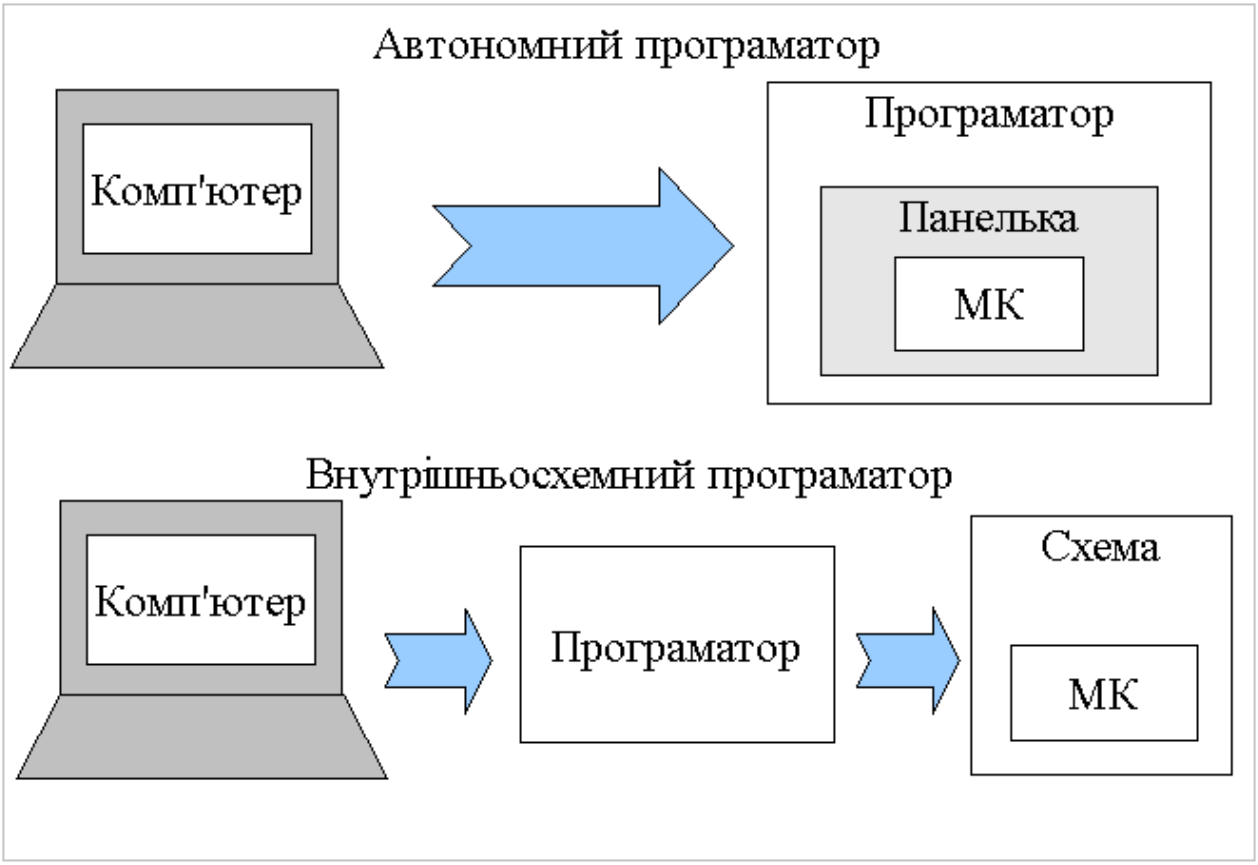


Рисунок 1 – Схеми запису прошивки в пам’ять мікроконтролера

Програми — програматори

Існує багато різних програм-програматорів для різних операційних систем. В лабораторних роботах використовуються деякі з них.

PonyProg (www.lancos.com). Програма підтримує велику кількість мікроконтролерів AVR, а також інші мікросхеми. Працює з програматорами, що підключається до портів COM та LPT, має графічний інтерфейс.

Avrdude (http://savannah.gnu.org/projects/avrdude/). Входить до пакету WinAVR, тож її не потрібно окремо скачувати і встановлювати на комп'ютер. Підтримує COM, LPT, а також USB-програматори. Працює в операційних системах Windows, Linux, MacOS та ін. Не має графічного інтерфейсу (GUI), проте існують графічні оболонки, наприклад, avrdude-gui, AVR8 Burn-O-Mat (http://www.brischalle.de/avr8_burn-o-mat_avrdude_gui/avr8_burn_o_mat_avrdude_gui_en.html).

Обладнання для лабораторних робіт

Навчальна плата EV8031/AVR працює з мікроконтролером ATmega8515. Крім мікроконтролера, на платі встановлені кнопки, світлодіоди, світлодіодні індикатори, рідкокристалічний індикатор (LCD), COM-порт та інші радіокомпоненти.

Щоб прошити МК, використовується найпростіший програматор. Він підключається до LPT порту та розніму на навчальній платі. Отже, він є внутрішньосхемним програматором. Всього 4 резистори знаходяться у рознімі LPT. Для прошивки цим програматором використовується комп'ютерна

програма PonyProg. Після встановлення програми на комп'ютер необхідно вибрати Setup — Calibration — Yes — OK. Для роботи PonyProg з цим програматором потрібно в меню Setup-Interface setup встановити налаштування відповідно до рис. 2.

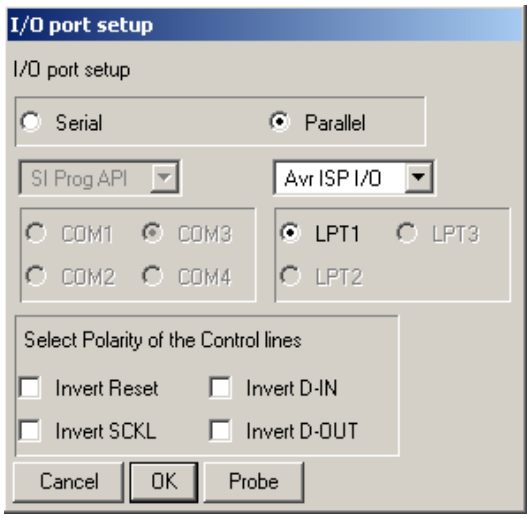


Рисунок 2 – Налаштування PonyProg для найпростішого програматора

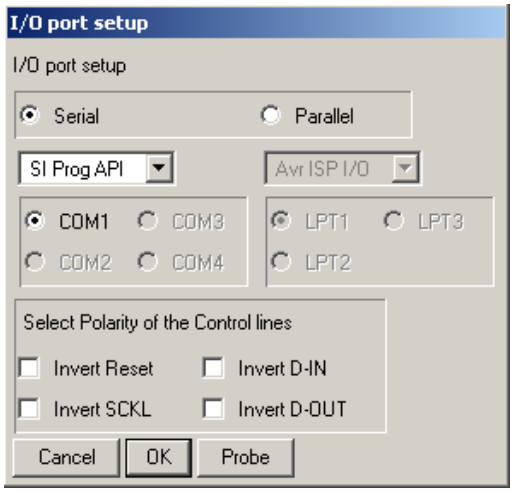


Рисунок 3 – Налаштування PonyProg для універсального автономного програматора

Універсальний автономний програматор підключається до COM-порту. Він має панельки для різних моделей мікроконтролерів AVR. Управляюча програма — PonyProg, налаштування на рис 3.

Хід роботи

1. Ознайомитись з апаратними пристроями на навчальній платі. Знайти на ній мікроконтролер, рознім для програмування та лінійку світлодіодів.
2. Відкрити програму PonyProg та вибрати мікроконтролер: Device — AVRmicro — ATmega8515. Вказати послідовність дій при програмуванні: Command — Program options, встановити прапорці лише біля пунктів: Reload

files (перезавантажити файли), Erase (стерти МК), Write Program memory (FLASH) (запрограмувати Flash-пам'ять), клацнути OK.

3. Налаштувати PonyProg для LPT-програматора.

4. Під'єднати LPT-програматор до плати і до комп'ютера. Увімкнути плату, під'єднавши кабель живлення.

5. Відкрити файл main.hex із першою тестовою програмою з папки lab1/test1, скориставшись командою File — Open Program (FLASH) file.

6. Запрограмувати МК. Для цього вибрати Command — Program. Перевірити відсутність помилок (повинно з'явитися повідомлення “Programming successful”).

7. Одразу після програмування на світлодіодних індикаторах повинно з'явитись число 4567, а лінійка світлодіодів повинна блимати. (Іноді потрібно від'єднати програматор від комп'ютера.)

8. Від'єднати програматор від комп'ютера та плати. Вимкнути плату.

9. Взяти автономний програматор, знайти в ньому панельку для ATmega8515. Якщо у будь-якій панельці програматора присутній інший МК, вийняти його. Обережно, щоб не пошкодити виводи, вийняти ATmega8515 з навчальної плати та вставити його у програматор. Під'єднати програматор до комп'ютера.

10. Налаштувати PonyProg для даного типу програматора.

11. Відкрити файл main.hex із другою тестовою програмою з папки lab1/test2

12. Запрограмувати МК, впевнитись у відсутності помилок.

13. Обережно вийняти МК з програматора та вставити його у навчальну плату.

14. Увімкнути плату, перевірити виконання програми. Повинен з'явитися “вогонь, що біжить” - послідовне періодичне включення світлодіодів по одному, з першого до останнього.

15. Вимкнути навчальну плату, закрити PonyProg.

Зміст звіту

1. Тема та мета роботи.

2. Перелік використаного обладнання.

3. Стислий зміст теоретичних відомостей. Короткий опис програматорів та програмного забезпечення, що використовувались в роботі.

4. Відповіді на контрольні запитання.

5. Висновок.

Лабораторна робота № 3
УПРАВЛІННЯ СВІТЛОДІОДАМИ

Мета роботи: ознайомитись з принципом управління дискретними пристроями за допомогою зовнішньої пам'яті, а також програмним забезпеченням WinAVR. Набути навиків створення та компіляції власної програми для мікроконтролера.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; середовище програмування WinAVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

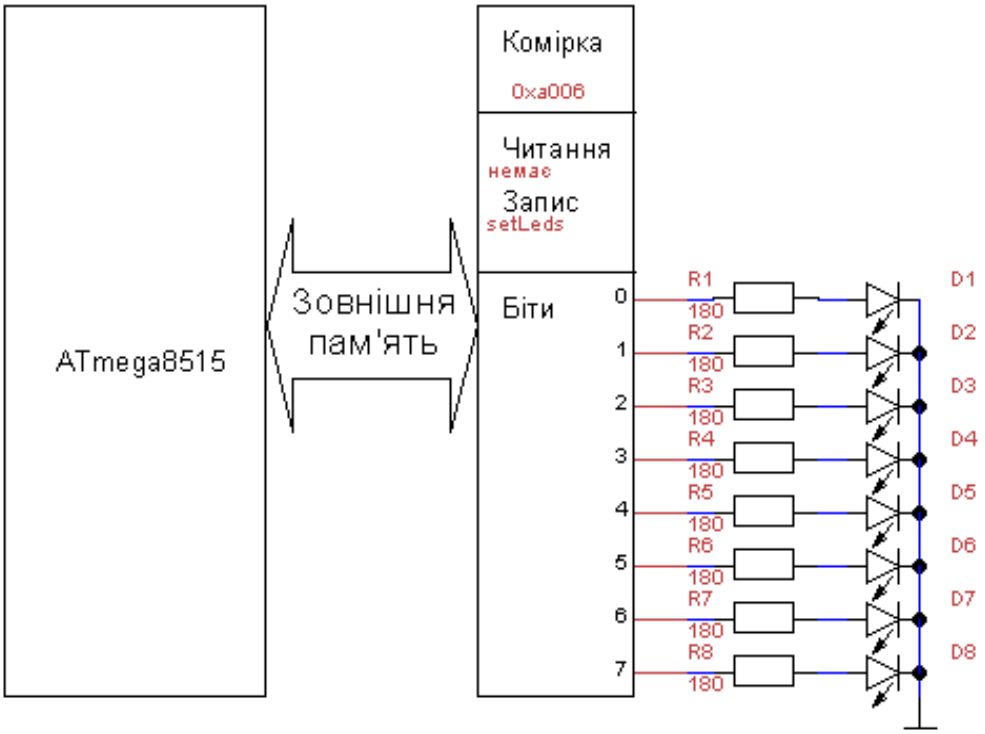


Рисунок 4 – Структурна схема ввімкнення світло діодів на навчальній платі Open System EV8031/AVR

Навчальна плата має лінійку світлодіодів (8 світлодіодів, запаяних в рядок). Вона під'єднана до мікроконтролера через комірку зовнішньої оперативної пам'яті. Щоб засвітити світлодіоди деяким чином, потрібно записати значення в цю комірку. Запис відбувається за допомогою функції setLeds(byte data) з файлу ev8031.c. Прочитати значення даної комірки неможливо, тож функції для читання немає.

Як випливає зі схеми, перший світлодіод під'єднаний до біту 0, другий – до біту 1 і т.д. Також зі схеми видно, що для ввімкнення світлодіоду потрібно записати “1” у відповідний біт, а для вимкнення – “0”. Отже, якщо значення біту – “0”, то світлодіод не горить, а якщо “1”, то горить. Щоб загорілися всі світлодіоди, необхідно встановити всі біти в “1”.

WinAVR

WinAVR – це пакет програм для Windows для розробки програм для мікроконтролерів AVR. Це проект з відкритим вихідним кодом (Open Source), він безкоштовний. Домашня сторінка http://sourceforge.net/projects/winavr/ . До складу WinAVR входить версія Сі-компілятора GCC, спеціально призначена для платформи AVR, бібліотека Сі для AVR під назвою avr-libc, програматор Avrdude, текстовий редактор Programmers Notepad, утиліта для збірки проекту make та інші програми. У майже всіх програм з пакету WinAVR існують версії для Linux, тож розробляти проекти на AVR можна у різних операційних системах.

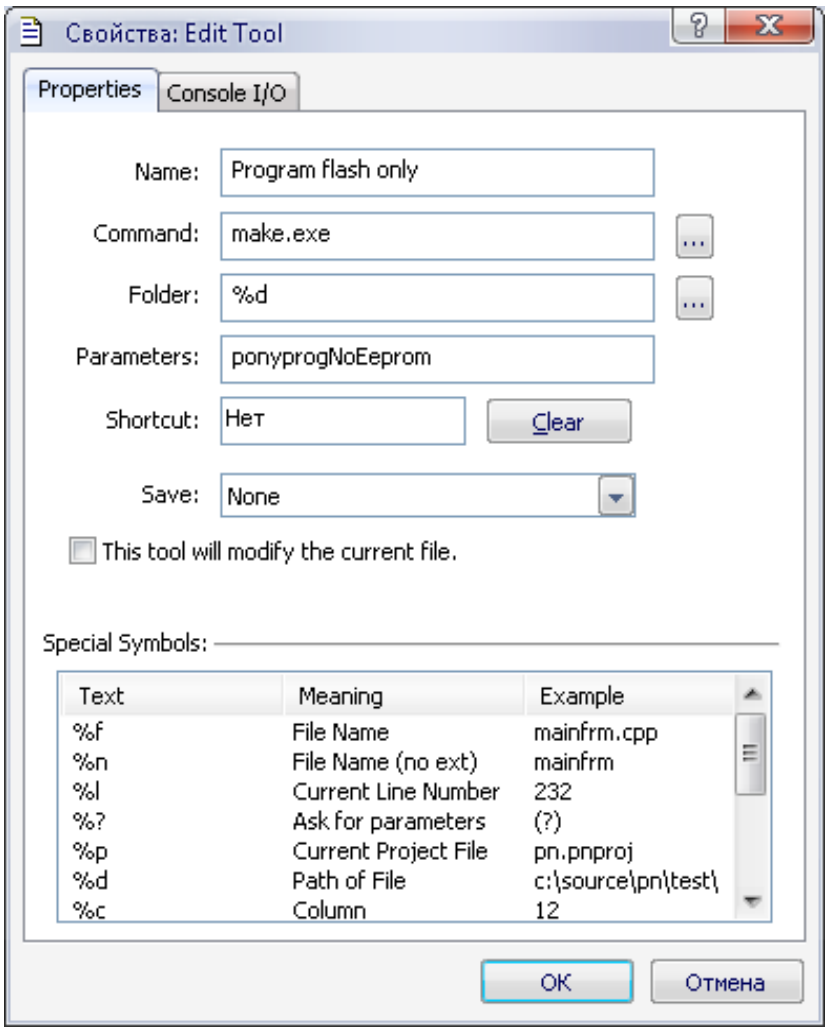


Рисунок 5 – Вікно розділу «Tools» з пункту меню «Tools — Options» з встановленими налаштуваннями для сумісної роботи WinAVR та PonyProg

Після встановлення WinAVR у головному меню Windows з'являється відповідний розділ. Найважливіші пункти – Programmers Notepad (запускає текстовий редактор) та avr-libc Manual (відкриває документацію бібліотеки Сі avr-libc). У розділі Library Reference знаходиться опис всіх доступних функцій та модулів, в яких вони знаходяться.

Щоб налаштувати Programmers Notepad для сумісної роботи з PonyProg, необхідно вибрати пункт меню Tools — Options, розділ Tools, натиснути кнопку Add та здійснити налаштування згідно з рис. 5.

У даних лабораторних роботах для розробки програм будемо застосовувати WinAVR.

Приклад програми

Наступна програма забезпечує таку функцію: усі світлодіоди блимають з частотою близько 1 Гц. Вона знаходиться у файлі lab2/test1/main.c

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
int main(void)
{ ENABLE_EXTERNAL_RAM();
for(;;)
{ setLeds(0xFF);
for(byte d = 50; d>0; d--) _delay_ms(10); //delay 500 ms
setLeds(0x00);
for(byte d = 50; d>0; d--) _delay_ms(10); //delay 500 ms
}
return 0;
}
```

У бібліотеці avr/io.h знаходяться деякі функції вводу-виводу та описи регістрів МК. Без цієї бібліотеки не обходиться жодна програма для AVR. У бібліотеці util/delay.h розміщені функції затримки (паузи).

У файлі ev8031.c містяться функції, специфічні для навчальної плати EV8031, а також описи типів даних: byte (байт, число 0—255) та boolean (true або false). Символи “..” означають папку на один рівень вище (програма знаходиться у файлі lab2/test1/main.c).

Головна функція – main. Мікроконтролер виконує її після скидання або ввімкнення живлення. Вона закінчується нескінченним циклом та поверненням нуля.

Оскільки практично всі периферійні пристрої під'єднані через зовнішню пам'ять, на початку програми потрібно ввімкнути інтерфейс зовнішньої пам'яті. Це робить функція ENABLE_EXTERNAL_RAM**()** з файлу ev8031.c.

Далі йде нескінченний цикл for(;;). На відміну від комп'ютерної програми, програма мікроконтролера працює у нескінченному циклі до вимкнення живлення або скидання МК.

Функція setLeds**(**0xFF**)** записує у комірку пам'яті, що керує лінійкою світлодіодів, число 0xFF (усі біти у стані “1”). Усі світлодіоди вмикаються. Пізніше, при записі числа 0x00, всі світлодіоди вимикаються.

Функція _delay_ms**(**10**)** з бібліотеки util/delay.h організує затримку 10 мілісекунд. Написати _delay_ms**(**500**)** не можна, бо максимальне значення

аргументу 262.14 мс, поділене на тактову частоту в МГц. Безпечне значення для усіх мікроконтролерів AVR та тактових частот є 10 мс. Щоб зробити затримку 0,5 с (500 мс), потрібно повторити 50 разів затримку 10 мс.

Цикл, у якому змінна зменшується, **for(**byte d = 50**;** d>0**;** d--**)** займає значно менше місця в пам'яті мікроконтролера та виконується значно швидше, ніж цикл, у якому змінна збільшується, **for(**byte d = 0**;** d<50**;** d++**)**, тому що порівняння з нулем виконати простіше, ніж з числом, відмінним від нуля.

Алгоритм роботи програми

Розглянемо, як працює наведена програма. На початку вмикається інтерфейс зовнішньої пам'яті. Потім МК входить у нескінченний цикл. Вмикаються всі світлодіоди, потім пауза 500 мс, вимикаються всі світлодіоди, знову пауза 500 мс. Далі цикл повторюється до нескінченності.

Блимання деяких світлодіодів

Можна зробити так, щоб не всі світлодіоди блимали. Наприклад, будемо вмикати лише другий, третій та сьомий світлодіоди. Для цього треба виставляти в “1” лише біти 1, 2 та 6 (згадаємо, що перший світлодіод під'єднаний до біту 0), інші біти – в “0”. У програму потрібно вписати деяке число у шістнадцятковій системі числення (HEX). Приставка “0x” означає, що число записане у шістнадцятковому вигляді. Щоб його отримати, запишемо значення всіх восьми бітів, починаючи з найстаршого.

Таблиця 4

Приклад кодування стану світлодіодів та їх запис в форматі HEX

Біти								**Число у HEX-форматі**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**	
0	1	0	0	0	1	1	0	0x46
1	1	0	1	1	0	0	1	0xD9
0	1	0	1	1	1	1	1	0x5F

Розділимо біти на групи по 4 і замінимо кожну групу однією цифрою згідно таблиці. Отримаємо потрібне число (табл. 5).

У наведеній вище програмі лінійка світлодіодів знаходиться у двох станах по черзі. У першому стані усі світлодіоди горять (0xFF), у другому усі погашені (0x00). Загалом, якщо в обох станах значення біту рівне “1”, відповідний світлодіод завжди горить, якщо “0”, то не горить, а якщо значення біту змінюється, то світлодіод блимає.

Таблиця 5

Таблиця відповідності двійкових та шістнадцяткових чисел

Двійкове число				Шістнадняткова цифра	Двійкове число				Шістнадняткова цифра
0	0	0	0	0	1	0	0	0	8
0	0	0	1	1	1	0	0	1	9
0	0	1	0	2	1	0	1	0	A
0	0	1	1	3	1	0	1	1	B
0	1	0	0	4	1	1	0	0	C
0	1	0	1	5	1	1	0	1	D
0	1	1	0	6	1	1	1	0	E
0	1	1	1	7	1	1	1	1	F

Створення нової програми

Щоб створити новий проект, необхідно створити нову папку, скопіювати туди файл Makefile з прикладу до лабораторної роботи 1, а також створити файл main.c, у якому написати текст програми. Перевірити в файлі Makefile значення таких величин:

```
MCU = atmega8515
F_CPU = 1000000
TARGET = main
...
PONYPROG = C:/Program Files/PonyProg2000/PonyProg2000.exe
ponyprog:
@echo -e "SELECTDEVICE $(MCU)\nLOAD-PROG $(TARGET).hex\nLOAD-DATA $(TARGET).eep\nERASE-ALL\nWRITE&VERIFY-ALL" >ponyprog.e2s
$(PONYPROG) ponyprog.e2s
ponyprogNoEeprom:
@echo -e "SELECTDEVICE $(MCU)\nLOAD-PROG $(TARGET).hex\nERASE-ALL\nWRITE&VERIFY-ALL" >ponyprog.e2s
$(PONYPROG) ponyprog.e2s
```

У цих рядках вказаний МК, тактова частота, ім'я файлу з програмою (без розширення), шлях до встановленого PonyProg та команди для його запуску.

Компіляція програми

Спочатку потрібно відкрити файл з текстом програми у текстовому редакторі Programmers Notepad. Щоб скомпілювати програму, виберіть Tools – Make all. Внизу з'явиться вікно “Output”. Треба дочекатися напису “Process exit

code: 0”, який повідомляє про успішне виконання. Якщо замість нуля з'явилась інша цифра, значить, у програмі є помилки. Вище у цьому самому вікні буде написано, які саме.

Якщо з'являється повідомлення про помилки, пов'язані з файлами у папці .dep, треба видалити цю папку, виконати команду Tools – Make clean та повторити компіляцію.

Знайомі з командним рядком можуть виконати команду “make all” у папці з файлом Makefile.

Хід роботи

1. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.
2. Скомпілювати тестову програму.
3. Створити свою програму згідно з індивідуальним завданням та скомпілювати її.
4. Під'єднати внутрішньосхемний програматор до плати та комп'ютера. Увімкнути плату.
5. Запрограмувати МК. При правильно налаштованих Programmers Notepad і PonyProg у програмі Programmers Notepad вибрати Tools – Program flash only. Відкриється PonyProg, запрограмує МК і після повідомлення про успішне або невдале завершення закриється.
6. Перевірити правильність виконання програми.

Індивідуальні завдання

Створити програму, яка виконує наступні функції: світлодіоди з вказаними номерами увімкнені, вимкнені або блимають згідно з таблицею.

Варіант	**Увімкнені**	**Вимкнені**	**Блимають**
1	1,2	3,4	5, 6, 7, 8
2	3,6	2,7	1, 4, 5, 8
3	3, 6, 8	2,5	1, 4, 7
4	2,5	1, 7, 8	3, 4, 6
5	2,3	4,5	1, 6, 7, 8
6	2, 3, 4	1,5	6, 7, 8
7	1, 5, 8	3,6	2, 4, 7
8	2,4	6,8	1, 3, 5, 7
9	1,7	3,5	2, 4, 6, 8
10	4,5	3,6	1, 2, 7, 8
11	1	2, 4, 6	3, 5, 7, 8
12	1,8	2,7	3, 4, 5, 6
13	3, 4, 8	6,7	1, 2, 5
14	6, 7, 8	3	1, 2, 4, 5
15	1, 2, 8	3, 6, 7	4, 5, 6

Зміст звіту

1. Тема та мета роботи.
2. Перелік використаного обладнання.
3. Стислий зміст теоретичних відомостей.
4. Лістинг власної програми з поясненням кожного рядка.
5. Відповіді на контрольні запитання.
6. Висновок.

Лабораторна робота № 4
СТВОРЕННЯ ЗАПРОГРАМОВАНИХ ПОСЛІДОВНОСТЕЙ ВИХІДНИХ СИГНАЛІВ

Мета роботи: ознайомитись з принципами утворення запрограмованих послідовностей вихідних сигналів на прикладі світлових ефектів. Використання операцій логічного зсуву. Набути навиків написання ускладнених програм.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; середовище програмування WinAVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

Вогонь, що біжить

Ефект вогню, що біжить полягає в послідовному засвіченні вогнів (ламп, світлодіодів) по одному, з першого до останнього. Після останнього знову вмикається перший вогонь і все повторюється. Спостерігачу здається, що вогонь “біжить”.

Ось програма, що створює ефект вогню, що біжить. У якості вогнів використовується лінійка світлодіодів. Файл програми lab3/test1/main.c

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/

#include <avr/io.h>
#include <util/delay.h>

#include "../../ev8031.c"

int main(void)
{ ENABLE_EXTERNAL_RAM();
byte i=0;
for(;;)
{ if(++i >= 8) i = 0;
setLeds(1 << i);
for(byte d = 10; d>0; d--) _delay_ms(10); //delay 100 ms
}
return 0;
}
```

Тут кожні 100 мс змінна і збільшується, набуваючи значень від 0 до 7, і виставляється “1” у біт і.

Ефекти зі зсувом значень

Суть цих ефектів полягає у зсуві значень бітів (“0” або “1”) вправо або вліво. Значення старшого біту зникає з картинки, а у молодший біт записується певне значення. Спостерігачу здається, що картинка рухається в одну сторону.

Біти	7	6	5	4	3	2	1	0
	■	■	■		■	■		■
■	■	■		■	■		■	
Зникле зна-чення	Зсунуті вліво значення							Нове зна-чення

Рисунок 6 – Приклад зсуву вліво

Нове значення молодшого біту може бути:

- таким самим, як і зникле значення старшого біту. Картинка буде весь час повторюватися. Можна отримати, наприклад, вогонь, що біжить, “2 вогні, що біжать” та тінь, що біжить (вимкнений лише один вогонь, який зсувається)
- протилежним до значення старшого біту. Якщо спочатку всі вогні були вимкнені, то ввімкнеться перший вогонь, до нього долучиться другий, потім третій, а коли всі ввімкнуться, погасне перший, потім другий, і так до кінця. Потім процес повториться.
- розрахованим з поточного значення декількох бітів. Воно буде не очевидним для спостерігача. Картинка буде повторюватися рідше, ніж у попередніх випадках.
- довільним. Вийде непередбачувана картинка.

Наведемо для прикладу програму, в якій нове значення молодшого біту буде протилежним то передостаннього (шостого) біту. На відміну від випадку, коли значення протилежне до останнього біту, тут не виникне ситуації, коли всі світлодіоди погашені та настає темрява.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
int main(void)
{ ENABLE_EXTERNAL_RAM();
byte state=0;
byte next;
for(;;)
{ next = !(state & 0x40);
state <<= 1;
```

```
if(next) state |= 1;
setLeds(state);
for(byte d = 25; d>0; d--) _delay_ms(10); //delay 250 ms
}
return 0;
}
```

Змінна state зберігає поточний стан світлодіодів, а змінна next дорівнює 0, якщо нове значення молодшого біту 0, та не 0, якщо нове значення молодшого біту 1.

Запрограмована послідовність

Можна також записати в пам'ять усю послідовність станів світлодіодів. Ця послідовність довільна, жоден стан не залежить від попереднього.

Дана програма здійснює той самий світловий ефект, що і попередня, проте уся послідовність станів занесена в масив. Байти стану попередньо розраховані та записані в шістнадцятковій формі. Ці байти по черзі виводяться на світлодіоди.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>

#include "../../ev8031.c"

byte states[] = { 1, 3, 7, 0x0F, 0x1F, 0x3F, 0x7F,
0xFE, 0xFC, 0xF8, 0xF0, 0xE0, 0xC0, 0x80 };

int main(void)
{ ENABLE_EXTERNAL_RAM();
byte i=0;
for(;;)
{ if(++i >= sizeof(states)) i=0;
setLeds(states[i]);
for(byte d = 25; d>0; d--) _delay_ms(10); //delay 250 ms
}
return 0;
}
```

Хід роботи

1. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.
2. Створити свою програму, яка здійснює світловий ефект, описаний у індивідуальному завданні. Скомпілювати свою програму.

3. Запрограмувати МК, перевірити правильність виконання.

Індивідуальні завдання

Зробити описаний світловий ефект. Ефект має повторюватися до нескінченності.

1. Вогонь, що біжить, пробігає 3 рази в одну сторону і 3 рази в іншу.
2. Перший і останній вогонь біжать до центру, назустріч один одному, 3 рази. Потім вони розбігаються від центру до країв, також 3 рази.
3. Два вогні, що біжать, пробігають 2 рази в одну сторону і 2 рази в іншу.
4. Перші 2 світлодіода блимають з частотою 2 Гц, наступні 2 — з частотою 1 Гц, ще 2 — з частотою 0,5 Гц і останні 2 — з частотою 0,25 Гц.
5. Вогні перший і п'ятий, що біжать, пробігають 2 рази в одну сторону і 2 рази в іншу.
6. Загоряється перший і останній світлодіод, потім другий і передостанній і так до центру. Потім всі світлодіоди погасають. Після цього світлодіоди загоряються з центру до країв по два, потім всі разом погасають.
7. Вогні перший, другий, п'ятий і шостий, що біжать, пробігають 2 рази в одну сторону і 2 рази в іншу.
8. Загоряється перший і п'ятий світлодіод, до них долучаються другий і шостий, потім третій і сьомий, далі четвертий і восьмий. Потім гаснуть послідовно з першого до останнього. Далі загоряються так само, а гаснуть з останнього до першого.
9. Перший і останній вогонь біжать до центру, назустріч один одному. Потім вони розбігаються від центру до країв.
10. Тінь, що біжить, пробігає 3 рази в одну сторону і 2 рази в іншу.
11. Перший і останній світлодіод блимають з частотою 0,25 Гц, другий і передостанній у 2 рази швидше (0,5 Гц) і так до середини частота блимання збільшується вдвічі.
12. Світлодіоди послідовно вмикаються, з першого до останнього, а потім погасають, також з першого до останнього. Потім вмикаються з останнього до першого та погасають з останнього до першого.
13. Світлодіоди послідовно вмикаються, з першого до останнього, а потім погасають у зворотньому напрямку, з останнього до першого.
14. Спочатку всі світлодіоди горять. Погасають перший і останній, потім другий і передостанній, і так до центру. Далі всі загоряються. Потім світлодіоди погасають з центру до країв, після чого всі загоряються.
15. Загоряється перший і другий світлодіод, до них долучаються сьомий і восьмий, потім третій і четвертий, далі п'ятий і шостий. Потім гаснуть послідовно з першого до останнього. Далі загоряються так само, а гаснуть з останнього до першого.

Зміст звіту

1. Тема та мета роботи.
2. Перелік використаного обладнання.

3. Стислий зміст теоретичних відомостей.
4. Лістинг власної програми з детальним поясненням кожного рядка.
5. Окремо виписати з файлу ev8031.c усі функції, що використовувались при написанні програми і дати їх детальне пояснення.
6. Відповіді на контрольні запитання.
7. Висновок.

Лабораторна робота № 5
КНОПКИ ТА СВІТЛОДІОДНІ ІНДИКАТОРИ

Мета роботи: ознайомитись з портами вводу-виводу мікроконтролера, принципом обробки сигналів дискретних датчиків. Набути навичок відображення інформації за допомогою світлодіодних індикаторів.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; середовище програмування WinAVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

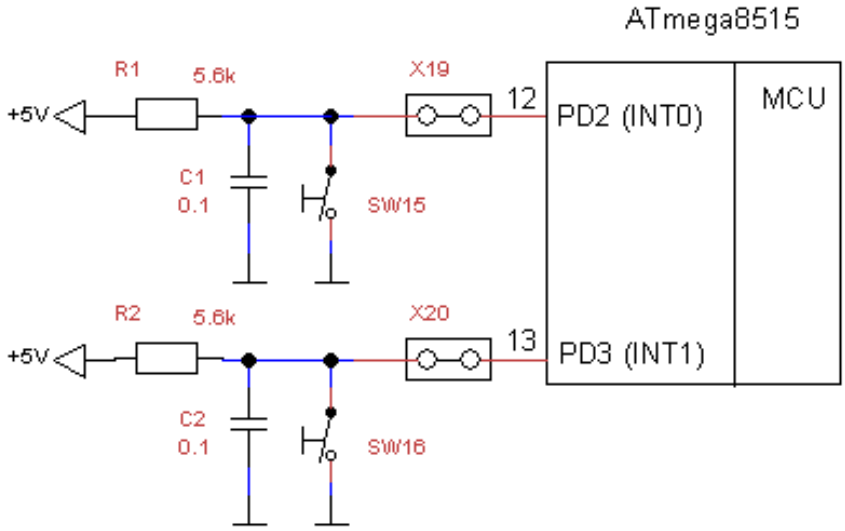


Рисунок 7 – Принципова схема під'єднання кнопок

На навчальній платі кнопки SW15 та SW16 під'єднані безпосередньо до виводів мікроконтролера. Як показано на схемі (рис. 7), один вивід кнопки під'єднаний до “землі” а інший – до лінії порта МК. Значить, коли кнопка не натиснута, на лінії порта логічна “одиниця”, а коли натиснута – логічний “нуль”. Конденсатор, що під'єднаний паралельно кнопці, призначений для усунення дребезгу контактів.

Для розуміння подальшого матеріалу необхідно прочитати розділ 12 книги [3] або крок 2 [1] та ступені 3, 4 [2].

Отже, у програмі для мікроконтролера потрібно встановити лінію порта, до якої під'єднана кнопка, на вхід зі внутрішнім резистором (DDRD &= ~_BV(PD2); PORTD |= _BV(PD2);). Для перевірки, чи натиснута кнопка, потрібно користуватися функцією bit_is_clear(PIND, PD2), а щоб перевірити, чи кнопка не натиснута, користуються функцією bit_is_set(PIND, PD2).

Управління світлодіодами за допомогою кнопок

Якщо натиснута перша кнопка, горить перша половина лінійки світлодіодів, а якщо друга – то друга половина. Кожна кнопка вмикає свою половину світлодіодів незалежно одна від одної.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
int main(void)
{ ENABLE_EXTERNAL_RAM();
DDRB = DDRD = 0x00;
PORTB = PORTD = 0xFF;
for(;;)
{ byte leds = 0;
if(bit_is_clear(PIND, PD2)) leds |= 0x0F;
if(bit_is_clear(PIND, PD3)) leds |= 0xF0;
setLeds(leds);
for(byte d = 10; d>0; d--) _delay_ms(10);
//delay 100 ms
}
return 0;
}
```

На початку програми усі лінії портів B та D встановлюються як входи зі внутрішніми резисторами.

Світлодіодні індикатори на навчальній платі

Ті світлодіодні індикатори на навчальній платі, які знаходяться біля мікроконтролера, під'єднані за спеціальною схемою для зручності користування. У інших схемах таке під'єднання не застосовується, тож ці індикатори призначені для того, щоб швидко і просто показати число.

Функції управління світлодіодними індикаторами:

- setLedIndicatorLeft(byte x) – виводить число x на два ліві розряди індикатора. Наприклад, виклик setLedIndicatorLeft(0x23) покаже число 23, setLedIndicatorLeft(0x92) – число 92.
- setLedIndicatorRight(byte x)-- аналогічно для двох правих розрядів
- setLedIndicatorEnabledAndDots(byte x) – вмикає/вимикає індикатори та десяткові крапки. 4 старших біта (0xF0) відповідають десятковим крапкам (1 – увімкнено, 0 - вимкнено), а 4 молодших – вмиканню розрядів індикатора (0 – увімкнено, 1 - вимкнено).

Управління індикаторами за допомогою кнопок

Якщо натиснута перша кнопка, на лівій частині індикатора горить число 99, інакше – 11. Якщо натиснута друга – то 87, інакше – 23. Кожна кнопка управляє своєю половиною індикатора незалежно одна від одної. Завжди горять перша і четверта десяткові крапки.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/

#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
int main(void)
{ ENABLE_EXTERNAL_RAM();
DDRB = DDRD = 0x00;
PORTB = PORTD = 0xFF;
setLedIndicatorEnabledAndDots(0x90);
byte value;
for(;;)
{ value = bit_is_clear(PIND, PD2) ? 0x99 : 0x11;
setLedIndicatorLeft(value);
value = bit_is_clear(PIND, PD3) ? 0x87 : 0x23;
setLedIndicatorRight(value);
for(byte d = 10; d>0; d--) _delay_ms(10);
//delay 100 ms
}
return 0;
}
```

Хід роботи

1. Знайти на навчальній платі потрібні кнопки та світлодіодні індикатори.
2. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.
3. Створити свою власну програму, яка забезпечує функції, описані в індивідуальному завданні, та скомпілювати її.
4. Запрограмувати МК, перевірити правильність виконання.

Індивідуальні завдання

1. На індикаторі світиться число 0. Коли натискають будь-яку кнопку, число збільшується на 1111 (вийдуть числа 1111, 2222 тощо) з інтервалом 300 мс. Після досягнення числа 9999 збільшення припиняється і програма повертається у початковий стан.
2. На індикаторі світиться число 1234. Коли натискають першу кнопку, число збільшується на 100, коли другу — на 1.
3. На індикаторі світиться число 9999. При натисненні першої кнопки остання цифра зменшується на 1 (виходить 9998), при наступному натисненні — третя цифра зменшується на 1 (виходить 9988), аналогічно для другої і першої цифри. Потім зменшується знову остання цифра.

4. Коли не натиснута жодна кнопка, світиться число 1111, коли натиснута перша — 8811, друга — 1188, обидві — 8888.

5. На індикаторі світиться число 4444. Коли натискають першу кнопку, число зменшується вдвічі, коли другу — збільшується на 123.

6. На індикаторі світиться число 9999. Коли натискають будь-яку кнопку, число зменшується на 1111 (вийдуть числа 8888, 7777 тощо) з інтервалом 300 мс. Після досягнення числа 0000 зменшення припиняється і програма повертається у початковий стан.

7. Коли не натиснута жодна кнопка, світиться число 9999, коли натиснута перша — 4949, друга — 9494, обидві — 4444.

8. На індикаторі світиться число 4320. Коли натискають будь-яку кнопку, число збільшується на 40 (вийдуть числа 4360, 4400 тощо) з інтервалом 300 мс. Після досягнення числа, більшого за 5000, збільшення припиняється і програма повертається у початковий стан.

9. На індикаторі світиться число 8675. Коли натискають будь-яку кнопку, число зменшується на 85 (вийдуть числа 8590, 8505 тощо) з інтервалом 300 мс. Після досягнення числа, меншого за 7500, зменшення припиняється і програма повертається у початковий стан.

10. На індикаторі світиться число 1. Коли натискають будь-яку кнопку, число збільшується вдвічі (вийдуть числа 2, 4, 8 тощо) з інтервалом 300 мс. Після досягнення числа, більшого за 9999, збільшення припиняється і програма повертається у початковий стан.

11. На індикаторі світиться число 0. При першому натисненні першої кнопки перша цифра збільшується на 1 (виходить 1000), при другому натисненні — друга цифра збільшується на 1 (виходить 1100), аналогічно для третьої і четвертої цифри. Потім збільшується знову перша цифра.

12. На індикаторі світиться число 1. Коли натискають будь-яку кнопку, число збільшується втричі (вийдуть числа 3, 9, 27 тощо) з інтервалом 300 мс. Після досягнення числа, більшого за 9999, збільшення припиняється і програма повертається у початковий стан.

13. На індикаторі число збільшується на 1 з інтервалом 1 мс. При натисненні першої кнопки збільшення припиняється та число тримається на індикаторі 5 секунд. Потім програма повертається у початковий стан. Рахунок продовжується з нуля.

14. На індикаторі світиться число 5678. Коли натискають першу кнопку, число зменшується на 200, коли другу — на 2.

15. Коли не натиснута жодна кнопка, світиться число 6543, коли натиснута перша — 1234, друга — 2198, обидві — 3333.

Зміст звіту

1. Тема та мета роботи.
2. Перелік використаного обладнання.
3. Стислий зміст теоретичних відомостей.
4. Лістинг власної програми з детальним поясненням кожного рядка.

5. Окремо виписати з файлу ev8031.c усі функції, що використовувались при написанні програми і дати їх детальне пояснення.
6. Відповіді на контрольні запитання.
7. Висновок.

Лабораторна робота № 6
ГЕНЕРАЦІЯ ЗВУКУ

Мета роботи: ознайомитись з базовими типами мікроконтролерів AVR, набути навиків роботи з п'єзовипромінювачами, видобування звуків різної тональності та частоти. Набути навиків прошивки управляючих програм в контролер.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; середовище програмування WinAVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

Принципова схема під'єднання п'єзовипромінювача

П'єзовипромінювач (зумер) (на схемі J7 speaker) випромінює звук. Він дешевший та менший за динамік, проте може випромінювати лише писк різної частоти (як старі мобільні телефони, що не мають поліфонії). Почути на ньому пісню у форматі MP3 не вдасться. Проте таких можливостей достатньо для вивчення мікроконтролерів та створення приладів, для яких достатньо найпростіших звуків.

Як бачимо зі схеми, він підключений до лінії порта PB6 через перемичку J8 та підсилювач на транзисторах VT4 та VT5. Тобто перемичка J8 повинна стояти. Коли виготовляли навчальну плату, не врахували, що лінія порта PB6 використовується також для прошивання МК, і упродовж процесу прошивання п'єзовипромінювач пищить.

Для того, щоб п'єзовипромінювач випромінював звук, необхідно програмно сформувати на лінії порта PB6 послідовність імпульсів. Звичайно, вона повинна бути налаштованою на вихід.

Періодичний писк

Періодичний писк часто використовується для привертання уваги: в будильниках, при виникненні аварійної ситуації тощо. У даній програмі зумер 0,3 с пищить з частотою 500 Гц, 0,7 с — пауза. Потім процес повторюється.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/

#include <avr/io.h>
#include <util/delay.h>

#define PORT_ZUMMER PORTB
#define ZUMMER PB6

typedef unsigned char byte;

int main(void)
```

```
{ DDRB = _BV(PB6);
PORTB = 0xFF;
unsigned int k=0;
for(;;)
{ _delay_ms(1);
PORT_ZUMMER &= ~_BV(ZUMMER);
_delay_ms(1);
PORT_ZUMMER |= _BV(ZUMMER);
if(++k >= 150)
{ k=0;
for(byte i=0; i<70; i++) _delay_ms(10);
}
}
}
```

На відміну від попередніх програм, ця не використовує зовнішньої пам'яті взагалі, тож команди для роботи з нею відсутні. Проте визначення типу byte (**typedef unsigned char** byte;) скопійовано з файлу ev8031.c.

Для зміни частоти звуку потрібно змінити затримку. Наприклад, для частоти 1000 Гц, написати _delay_ms(0.5).

Двотональний писк

Частота писку (500 Гц та 1000 Гц) змінюється 2 рази в секунду.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/

#include <avr/io.h>
#include <util/delay.h>
#define PORT_ZUMMER PORTB
#define ZUMMER PB6
#define false 0
#define true (!false)
typedef unsigned char boolean;

boolean zummerState = true;

void toggleZummer(void)
{ if(zummerState)
{ zummerState = false;
PORT_ZUMMER &= ~_BV(ZUMMER);
}
else
{ zummerState = true;
PORT_ZUMMER |= _BV(ZUMMER);
```

```
}
}

int main(void)
{ DDRB = _BV(PB6);
PORTB = 0xFF;
unsigned int k=0;
for(;;)
{ _delay_ms(0.5);
if(k >= 500) toggleZummer();
_delay_ms(0.5);
toggleZummer();
if(++k >= 1000) k=0;
}
}
```

Оскільки файл ev8031.c не використовується у даній програмі, визначення типу boolean скопійовано з нього.

Хід роботи

1. Знайти на навчальній платі п'єзовипромінювач (зумер) та встановити перемичку J8.
2. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.
3. Створити свою власну програму, яка забезпечує функції, описані в індивідуальному завданні, та скомпілювати її.
4. Запрограмувати МК, перевірити правильність виконання.

Індивідуальні завдання

Створити описаний звуковий ефект. Він має повторюватися до нескінченності.

1. Писк 660 Гц протягом 0,3 с, писк 909 Гц протягом 0,3 с, пауза 1,6 с.
2. Писк 526 Гц протягом 0,3 с, писк 909 Гц протягом 0,3 с, пауза 1,5 с.
3. Писк 500 Гц протягом 0,4 с, пауза 1,4 с.
4. Писк 588 Гц протягом 0,35 с, писк 877 Гц протягом 0,35 с, пауза 1,75 с.
5. Писк 650 Гц протягом 0,35 с, писк 1000 Гц протягом 0,35 с, пауза 1,8 с.
6. Писк 625 Гц протягом 0,3 с, пауза 1,1 с.
7. Писк 1000 Гц протягом 0,35 с, писк 526 Гц протягом 0,35 с, пауза 1,8 с.
8. Писк 833 Гц протягом 0,4 с, писк 500 Гц протягом 0,4 с, пауза 1,6 с.
9. Писк 833 Гц протягом 0,4 с, пауза 1,4 с.
10. Писк 909 Гц протягом 0,4 с, писк 588 Гц протягом 0,4 с, пауза 1,5 с.
11. Писк 877 Гц протягом 0,45 с, писк 555 Гц протягом 0,45 с, пауза 1,55 с.
12. Писк 555 Гц протягом 0,35 с, пауза 1,25 с.
13. Писк 660 Гц протягом 0,45 с, писк 1000 Гц протягом 0,45 с, пауза 1,9 с.
14. Писк 714 Гц протягом 0,45 с, писк 555 Гц протягом 0,45 с, пауза 2 с.
15. Писк 714 Гц протягом 0,3 с, пауза 1,2 с.

Зміст звіту

1. Тема та мета роботи.
2. Перелік використаного обладнання.
3. Стислий зміст теоретичних відомостей.
4. Лістинг власної програми з детальним поясненням кожного рядка.
5. Окремо виписати з файлу ev8031.c усі функції, що використовувались при написанні програми і дати їх детальне пояснення.
6. Відповіді на контрольні запитання.
7. Висновок.

Лабораторна робота № 7
РІДКОКРИСТАЛІЧНИЙ ІНДИКАТОР

Мета роботи: ознайомитись з базовими типами мікроконтролерів AVR. Набути навиків роботи з рідкокристалічними індикаторами, способів виведення текстової інформації на табло, робота з рядками.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; середовище програмування WinAVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

Алфавітно-цифрові рідкокристалічні індикатори

У даній роботі розглядаються алфавітно-цифрові рідкокристалічні індикатори (LCD) на основі контролера, сумісного з HD44780. Вони відображають символи ASCII з кодами від 32 до 122 та деякі інші, залежно від виробника та моделі. Також можна запрограмувати свої символи.

Для розуміння подальшого матеріалу необхідно прочитати ступені 6, 7 [2].

LCD на навчальній платі

На навчальній платі встановлено індикатор розміру 10x4. З точки зору керуючої програми він працює так само, як індикатор 20x2. Третій рядок є продовженням першого, а четвертий — другого.

LCD під'єднаний по 8-бітному інтерфейсу як комірка оперативної пам'яті МК. Для запису команди використовується функція Lcd_sendCommand(byte data), а даних — Lcd_sendData(byte data). Напряму їх викликати не доведеться, оскільки використовується бібліотека зі ступені 7 [2], змінена для LCD, що під'єднаний через інтерфейс зовнішньої оперативної пам'яті. Вона знаходиться у файлі lcd_memory.c.

Виведення тексту на LCD

Дана програма ініціалізує LCD і виводить текст “Hello, LCD!!! 10 chars, 4 lines.”

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
#include "lcd_memory.c"
unsigned char text[] =
"Hello, 10 chars, LCD!!! 4 lines. ";
int main(void)
{ ENABLE_EXTERNAL_RAM();
lcd_init();
lcd_com(0x80);
for(byte i=0; i<40; i++)
```

```
{ if(i == 20) lcd_com(0xC0);
lcd_dat(text[i]);
}
for(;;);
}
```

У масиві текст написаний не по порядку, тому що спочатку виводяться перший і третій рядки, а потім другий і четвертий. У нескінченному циклі процесор нічого не робить, але вилучити цикл не можна, щоб МК не скидався і не починав виконувати усю програму спочатку.

В кінці функції main() обов'язково повинен бути нескінченний цикл.

Символи кирилиці

Записувати символи як рядок у програмі дуже зручно, проте це працює лише для символів з кодами ASCII від 32 до 122. Решта символів, зокрема, букви кирилиці, закодовані не так, як у комп'ютері. Щоб правильно вивести їх на екран LCD, необхідна програма, що перекодовує символи. Її можна написати самому, а можна скористатися готовою програмою, написаною автором даних лабораторних робіт. Вона знаходиться у файлі КодировщикЖКИ.htm . Результатом програми є масив, який необхідно скопіювати у свою програму.

Рядок, що біжить

Програма показує у другому рядку заголовок “Планети”, а третій рядок біжить: “Сонячна система містить 8 планет: Меркурій, Венера, Земля, Марс, Юпітер, Сатурн, Уран, Нептун.”.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
#include "lcd_memory.c"
unsigned char text[] = { 0x20, 0x20,
0x43,0x6F,0xBD,0xC7,0xC0,0xBD,0x61,0x20,0x63,0xB8,0x63,0xBF,0x65,0xBC,0x
61,0x20,0xBC,0x69,0x63,0xBF,0xB8,0xBF,0xC4,0x20,'8',0x20,0xBE,0xBB,0x61,0x
BD,0x65,0xBF,0x3A,0x20,0x4D,0x65,0x70,0xBA,0x79,0x70,0x69,0xB9,0x2C,0x20
,0x42,0x65,0xBD,0x65,0x70,0x61,0x2C,0x20,0xA4,0x65,0xBC,0xBB,0xC7,0x2C,0
x20,0x4D,0x61,0x70,0x63,0x2C,0x20,0xB0,0xBE,0x69,0xBF,0x65,0x70,0x2C,0x20
,0x43,0x61,0xBF,0x79,0x70,0xBD,0x2C,0x20,0xA9,0x70,0x61,0xBD,0x2C,0x20,0x
48,0x65,0xBE,0xBF,0x79,0xBD,'.' };
unsigned char text1[] = "----------";
unsigned char text2[] = {
0x20,0xA8,0xBB,0x61,0xBD,0x65,0xBF,0xB8,0x20,0x20 };
unsigned char text3[] = "==========";
int main(void)
{ ENABLE_EXTERNAL_RAM();
lcd_init();
```

```
lcd_com(0x0C); //turn cursor off
lcd_com(0x80);
for(byte i=0; i<10; i++) lcd_dat(text1[i]);
lcd_com(0xC0);
for(byte i=0; i<10; i++) lcd_dat(text2[i]);
for(byte i=0; i<10; i++) lcd_dat(text3[i]);
for(;;)
{ for(byte offset=0; offset<=sizeof(text)-10; offset++)
{ lcd_com(0x8A);
for(byte i=0; i<10; i++) lcd_dat(text[offset+i]);
for(byte i=0; i<40; i++) _delay_ms(10); //delay 400 ms
}
for(byte i=0; i<200; i++) _delay_ms(10); //delay 2 s
}
}
```

Оскільки значення першого, другого та четвертого рядків не змінюються, вони передаються у LCD один раз на початку програми. У нескінченному циклі передається лише значення третього рядка.

Оператор **sizeof(**text**)** визначає довжину масиву text. Якби масив був описаний у вигляді рядка (рядок в Сі — це масив, що закінчується нуль-символом — символом з кодом 0), його довжина виявилась би на одиницю більша, тому що нуль-символ в кінці також враховується в довжину рядка.

Хід роботи

1. Знайти на навчальній платі рідкокристалічний індикатор.
2. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.
3. Створити свою власну програму, яка робить рядок, що біжить. Текст, який біжить, має складатися з Вашого прізвища та будь-якого речення кирилицею. Назва тексту відображається на другому рядку. Кнопка виконує функцію “Пауза/Продовження” (при одному натисненні рядок припиняє бігти, при повторному — продовжує з того ж місця, де зупинився). Скомпілювати програму.
4. Запрограмувати МК, перевірити правильність виконання.

Зміст звіту

1. Тема та мета роботи.
2. Перелік використаного обладнання.
3. Стислий зміст теоретичних відомостей.
4. Лістинг власної програми з детальним поясненням кожного рядка.
5. Окремо виписати з файлу ev8031.c усі функції, що використовувались при написанні програми і дати їх детальне пояснення.
6. Відповіді на контрольні запитання.
7. Висновок.

Лабораторна робота № 8
ВИМІРЮВАННЯ ТЕМПЕРАТУРИ. ІНТЕРФЕЙС I²C

Мета роботи: ознайомитись з базовими типами мікроконтролерів AVR, ознайомитись з принципами робот з інтерфейсом передачі інформації I²C. Набути навиків прошивки управляючих програм в контролер.

Обладнання: навчальна плата Open System EV8031/AVR; мікроконтролер Atmega8515; середовище програмування WinAVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів PonyProg.

Теоретичний матеріал

Принципова схема під'єднання датчика температури DS1621

На навчальній платі встановлений датчик температури DS1621. Він під'єднується до мікроконтролера по інтерфейсу I²C. У фірми Atmel та в книзі [3] він називається TWI з ліцензійних міркувань.

Для розуміння подальшого матеріалу необхідно прочитати крок 9 [1], а також розділи 18.1 та 18.2 [3].

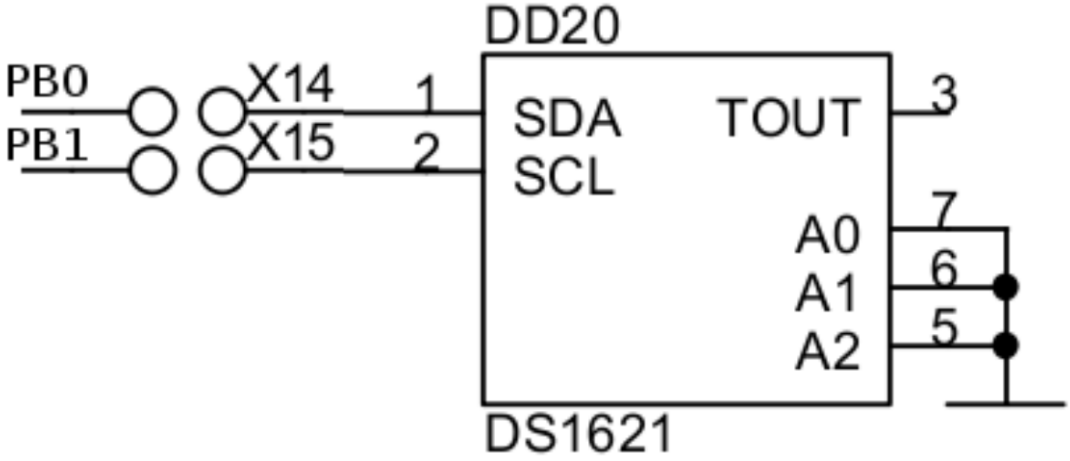


Рисунок 8 – Принципова схема під'єднання датчика температури DS1621

Вивід SDA під'єднаний до лінії порта PB0, а SCL — до PB1. Звичайно ж, перемички X14 та X15 повинні стояти. Слід зазначити, що резистори від цих двох ліній до +5В відсутні на навчальній платі, хоча й повинні стояти за стандартом I²C.

Інтерфейс I²C у мікроконтролері реалізований програмно за допомогою бібліотеки у файлі i2cSoft.c . Можна також написати свою бібліотеку та використовувати її. Проте для початку бажано отримати програму, що працює, з готовою бібліотекою, а потім змінювати на свій смак. Рядок

//#define USE_EXTERNAL_PULLUPS

закоментований, оскільки резистори від SCL і SDA до +5В відсутні. Якщо в іншому пристрої резистори будуть стояти, дві похилі риски на початку потрібно буде видалити.

Термометр

Вимірюється температура, результат відображається на світлодіодному індикаторі навчальної плати.

```
/*
Target MCU: ATmega8515
Target device: OpenSys EV8031
*/
#include <avr/io.h>
#include <util/delay.h>
#include "../../ev8031.c"
#include "i2cSoft.c"
#define THERMO_ADDR 0x90
//DS1621 thermometer test
int main(void)
{ ENABLE_EXTERNAL_RAM();
byte value, fraction;
byte digit[4];
DDRB = _BV(PB1);
PORTB = 0xFF;
setLedIndicatorEnabledAndDots(0x40);
i2c_start();
if(i2c_write(THERMO_ADDR)) setLeds(0xFF);
else setLeds(0x10);
i2c_write(0xAC);
i2c_write(0x01);
i2c_stop();
while(true)
{ i2c_start();
i2c_write(THERMO_ADDR);
i2c_write(0xAA);
i2c_start();
i2c_write(THERMO_ADDR | 0x01);
value = i2c_read(true);
fraction = i2c_read(false);
//start next measure
i2c_start();
i2c_write(THERMO_ADDR);
i2c_write(0xEE);
i2c_stop();
digit[0] = 0;
digit[1] = value / 10;
digit[2] = value % 10;
digit[3] = (fraction) ? 5 : 0;
setLedIndicatorLeft((digit[0] << 4) | digit[1]);
setLedIndicatorRight((digit[2] << 4) | digit[3]);
for(byte d = 80; d>0; d--) _delay_ms(10);
}
}
```

Обробка ACK проводиться у кінці функцій i2c_write та i2c_read. Затримка між вимірюваннями 800 мс, оскільки для вимірювання температури мікросхемі DS1621 потрібно близько 750 мс.

Хід роботи

1. Знайти на навчальній платі датчик температури.
2. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.
3. Створити свою власну програму, яка забезпечує функції, описані в індивідуальному завданні, та скомпілювати її.
4. Запрограмувати МК, перевірити правильність виконання.

Індивідуальні завдання

1. Вимірювати температуру кожну секунду та відображати її на світлодіодному індикаторі. Якщо значення температури більше 25 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.
2. Спочатку вивести на LCD напис “Термометр”. Після натиснення кнопки почати вимірювання температури та вивести на LCD напис “Секунду”. Через секунду зчитати результати вимірювання та відобразити їх на LCD. Чекати наступного натискання кнопки.
3. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури більше 24 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.
4. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури більше 24 °С, вивести на індикатор напис “Вище норми”, інакше “Норма”.
5. Вимірювати температуру кожну секунду та відображати її на світлодіодному індикаторі. Якщо значення температури менше 22 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.
6. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури менше 23 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.
7. Спочатку вивести на LCD напис “Термометр”. Після натиснення кнопки почати вимірювання температури та очистити другий рядок LCD. З інтервалом 0,15 с посилати на LCD символ “=”. Коли рядок заповниться, зчитати результати вимірювання та відобразити їх на LCD. Чекати наступного натискання кнопки.
8. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури знаходиться в діапазоні від 19 °С до 25 °С, вивести на індикатор напис “Норма”, якщо нижче 19 °С — “Холодно”, якщо вище 25 °С— “Жарко”.
9. Вимірювати температуру кожну секунду та відображати її на світлодіодному індикаторі. Якщо значення температури знаходиться в діапазоні від 20 °С до 24 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.

10. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури знаходиться в діапазоні від 19 °С до 25 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.

11. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури виходить за межі діапазону від 20 °С до 26 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.

12. Спочатку вивести на LCD напис “Термометр” та погасити лінійку світлодіодів. Після натиснення кнопки почати вимірювання температури та увімкнути лінійку світлодіодів. Через секунду вимкнути світлодіоди, зчитати результати вимірювання та відобразити їх на LCD. Чекати наступного натискання кнопки.

13. Вимірювати температуру кожну секунду та відображати її на світлодіодному індикаторі. Якщо значення температури виходить за межі діапазону від 21 °С до 25 °С, на лінійці світлодіодів ввімкнути світлодіоди через один, інакше вимкнути.

14. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури менше 23 °С, вивести на індикатор напис “Нижче норми”, інакше “Норма”.

15. Вимірювати температуру кожну секунду та відображати її на рідкокристалічному індикаторі. Якщо значення температури знаходиться в діапазоні від 20 °С до 27 °С, вивести на індикатор напис “Норма”, якщо нижче 20 °С — “Холодно”, якщо вище 27 °С— “Жарко”.

Зміст звіту

1. Тема та мета роботи.
2. Перелік використаного обладнання.
3. Стислий зміст теоретичних відомостей.
4. Лістинг власної програми з детальним поясненням кожного рядка.
5. Окремо виписати з файлу ev8031.c усі функції, що використовувались при написанні програми і дати їх детальне пояснення.
6. Відповіді на контрольні запитання.
7. Висновок.

ЛІТЕРАТУРА

1. Рюмік С. М. Мікроконтролери. Крок 1-10 //журнал Радіоаматор № 3-12/2004, с. 35–39.
2. Рюмік С. М. Мікроконтролери AVR. Ступінь 1-10 //Радіоаматор №1-7, 9-11/2005, с. 35–39.
3. Евстифеев А. В. Микроконтроллеры AVR семейств Tiny и Mega фирмы “Atmel” — М.: Издательский дом “Додэка-XXI”, 2004. – 560 с.
4. Баранов В. Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы.—М.: Издательский дом “Додэка-XXI”, 2004. – 288 с.
5. Технічний опис та інструкція по експлуатації учбово-налагоджувального стенду EV8031/AVR.
6. Саямов Э.А. Средства воспроизведения и отображения информации: Учебн. пособие для вузов. – М.: Высш. школа, 1982. – 335 с.
7. Яблонский Ф.М., Троицкий Ю.В. Средства отображения информации: Учеб. для вузов спец. «Промышленная электроника». – М.: Высш. школа. – 1985. – 200 с.
8. Виноградов В. Обслуживание и ремонт стационарных цветных телевизоров. – Санкт-Петербург, «Кристал», 1996. – 384 с.
9. Гусев В.Г., Гусев Ю.М. Электроника: Учеб. пособие для приборостроит. спец. вузов. – 2-е изд., перераб. и доп. – М.: Высш. шк. 1991. – 622 с.
10. Савета Н.Н. Периферийные устройства ЭВМ: Учебное пособие для студентов вузов. – М.: Машиностроение, 1987. – 304 с.
11. Иванов Е.Л. и др. Периферийные устройства ЭВМ и систем: Учеб. пособие для вузов по спец. «ЭВМ». – М.: Высш. шк., 1987. – 319 с.