МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ МІСЬКОГО ГОСПОДАРСТВА імені О. М. БЕКЕТОВА

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

до виконання лабораторних робіт
з навчальної дисципліни

«МІКРОКОНТРОЛЕРИ В ЕЛЕКТРОЕНЕРГЕТИЦІ»

(для студентів 3 курсу зі скороченим терміном навчання, 4 курсу денної та заочної форм навчання спеціальності 141 – Електроенергетика, електротехніка та електромеханіка)

Харків
ХНУМГ ім. О. М. Бекетова
2020

Методичні рекомендації до виконання лабораторних робіт з навчальної дисципліни «Мікроконтролери в електроенергетиці» для студентів 3 курсу зі скороченим терміном навчання, 4 курсу денної та заочної форм навчання спеціальності 141 – Електроенергетика, електротехніка / Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова ; уклад. Ю. В. Ковальова. – Харків : ХНУМГ ім. О. М. Бекетова, 2020. – 64 с.

Укладач канд техн. наук, ст. викл. Ю. В. Ковальова

Рецензент

П. П. Рожков, кандидат технічних наук, доцент кафедри систем електропостачання та електроспоживання міст Харківського національного університету міського господарства імені О. М. Бекетова

Рекомендовано кафедрою систем електропостачання та електроспоживання міст, протокол № 7 від 25.02.2020.

ЗМІСТ

Вступ.....4
Лабораторна робота № 1.....5
Лабораторна робота № 2.....17
Лабораторна робота № 3.....27
Лабораторна робота № 4.....36
Лабораторна робота № 5.....47
Лабораторна робота № 6.....55
Список рекомендованої літератури.....63

ВСТУП

Метою проведення лабораторних робіт є практичне ознайомлення студентів з принципом роботи мікроконтролерів, методами їх програмування та перевірки. Перед виконанням кожної лабораторної роботи студенти повинні вивчити теоретичний матеріал з даної теми за конспектом лекцій, підручником або за методичними рекомендаціями, ознайомитись з програмою роботи.

Студенти, які прийшли на заняття непідготовленими або не склали звіт про попередню роботу, до виконання наступної роботи не допускаються.

Звіт оформлюється кожним студентом самостійно та прикладає письмові відповіді на три контрольних запитання до роботи. Робота вважається виконаною і прийнятою при правильних і повних відповідях на контрольні запитання. Після закінчення лабораторної роботи викладач перевіряє правильність її виконання і робить відмітку в своєму журналі.

При виконанні лабораторних робіт студенти повинні дотримуватись правил техніки безпеки.

ЛАБОРАТОРНА РОБОТА № 1

КОМП'ЮТЕРНІ ЗАСОБИ ДЛЯ ПРОГРАМУВАННЯ МІКРОКОНТРОЛЕРІВ

Мета роботи – отримати практичні навички складання та перевірки програм для мікроконтролерів в програмних середовищах AVR Studio та Proteus.

КОРОТКА ІНФОРМАЦІЯ ДО ВИКОНАННЯ РОБОТИ

Розробка пристроїв для електроенергетики на основі мікроконтролерів відбувається в наступній послідовності: розробка електричної схеми мікроконтролерного пристрою; розробка програмного забезпечення для мікроконтролера; перевірка правильності роботи програмного забезпечення.

Для складання програм та перевірки правильності їх роботи кампанії-виробники мікроконтролерів створюють комп'ютерні програми, які мають абревіатуру IDE, що розшифровується як «Integrated Development Environment» та перекладається як «інтегроване середовище розробки». Кампанія-виробник ATmel мікроконтролерів серії AVR розробила IDE під назвою AVR Studio, яка займає пам'яті близько 70 МБ, сумісна з Windows сучасних версій та забезпечує: набір тексту програм за допомогою клавіатури; перевірку правильності набору мнемоніки команд; перетворення (трансляція) тексту програм в 16-річний код; створення окремого файлу програми формату «hex» в двійковому коді для завантаження в ROM-пам'ять мікроконтролера; покрокову перевірку виконання команд програми.

Програмні витримки часу в AVR Studio не перевіряють, оскільки процес покрокової перевірки потребує забагато часу. Витримку часу і роботу всієї електричної схеми мікроконтролерного пристрою перевіряють з використанням інших комп'ютерних програм. Такі програми містять у своїй бібліотеці необхідні елементи для складання схем: резистори, діоди, транзистори і таке інше та дозволяють на екрані комп'ютера складати будь-яку віртуальну електричну

схему без використання реальних деталей. Існують декілька таких програм, але не всі з них в своїх бібліотеках містять мікроконтролери необхідних серій. Компанія Labcenter Electronics розробила програму-імітатор електронних схем Proteus, яка займає близько 100 мегабайт пам'яті, сумісна з Windows та має в своїй бібліотеці мікроконтролери серії AVR. Програма Proteus містить дві підпрограми: ISIS – для складання віртуальних електричних схем з мікроконтролерами і ARES – для розробки друкованих плат.

Необхідно відзначити, що програми AVR Studio і Proteus містять значну кількість функціональних можливостей, але в лабораторних роботах використовуються лише деякі з них.

ПРОГРАМА РОБОТИ

1. Складання електричної схеми лабораторної роботи в програмному середовищі Proteus.

2. Набір тексту програми та її трансляція в програмному середовищі AVR Studio.

3. Перевірка правильності роботи схеми лабораторної роботи.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

І Складання електричної схеми лабораторної роботи в програмному середовищі Proteus.

Схема лабораторної роботи для керування світлодіодами з використанням мікроконтролера ATmega8 показана на рисунку 1.1. Схема забезпечує почергове включення світлодіодів з частотою 1 Гц для створення ефекту «рухомого вогню». На схемі показаний корпус мікросхеми мікроконтролера ATmega8, до виводів якого підключені 8 світлодіодів VD1-VD8 з резистором R1.

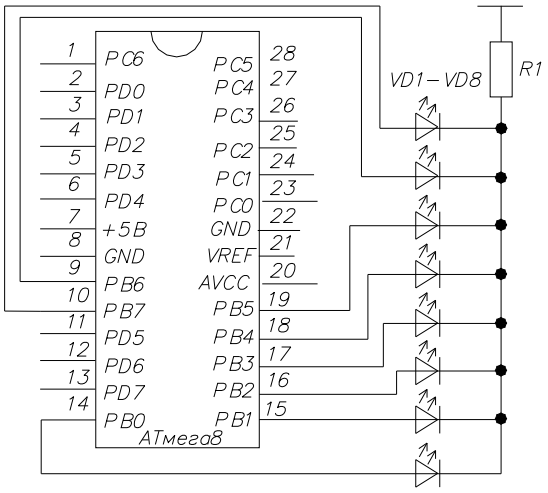

Рисунок 1.1 – Схема для почергового вмикання світлодіодів

Виводи мікроконтролера мають такі параметри: напруга $U_{МК} = 5$ В, допустимий струм $I_{МК} = 20$ мА. Для обмеження величини струму на рівні допустимого використаний резистор R1, величина опору якого обчислюється за формулою $R_{R1} = (U_{МК} - U_{СД})/I_{МК}$, де $U_{СД} = 2$ В – спад напруги на світлодіоді.

Схема складена в програмному середовищі Proteus показана на рисунку 1.2.

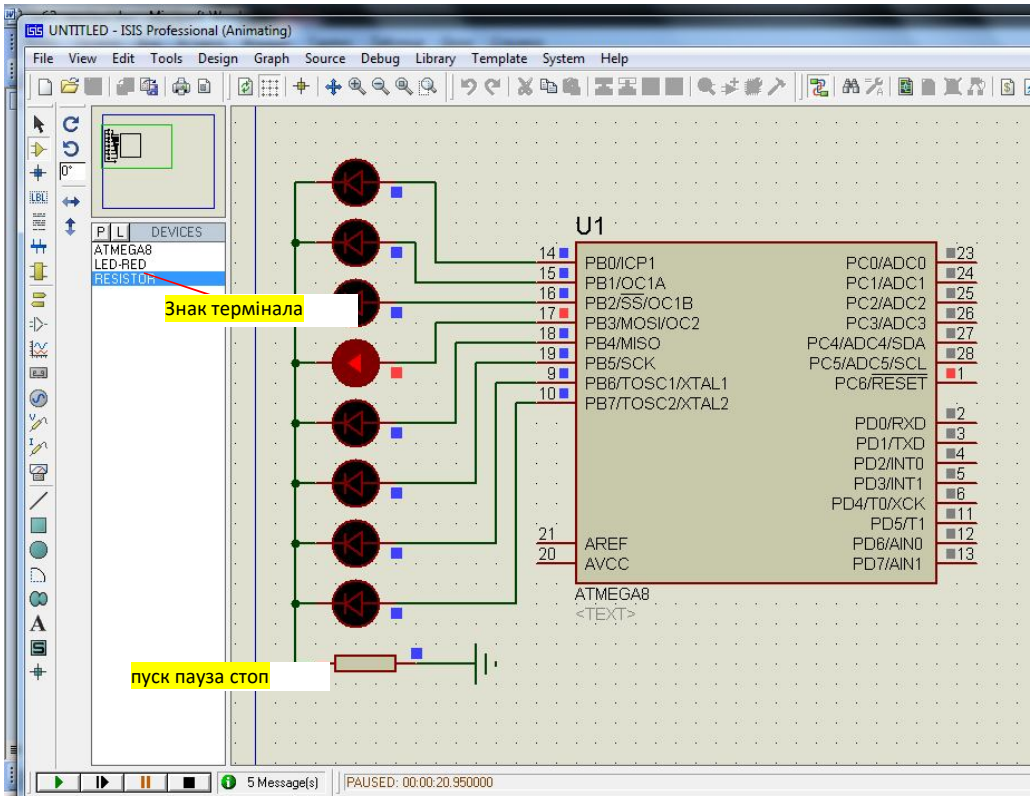

Рисунок 1.2 – Схема лабораторної роботи в програмному середовищі Proteus

Складання схеми відбувається так:

1. Запустити програму Proteus подвійним кліком на ярлик ISIS на робочому столі Windows. З'являється робочий стіл програми ISIS (див. рис. 1.2), який складається з головного меню і панелей інструментів. Верхня панель інструментів призначена для редагування схеми. Нижня панель інструментів призначена для керування схемою кнопками пуску, паузи та зупинки. Ліва бокова призначена для складання схеми, праворуч від якої розташоване вертикальне віконце призначене для складання бібліотеки елементів схеми. Робоче поле призначене для складання схеми.

Для складання бібліотеки елементів схеми навести курсор на букву «P», спливають слова «Pick from librerias», які перекладаються як «вибирати з бібліотеки». Кнопка з літерою «L» відкриває менеджер бібліотек для підключення нових бібліотек елементів.

2. Клікнути на кнопку з буквою «P», відкривається вікно «Pick Devices» з віконцями «Keywords», «Category», «Sub-Category», які перекладаються як: «вибір приладів», «ключові слова», «каталог» і «підкаталог». Елементи схем можна вибирати за двома варіантами: вибирати в каталозі та в подкаталозі або вводити назву елемента у вікні «Keywords». Другий варіант зручніший, оскільки перший варіант вимагає знання маркування типів елементів в англійському позначенні.

3. У віконці «Keywords» набрати на клавіатурі малими або великими літерами слово «ATmega8», в правій частині робочого столу з'являється віконце зі схемою мікроконтролера ATmega8 і зовнішнім виглядом мікросхеми. У нижній правій частині робочого столу клікнути на кнопку ОК, в полі бібліотеки з'являється слово «ATmega8».

4. У віконці «Keywords» набрати на клавіатурі «led red» – червоний світлодіод. У верхній частині панелі з'являється зображення світлодіода. У нижній правій частині робочого столу клікнути на кнопку ОК, в полі бібліотеки з'являється слово «led red».

5. У віконці «Keywords» набрати на клавіатурі слово «resistor» і виконати аналогічні дії.

6. Перенести елементи з бібліотеки на робоче поле. Для цього клікнути на назві кожного елемента у віконці бібліотеки і далі подвійний "клік" на робочому полі в місці розташування елемента.

7. Вивід джерела напруги +5 В підключається до мікроконтролера автоматично. На робоче поле перенести вивід джерела напруги -5 В, який має назву «ground», тобто, «земля». На боковій панелі інструментів клікнути на позначку термінал (див. рис. 1.2), в полі бібліотеки з'являється список елементів джерела живлення. Клікнути на кнопці «ground» і далі подвійний клік на робочому полі в місті розташування.

8. Для з'єднання елементів між собою підвести курсор до виводу елемента, колір курсора стає зеленим, зробити клік на ліву кнопку мишки і, не відпускаючи, вести лінію до виводу іншого елемента і зробити клік.

9. Для установки величину опору резистора зробити подвійний клік лівою кнопкою мишки на схемному зображенні резистора. У віконці, що відкрилося ввести величину опору резистора «0.2k». В десяткових дробах замість коми ставиться крапка без інтервалу між числом і буквою «k».

Способи редагування схеми. Для перенесення елемента в інше місце робочого поля зробити клік лівою кнопкою миші на елементі і, не відпускаючи кнопки, перенести в інше місце. Для копіювання елементів в робочому полі клікнути лівою кнопкою миші на елементі, встановити курсор на нове місце і подвійний клік лівої кнопкою в місті розташування елемента.

Видаляється елемент подвійним кліком правою кнопкою мишки по його зображенню. Для повороту елементів на робочому полі навести курсор на елемент та зробити клік правою кнопкою миші. Відкривається вікно форматування, в якому повернути елемент на потрібний кут або його віддзеркалення.

Буквені позначення елементів схеми від Proteus можна видаляти: подвійний клік на схемному позначенні і у віконці, що відкрилося клікнути на слово «delete».

Робити пояснювальні написи біля елементів схеми: клік на місці установки напису; клік на букву «А» на вертикальній панелі інструментів; у вікні, що відкрилося в боксі String набрати слово англійською мовою; справа в боксі Fort fase встановити шрифт Times New Roman; нижче в боксі Height встановити висоту букв «100.ih».

Для копіювання схеми з Proteus в документ Word в головному меню клікнути на кнопки edit – правка та copy to clipboard. Далі відкрити документ Word і виконати вставку.

10. Звернути програму Proteus та перейти в програмне середовище AVR Studio.

II Набір тексту програми та її трансляція в програмному середовищі AVR Studio

Програма для почергового включення світлодіодів представлена в таблиці 1.1. В лівому стовпчику таблиці наведені мнемоніки команд, а в правому – коментарі до команд. Мнемоніки команд набрати в документі Word, шрифт Times New Roman, розмір 12, коментарі до команд не набирати. Набраний текст зберегти на диску D.

Таблиця 1.1 – Текст програми для керування світлодіодами

Мнемоніка команд	Коментарі до команд
.devise ATmega8	директива вказує, що програма для ATmega8
.include "m8def.inc"	директива підключає файл з бібліотеки AVR studio
.cseg	команди завантажуються в комірки ROM-пам’яті
.org 0	директива вказує на адресу 1-ї комірки ROM-пам’яті
ldi r16,high(RAMEND)	завантаження адреси останньої комірки RAM-пам’яті
out SPH, r16	пересилання даних в старший байт показника стека
ldi r16, low(RAMEND)	адреса передостанньої комірки RAM-пам’яті
out SPL, r16	пересилання даних в молодший байт показника стека
ldi r16,0xff	завантаження робочого регістра r16 одиницями
out ddrb, r16	настройка всіх виводів порта b на вихід
out portb, r16	потушити всі світлодіоди

Продовження таблиці 1.1

Мнемоніка команд	Коментарі до команд
ldi r16, 0x05	завантаження коду для настройка таймера
out TCCR1B, r16	налаштування таймера
control:	мітка для безкінечного циклу
ldi r17, 0b10000000	завантажити одиницю в старший розряд регістра r17
sdvig:	мітка для переходу
ldi r16, 0xff	завантаження робочого регістра r16 одиницями
eor r16, r17	пересилання даних
out portb, r17	включити світлодіод
rcall zad	виклик підпрограми затримки часу
lsr r17	зсув одиниці вправо
brcc sdvig	якщо зсув не дійшов до кінця перейти на мітку
rjmp control	перехід в початок програми
zad:	мітка підпрограми затримки часу
push r16	завантажити дані регістра r16 в стек
ldi r16,0	завантажити регістр r16 нулями
out TCNT1H, r16	завантажити в лічильні регістри таймера «нулі»
out TCNT1L, r16	завантажити в лічильні регістри таймера «нулі»
w1:	мітка для переходу
in r16, TCNT1L	дані з TCNT1L переслати в r16
cpi r16, low(0x010C)	порівняти дані r16 з витримкою часу
brlo w1	перейти на мітку якщо менше
in r16, TCNT1H	дані з TCNT1L переслати в r16
cpi r16, high(0x010C)	порівняти дані r16 з витримкою часу
brlo w1	перейти на мітку якщо менше
pop r16	переслати дані в основну програму
ret	вихід з підпрограми затримки

Для трансляції програми в програмному середовищі AVR Studio виконати такі дії:

1. Запустити програму AVR Studio. Для цього на робочому столі Windows послідовно клікати на кнопки Пуск, Всі програми, Atmel AVR Tools і AVR studio. В результаті з’являється вікно № 1 (рис. 1.3), яке має дві кнопки New Project – новий проект і Open – відкрити раніше створений проект.

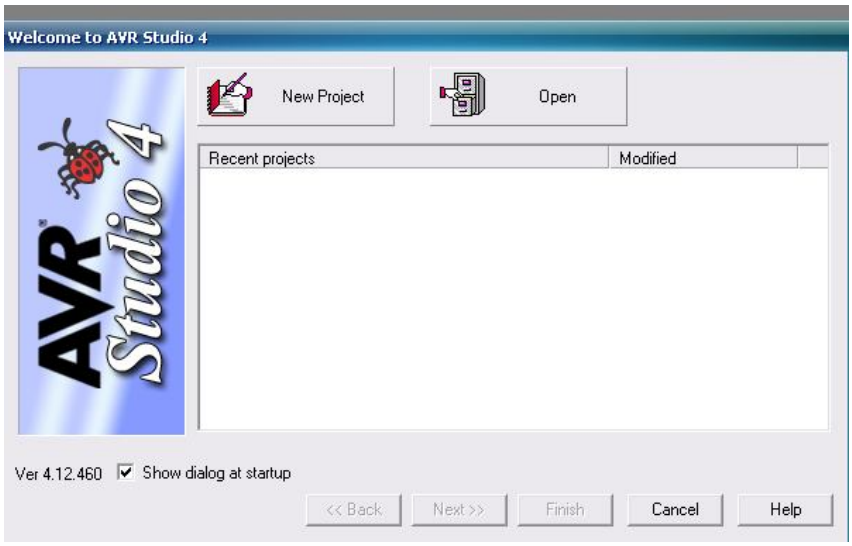

Рисунок 1.3 – Вікно № 1

2. Клікнути на кнопку New Project, з'являється вікно № 2 (рис. 1.4) з назвою Create New Project.

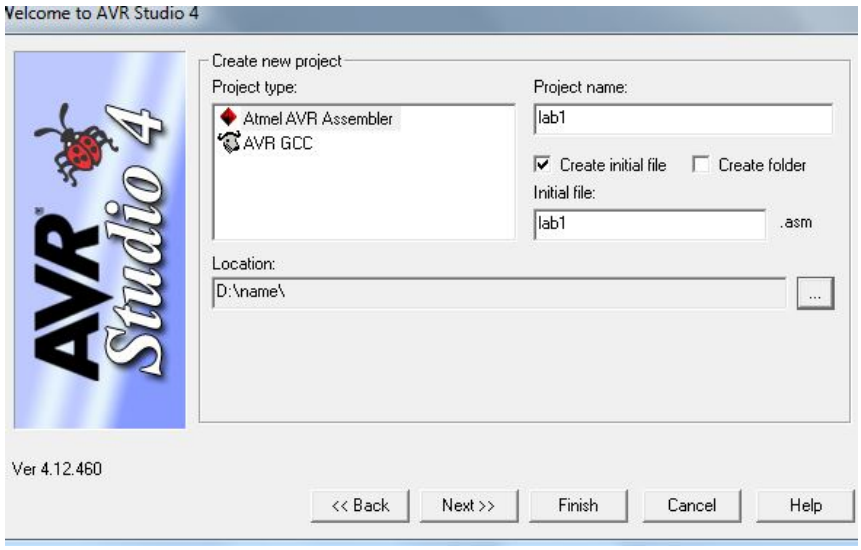

Рисунок 1.4 – Вікно № 2

Вікно № 2 має 6 віконець: Project type, Project name, Create initial file, Create folder, Initial file і Location.

3. У віконці Project type клікнути на кнопку Atmel AVR Assembler (вибір мови програмування Assembler); у віконці Project name набрати на клавіатурі ім'я проекту «lab1», яке повториться у віконці Initial file; у віконці

Create initial file залишити галку; у віконці Create folder прибрати галку. Клікнути на кнопку з трьома крапками, яка розташована праворуч від віконця Location, з'являється вікно № 3 під назвою Selekt folder (рис. 1.5).

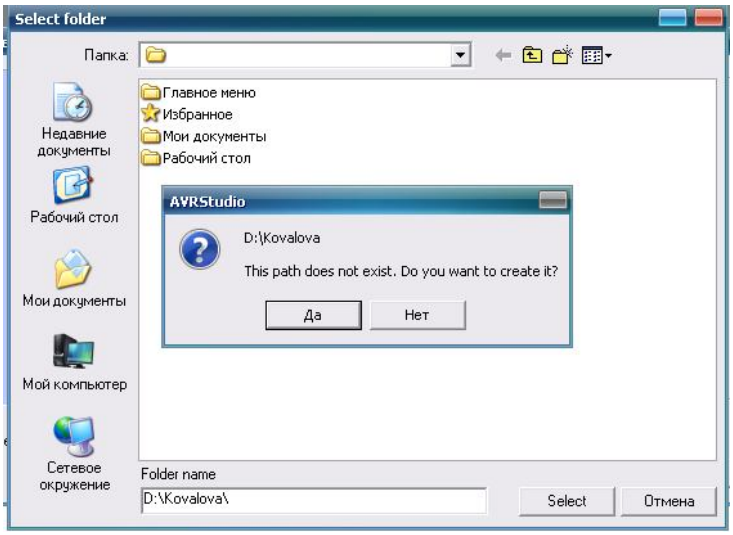

Рисунок 1.5 – Вікно № 3

У вікні № 3 клікнути на кнопку Мій комп'ютер і подвійно клікнути на диск D. У нижньому віконці Folder name на клавіатурі набрати своє прізвище англійською мовою і праворуч від нього клікнути на кнопку Selekt. При цьому з'являється вікно з пропозицією створити папку, клікнути на кнопку «так» і далі на кнопку «Next».

4. З'являється вікно під назвою Select debug platform and device з двома віконцями (рис. 1.6). У віконці Debug platform клікнути на кнопку AVR Simulator, у віконці Device клікнути на кнопку ATмега8 і далі клікнути на кнопку Finish.

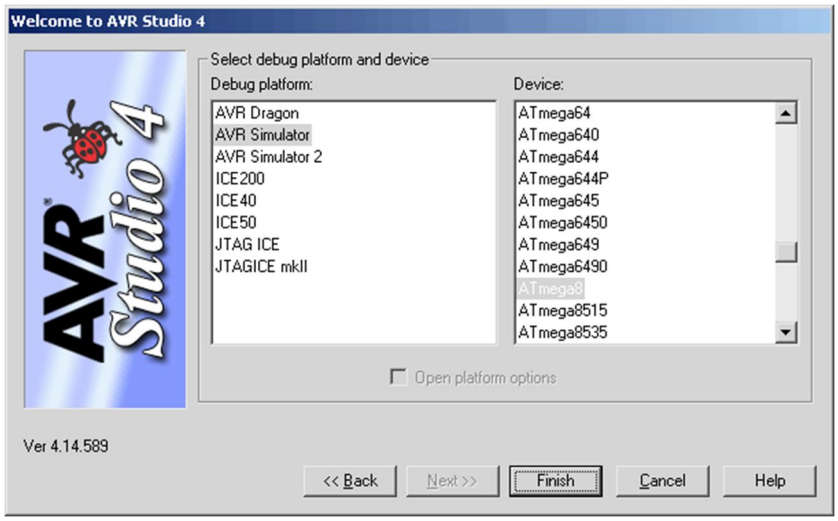

Рисунок 1.6 – Вікно № 4

5. З’являється вікно робочого стола програми AVR studio (рис. 1.7), яке складається з головного меню і панелі інструментів. Нижче панелі інструментів розташовані три віконця: а) зліва віконце проекту; б) справа віконце для набору команд програми; в) внизу віконце повідомлень.

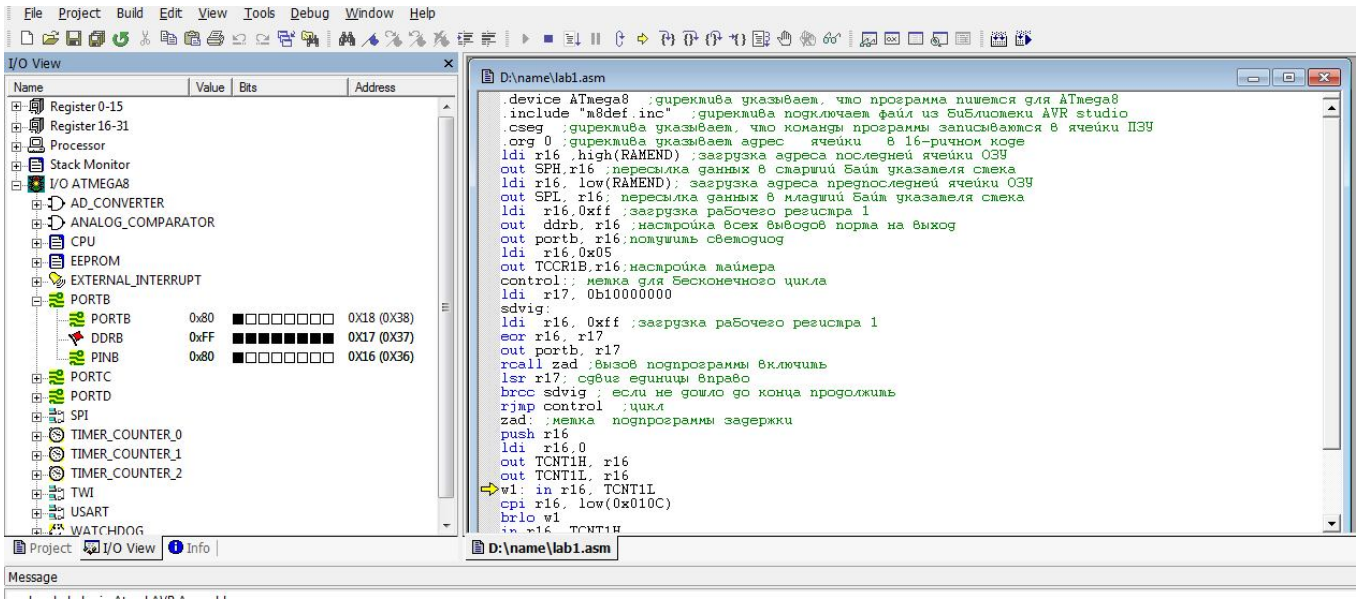

Рисунок 1.7 – Робочий стіл AVR studio

Робочий стіл AVR studio тимчасово звернути, відкрити на диску D текстовий документ Word з набраною програмою та скопіювати. Відкрити

робочий стіл AVR studio та вставити текст програми у віконце для набору програм.

6. Для трансляції програми в головному меню клікнути на кнопку Build і в розкритому підменю клікнути на кнопку Build. У нижній частині робочого столу з'являється віконце повідомлень Build з результатами трансляції програми (рис. 1.8).

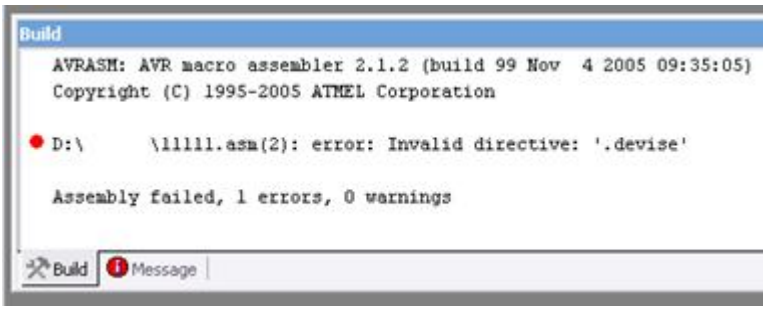

Рисунок 1.8 – Вікно повідомлень про результати трансляції

У нижньому рядку написано «Assambly falled, 1 errors», тобто, в тексті програми виявлена синтаксична помилка при наборі команди. Над повідомленням про помилку вказаний рядок, в якому виявлена помилка. Для виправлення помилки навести курсор на цей рядок і двічі клікнути лівою кнопкою мишки, при цьому в тексті програми з'являється синя стрілка навпроти команди з помилкою. Після виправлення помилки повторити трансляцію. Якщо при трансляції знову вказується на ту ж команду, а там все правильно, перенабрати цю команду по новому безпосередньо у віконці. Така ситуація виникає тому, що при копіюванні тексту програми в форматі word і вставки в робоче поле можливі випадки неправильного копіювання окремих букв в командах, тому їх треба перенабрати безпосередньо на робочому полі. Якщо виявлено декілька помилок, слід повторити аналогічні дії для кожної помилки. Після виправлення всіх синтаксичних помилок з'являється повідомлення «Assembly complete, 0 errors, 0 warnings» (рис. 1.9), що в перекладі означає «Асемблування завершено, нуль помилок, нуль попереджень».

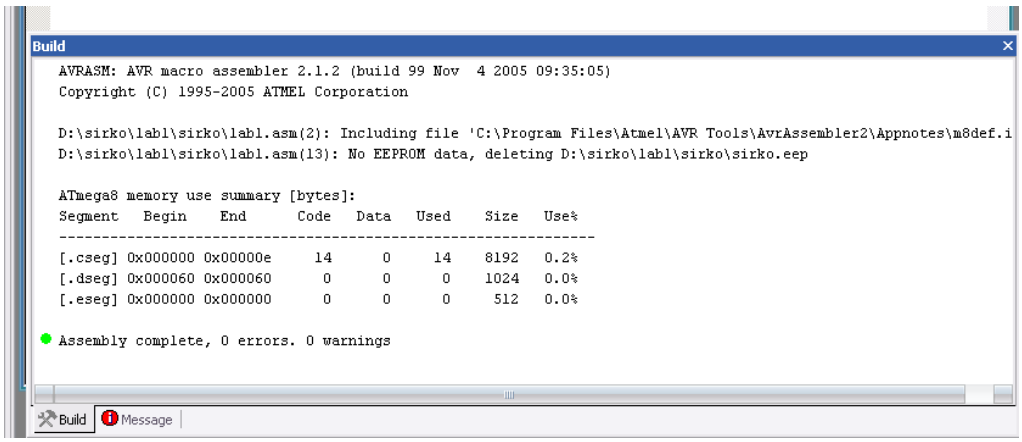

Рисунок 1.9 – Вікно повідомлень про результати трансляції після виправлення помилок

Це означає, що трансляція успішно завершена та AVR studio створила файл формату «.hex» для завантаження в ROM-пам’ять мікроконтролера. При закриванні програми створюється папка, яка автоматично зберігається на диску D. Важливо! Переносити папку проекту в інші папки на диску D для зберігання можна, але для роботи програми AVR studio папку необхідно повернути на диск D, оскільки в інших папках програма AVR studio не працює. Це викликано тим, що при налаштуванні програми був вибраний саме диск D.

III Перевірка правильності роботи програми

Для завантаження програмного файла формату «.hex» в ROM-пам’ять мікроконтролера відкрити програму Proteus та зробити подвійний клік лівою кнопкою миші по схемі мікроконтролера. На робочому полі з’являється вікно Edit Component (рис 1.10). У віконці CKSEL fuses встановити частоту синхрогенератора 1 МГц. Праворуч від напису Program file розташоване віконце та знак відкрита папка, клікнути на цей знак та вказати шлях до файлу формату «.hex» і клікнути на кнопку ОК.

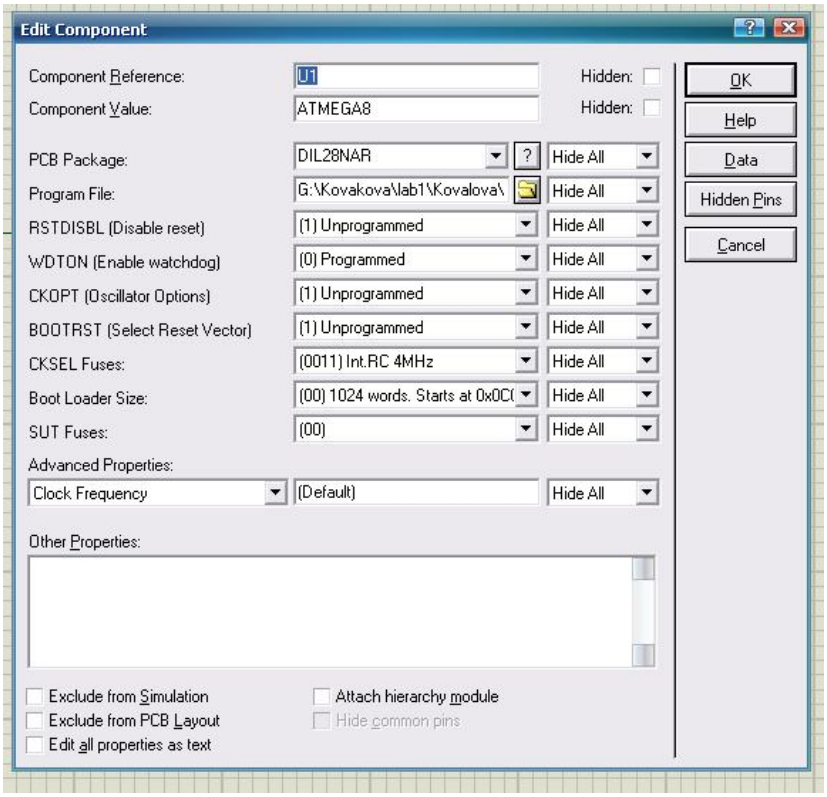

Рисунок 1.10 – Вікно параметрів мікроконтролера

Для перевірки роботи схеми в нижньому лівому кутку робочого столу Proteus розташована панель керування роботою схеми (див. рис. 1.9) з кнопками «пуск», «пауза», «стоп». Клікнути на кнопку «пуск», світлодіоди починають по черзі включатися червоним світлом. Для зупинки роботи схеми навести курсор на кнопку «стоп».

При роботі схеми змінюються логічні рівні на виводах мікроконтолера кольоровими квадратами: червоний колір – логічна «1»; синій колір – логічний «0»; сірий колір – високоомний вхід «Z-стан»; жовтий квадрат – конфлікт на лінії, наприклад, лінія замкнута на землю.

ПОРЯДОК ОФОРМЛЕННЯ ЗВІТУ

Звіт про виконану лабораторну роботу оформлюється в тому ж зошиті для конспекту лекцій та практичних занять. Вказати номер роботи, її назву та мету. Намалювати реальну принципову електричну схему лабораторної роботи,

переписати команди програми з коментарями, привести письмові відповіді на три контрольних запитання, вказані викладачем.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Пояснити призначення програмного середовища AVR Studio.
2. Пояснити, що означає трансляція програми.
3. Пояснити, що означає симуляція програми в AVR Studio.
4. Намалювати вольт-амперну характеристику світлодіода.
5. Пояснити призначення вікон робочого стола AVR Studio.
6. Пояснити призначення віконець вікна з назвою Create New Project.
7. Пояснити призначення віконець вікна з назвою Select debug platform and device.
8. Пояснити послідовність виправлення синтаксичних помилок в командах.
9. Пояснити призначення програмного середовища Proteus.
10. Скласти послідовність завантаження програми в постійну пам’ять.
11. Пояснити послідовність дій для встановлення величини опору резистора в програмі Proteus.
12. Розрахувати величину опору резистора R1.
13. Пояснити значення кольору квадратів на виводах мікроконтролера.
14. Вказати електричні параметри виводів мікроконтролера.

ЛАБОРАТОРНА РОБОТА № 2

ПРОГРАМУВАННЯ ПОРТІВ ТА АНАЛОГОВОГО КОМПАРАТОРА

Мета роботи – отримати практичні навички для програмування портів та аналогового компаратора з використанням програмних середовищ AVR Studio та Proteus.

КОРОТКА ІНФОРМАЦІЯ ДО ВИКОНАННЯ РОБОТИ

Порти це периферійні пристрої мікроконтролера для прийому вхідних і видачі вихідних сигналів, тобто, є виводами мікросхеми для зовнішніх підключень. Мікроконтролер Atmega8 має три 8-мирозрядних порти з іменами B, D, C. Кожен порт або окремий його розряд можливо програмувати на вхід або на вихід. Для цього використовують три регістра: регістр «ddr» для програмування порту на вхід або на вихід; регістр «port» для керування портом; регістр «pin» для зчитування даних з порту.

Для програмування окремих розрядів порту на вхід у відповідні розряди регістра «ddr» завантажують «нулі», а на вихід – «одиниці». При підключенні до вхідного порту контактних елементів необхідно апаратно підключати зовнішній резистор або програмно підключати вбудований резистор. В портах запрограмованих на вхід регістр «port» підключає на вході порту вбудовані резистори. У портах запрограмованих на вихід регістр «port» подає на виходи логічний «0» або «1».

Сигнал називається аналоговим, якщо його величина змінюється в часі без розривів, наприклад, синусоїда напруги. Аналоговий компаратор це периферійний пристрій з двома входами: сигнальний і опорний. В результаті порівняння напруг на входах на виході змінюється стан з логічного «нуля» на «одиницю» або навпаки. Програмування компаратора полягає в установці комбінації біт в розряди регістра ACSR для керування компаратором.

ПРОГРАМА РОБОТИ

1. Вивчити принципову електричну схеми лабораторної роботи та скласти її у програмному середовищі Proteus.

2. Вивчити програму керування портами, провести набір та трансляцію в програмному середовищі AVR Studio.

3. Перевірити правильність роботи схеми.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

І Складання електричної схеми лабораторної роботи в програмному середовищі Proteus. Принципова електрична схема лабораторної роботи показана на рисунку 2.1.

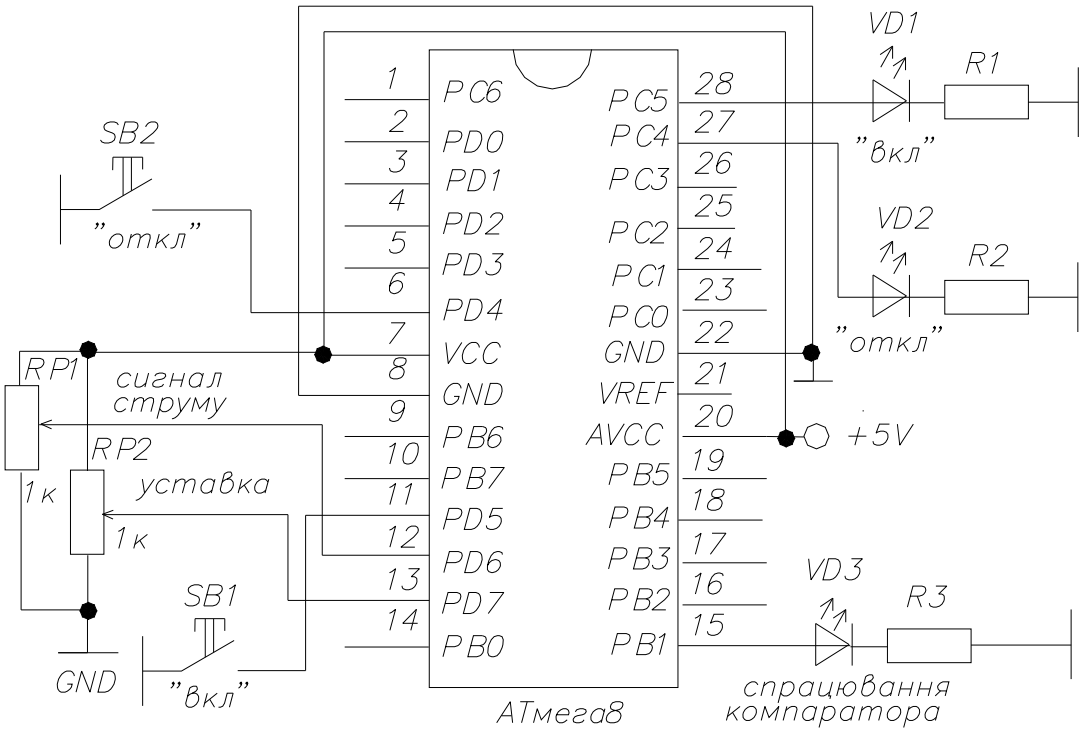

Рисунок 2.1 – Схема лабораторної роботи

Схема лабораторної роботи імітує ручне вмикання та вимикання вакуумного вимикача серії ВР1. Конструкція вимикача містить електромагніт з

двома котушками: одна для вмикання вимикача, інша для вимикання. Котушку вмикання імітує світлодіод VD1 «вкл», а котушку вимикання – VD2 «вимк». Включення та відключення світлодіодів реалізується кнопками SB1 – «вкл» та SB2 – «вимк».

Автоматичне вимикання вимикача при коротких замиканнях реалізує вбудований в схему мікроконтролера аналоговий компаратор. Виводи компаратора для зовнішніх підключень:

– № 7, 8 – напруга живлення процесора;

– № 20, 22 – напруга живлення компаратора;

– № 12 (PD6)(AIN0) – 6-й розряд порту D – опорний вхід компаратора, подається напруга уставки спрацювання;

– № 13 (PD7)(AIN1) – 7-й розряд порту D – сигнальний вхід компаратора подається сигнал від датчика струму. Компаратор спрацьовує, якщо напруга на сигнальному вході більша, ніж на опорному. При спрацюванні компаратора включається світлодіод VD3 спрацювання компаратора і світлодіод VD2 – «вимк», який імітує відключення вимикача.

Схема складена в програмному середовищі Proteus показана на рисунку 2.2.

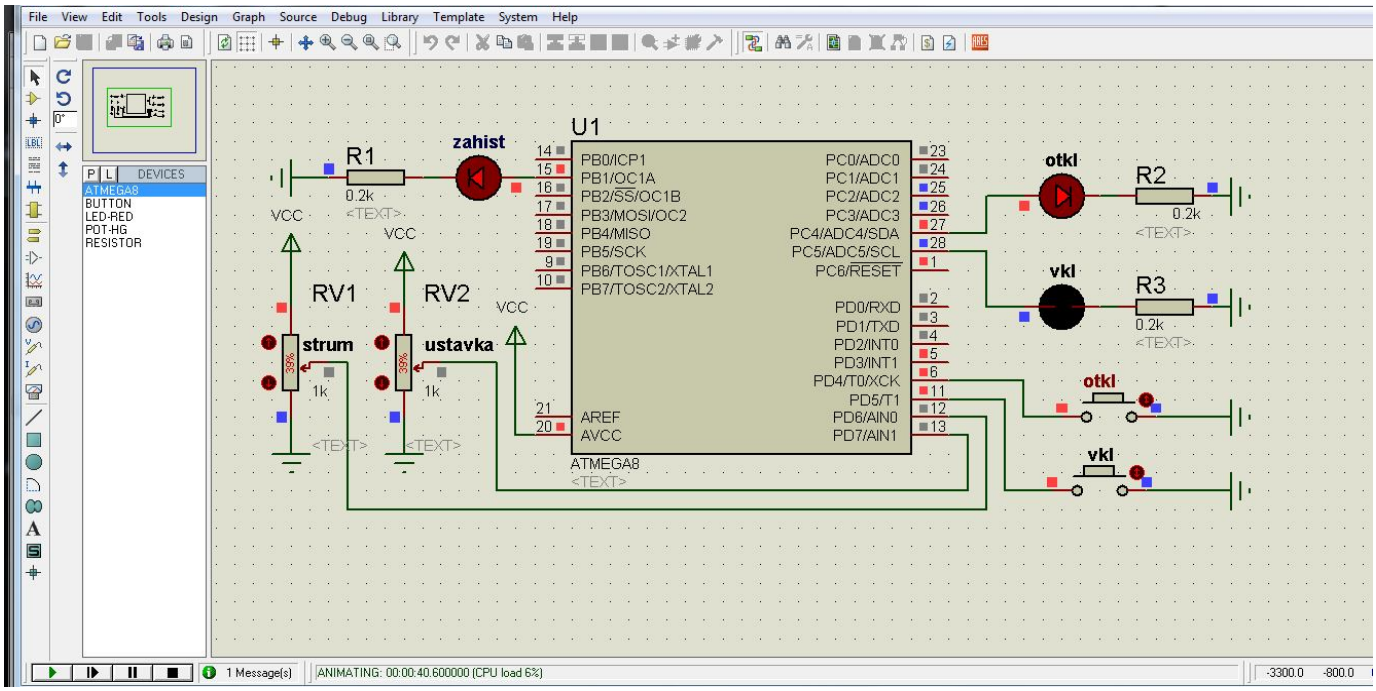

Рисунок 2.2 – Схема лабораторної роботи

Послідовність складання схеми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1 з наступними особливостями. В даній лабораторній роботі використовуються два нових схемних елемента: pot-hg – змінний резистор та button – кнопка. Для складання бібліотеки елементів у віконці Keywords набрати ключові слова: «pot-hg», «button». Всі інші дії виконуються аналогічно.

II Набір тексту програми та її трансляція в програмному середовищі AVR Studio

Зазвичай програми складаються з основної програми та підпрограм. До основної програми відносимо команди, які виконуються в нескінченому циклі, тобто, команди опитування стану кнопок: натиснута чи ні. До підпрограми відносимо команду, яка виконуються викликом з таблиці переривань при спрацюванні компаратора. Виходячи з цього алгоритм основної програми має вигляд на рисунку 2.3.

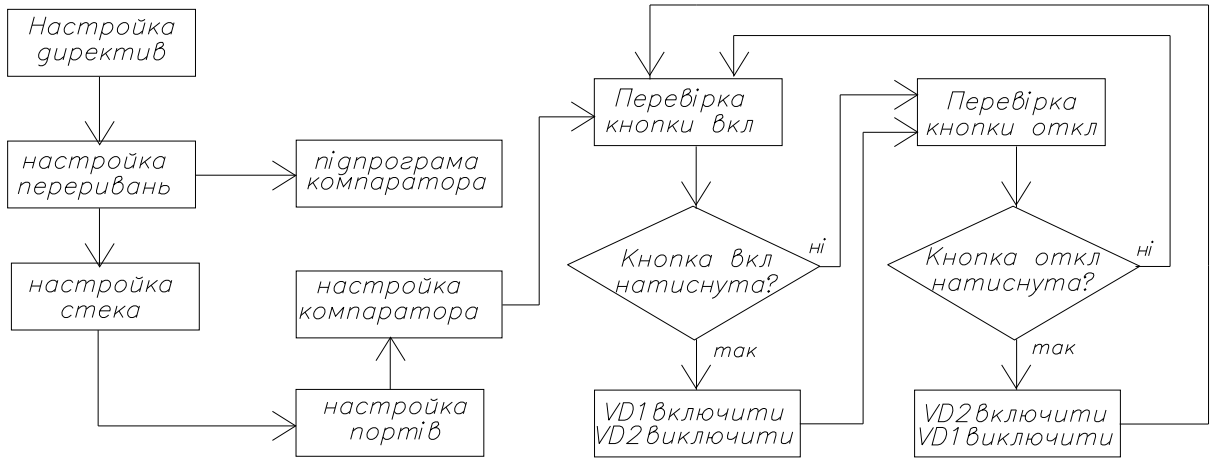

Рисунок 2.3 – Алгоритм основної програми для роботи схеми

Для кожного блоку алгоритму складаються відповідні команди. Для першого блоку алгоритму під назвою «Настройка директив» перелік директив

зведений в таблицю 2.1.

Таблиця 2.1 – Перелік директив Асемблера

Директиви	Коментарі до директив
.device ATmega8	директива вказує, що програма складена для ATmega8
.include “m8def.inc”	директива підключає файл з бібліотеки AVR studio
.cseg	директива вказує про запис команд в ROM-пам’ять

Далі за алгоритмом програмується таблиця переривань. При використанні в програмі викликів підпрограм з таблиці переривань в 1-у комірку ROM-пам’яті мікроконтролера з нульовою адресою завжди завантажують команду безумовного переходу на мітку основної програми. Інші комірки таблиці переривань зарезервовані за периферійними пристроями, в які завантажують команди безумовного переходу на мітки відповідних підпрограм.

При спрацюванні компаратора основна програма зупиняється і викликається підпрограма, що закріплена за компаратором. Для виклику підпрограми необхідно в 17-у комірку таблиці переривань з адресою 0x0010, що закріплена за компаратором, завантажити команду переходу rjmp otsechka на підпрограму компаратора. Програмування таблиці переривань зведене в таблицю 2.2.

Таблиця 2.2 – Програмування таблиці переривань

Директиви та команди	Коментарі
.org 0x0000	директива вказує адресу 1-ї комірки в ROM-пам’яті
rjmp stek	команда переходу в 1-й комірці ROM-пам’яті на мітку «stek»
.org 0x0010	директива вказує адресу 17-ї комірки для компаратора
rjmp otsechka	команда в 17-у комірці для переходу на мітку «otsechka»

Далі за алгоритмом програмується блок «налаштування стеку», оскільки в програмі використовується підпрограма. Для програмування стеку в регістр SP (покажчик стеку) завантажують адресу двох останніх 8-розрядних комірок в RAM-пам’яті, в які при виклику підпрограм завантажуються старший і

молодший байти 16-тирозрядної адреси комірки ROM-пам’яті з наступною командою програми. Результати програмування стеку зведені в таблицю 2.3.

Таблиця 2.3 – Програмування стека

Команди	Коментарі
stek:	мітка основної програми
ldi r16, high(RAMEND)	завантаження в r16 адреси останньої комірки RAM
out SPH, r16	пересилка адреси в старший регістр показчика стека
ldi r16, low(RAMEND)	завантаження адреси передостанньої комірки RAM
out SPL, r16	пересилка адреси в молодший регістр показчика стека

Далі за алгоритмом необхідно програмувати блок «налаштування портів». Зі схеми зовнішніх підключень мікроконтролера видно, що до порту D підключені кнопки керування світлодіодами, отже, порт необхідно програмувати на вхід. До портів B і C підключені світлодіоди, отже, порти необхідно програмувати на вихід.

Необхідно відзначити, що при подачі живлення на мікроконтролер у всіх регістрах завантажуються логічні нулі, тобто, при включенні живлення всі порти налаштовані на вхід. Отже, нема необхідності програмувати порт D на вхід, але до розрядів pd4 і pd5 порту D з кнопками керування потрібно підключити внутрішні резистори. Результати програмування портів на вхід та вихід зведені в таблицю 2.4.

Таблиця 2.4 – Програмування портів на вхід та вихід

Перелік команд	Коментарі до команд
ldi r16, 0b00011000	завантаження одиниць в 4-й і 5-й розряди регістра r16
out portd, r16	підключення внутрішніх резисторів в розряди порту D
ldi r16, 0b00110000	завантаження робочого регістра r16 двійковим кодом
out ddrc, r16	програмування розрядів 4,5 порту C на вихід
ldi r16, 0b00000010	завантаження робочого регістра r16 двійковим кодом
out ddrb, r16	програмування розряду 1 порту B на вихід, інші на вхід

Далі за алгоритмом необхідно програмувати блок «регістр керування компаратором». Аналоговий компаратор має два входи: сигнальний та опорний.

На сигнальний вхід підключається вихід з датчика струму. На опорний вхід подається уставка спрацювання компаратора. При перевищенні напруги сигнального входу над опорним на виході компаратора з'являється логічна «1». При цьому переривається основна програма і викликається підпрограма компаратора. Програмування компаратора полягає у встановленні відповідних біт в розряди периферійного регістра ACSR, який керує роботою компаратора. На основі аналізу призначення розрядів регістра та з урахуванням принципу дії компаратора в розряди регістра ACSR завантажити біти згідно таблиці 2.5.

Таблиця 2.5 – Завантаження розрядів регістра ACSR

Номер розряду	7	6	5	4	3	2	1	0
біти	0	0	0	0	1	0	1	1

Завантаження комбінації біт в регістр ACSR реалізується командами згідно таблиці 2.6.

Таблиця 2.6 – Програмування компаратора

Перелік команд	Коментарі
ldi r16, 0b00001011	завантаження комбінацію біт для регістра ACSR
out ACSR, r16	налаштування регістра ACSR для управління компаратором

Далі за алгоритмом необхідно скласти програму опитування стану кнопок: натиснута чи ні. Необхідно відзначити: до розрядів порту з кнопками програмно підключені внутрішні резистори, через які подається логічна «1». Якщо кнопки не натиснуті, то на вході портів логічна «1». При натисканні кнопки на вході портів логічний «0» і необхідно включати відповідні світлодіоди. Перевірка стану кнопок і, в залежності від результату перевірки, зміна ходу виконання програми реалізується командами умовного переходу. Команди опитування стану кнопок зведені в таблицю 2.7.

Таблиця 2.7 – Команди основної програми опитування стану кнопок

Команди	Коментарі
knopki:	мітка основної програми опитування кнопок
sbic pind,4	пропуск наступної команди при натиснутій кнопки «відкл»
rjmp PC+3	пропуск двох наступних команд
sbi portc,4	включити світлодіод «відкл»
cbi portc,5	відключити світлодіод «вкл»
sbic pind,5	пропуск наступної команди при натиснутій кнопки «вкл»
rjmp PC+3	пропуск двох наступних команд
sbi portc,5	включити світлодіод «вкл»
cbi portc,4	відключити світлодіод «відкл»
rjmp knopki	команда переходу на початок програми – безкінечний цикл

Перелік команд для підпрограми компаратора наведений в таблиці 2.8.

Таблиця 2.8 – Команди підпрограми компаратора

Команди	Коментарі до команд
otsechka:	мітка підпрограми струмової відсічки
push r16	завантаження в стек даних регістру r16
sbi portc,4	включити світлодіод «відкл»
cbi portc,5	відключити світлодіод «вкл»
sbi portb,1	включить світлодіод «відсічка»
ldi r16, 0b00001011	завантаження двійкового коду в регістр r16
out acsr, r16	відновлення даних регістра управління компаратором
pop r16	пересилка зі стеку даних r16 в основну програму
reti	вихід з підпрограми компаратора

Послідовність набору тексту програми наведена в тексті лабораторної роботи № 1. Особливості набору: у віконці Project name клавіатурою набрати на англійській мові «lab2».

III Перевірка правильності роботи схеми

Програмне середовище AVR studio після трансляції створює файл з машинним кодом формату «.hex» для завантаження в ROM-пам’ять мікроконтролера. Для цього відкрити Proteus та зробити подвійний клік лівою кнопкою миші по схемі мікроконтролера. На робочому полі з’являється вікно

Edit Component (рис 2.4). У віконці CKSEL fuses встановити частоту синхрогенератора 1 МГц. Праворуч від напису Program file клікнути на кнопку «відкрита папка», вказати шлях до файлу формату «.hex» і клікнути на кнопку ОК.

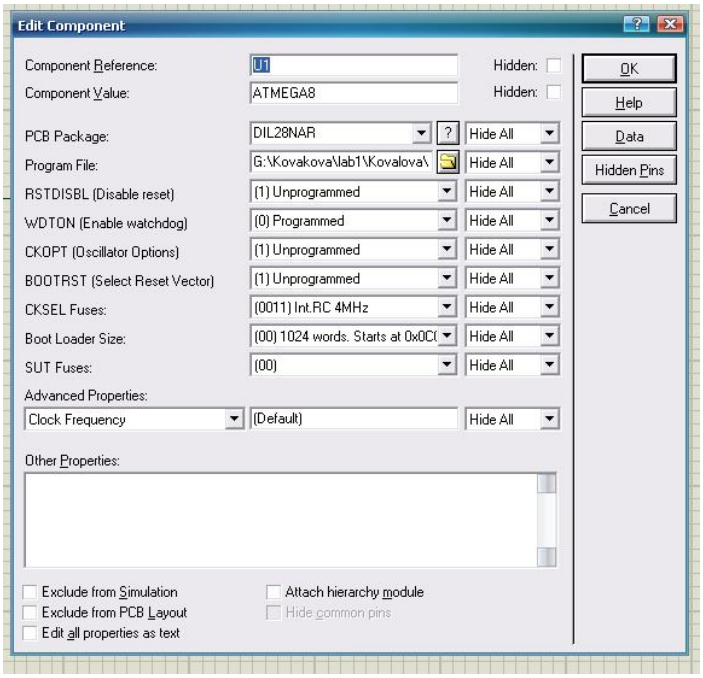

Рисунок 2.4 – Вікно параметрів мікроконтролера

Для перевірки правильності роботи схеми в нижньому лівому кутку робочого столу Proteus розташована панель керування (див. рис. 2.2) з кнопками «пуск», «пауза», «стоп». Лівою кнопкою мишки клікнути на кнопку «пуск».

Натискати почергово на кнопки «вкл» та «відкл» та спостерігати за станом світло діодів «вкл» та «відкл». Натиснути на кнопку «вкл» і збільшувати напругу змінним резистором «strum». При спрацюванні компаратора загоряються світлодіоди «спрацювання компаратора» та «відкл», світлодіод «вкл» гасне.

ПОРЯДОК ОФОРМЛЕННЯ ЗВІТУ

Звіт про виконану лабораторну роботу оформлюється в тому ж зошиті для конспекту лекцій та практичних занять. Вказати номер роботи, її назву та мету. Намалювати реальну принципову електричну схему лабораторної роботи,

переписати команди програми з коментарями, привести письмові відповіді на три контрольних запитання, вказані викладачем.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Пояснити призначення виводів мікросхеми контролера № 20 і № 22.
2. Дати характеристику портам мікроконтролера та пояснити їх призначення.
3. Чому в програмі відсутня команда програмування порту D на вхід?
4. Яким чином відбувається програмування розрядів порту на вихід?
5. Розрахувати величину опорів струмообмежуючих резисторів R2 та R3.
6. Пояснити фізичний зміст та призначення частоти синхрогенератора.
7. Пояснити принцип роботи аналогового компаратора.
8. Пояснити призначення таблиці переривань.
9. Пояснити, що відбувається з програмою при спрацюванні компаратора.
10. Пояснити призначення стеку.
11. Пояснити необхідність програмного підключення внутрішніх резисторів до портів з підключеними кнопками.
12. Скласти послідовність завантаження програми в постійну пам’ять.
13. Прокоментувати команди в рядках № 1 та № 2 в таблиці 2.4.
14. Прокоментувати команди в рядках № 3 та № 4 в таблиці 2.4.
15. Прокоментувати команди в рядках № 5 та № 6 в таблиці 2.4.
16. Пояснити призначення команд в таблиці 2.3.
17. Пояснити у чому полягає програмування компаратора.
18. Пояснити призначення команд «push r16» і «pop r16».

ЛАБОРАТОРНА РОБОТА № 3

ПРОГРАМУВАННЯ АНАЛОГО-ЦИФРОВОГО ПЕРЕТВОРЮВАЧА

Мета роботи – отримати практичні навички програмування аналого-цифрового перетворювача та перевірка правильності роботи в програмних середовищах AVR Studio та Proteus.

КОРОТКА ІНФОРМАЦІЯ ДО ВИКОНАННЯ РОБОТИ

Мікроконтролер серії ATmega8 має 10-тирозрядний аналого-цифровий перетворювач (АЦП), який перетворює величину аналогової напругу в 10-тирозрядне двійкове число. Зміст аналого-цифрового перетворення полягає у вимірюванні напруги в цифровому коді, що потребує впровадження термінів «аналогова шкала», «цифрова шкала» і «поділка шкали». Кількість поділок аналогової $N_{АН}$ і цифрової $N_{АЦП}$ шкали однакове і визначаються за формулою $N_{АЦП} = N_{АН} = 2^p-1 = 2^{10} - 1 = 1023$ поділки, де p – кількість розрядів двійкового числа, тобто, розрядність АЦП. Якщо максимальна вхідна напруга АЦП дорівнює 5 В, то ціна поділки аналогової шкали дорівнює $Ц_{АН} = U_{макс} / N_{АН} = 5/1023 = 0,0049$ В. Програмування АЦП полягає в установці комбінації біт в розрядах регістрів управління під назвою ADCSRA і ADMUX.

ПРОГРАМА РОБОТИ

1. Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.
2. Вивчення тексту програми, набір та її перевірка в програмному середовищі AVR Studio.
3. Перевірка правильності роботи програми.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

І Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.

Схема лабораторної роботи показана на рисунку 3.1.

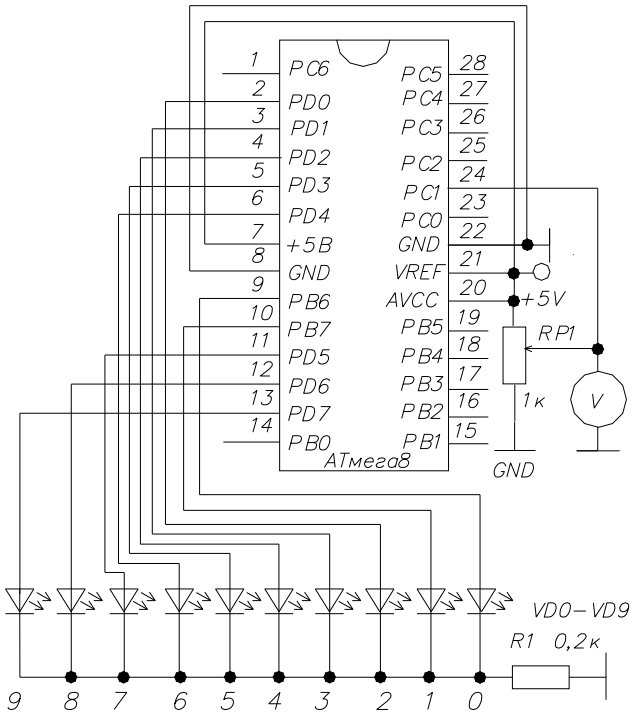

Рисунок 3.1 – Схема лабораторної роботи

Вивід мікросхеми № 24 є другим каналом АЦП, до якого підключений середній вивід змінного резистора RP1. Вивід № 20 – це напруга живлення АЦП, вивід № 21 призначений для підключення опорної напруги для АЦП, яка визначає максимальну вхідну напругу при перетворенні аналогової напруги в цифрову. Зі схеми видно, що в якості опорної напруги використовується напруга живлення мікроконтролера. До виводів портів D і B підключено 10 світлодіодів, стан яких включено і відключено являє собою 10-розрядний двійковий код, тобто, на виході буде з’являтися двійковий код вхідного аналогового сигналу. Виводи 9, 10 порту B це молодші розряди двійкового коду числа. Виводи 2–13 порту d це 8 старших розрядів двійкового коду числа.

Аналоговий сигнал напруги від змінного резистора підключений до виводу № 24, який є 1-м розрядом порту С. Величина аналогової напруги вимірюється вольтметром. Схема лабораторної роботи може бути складена з реальних елементів або з віртуальних елементів в програмному середовищі Proteus показана на рисунку 3.2.

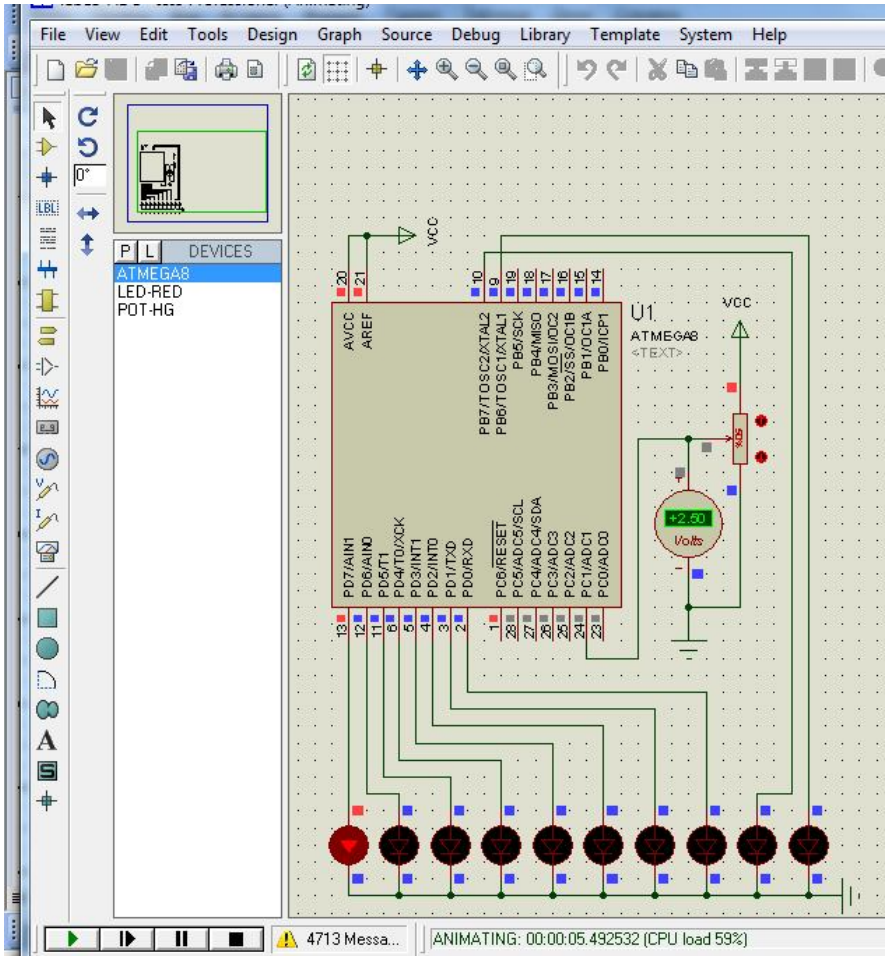

Рисунок 3.2 – Схема лабораторної роботи

Послідовність складання схеми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1 з наступними особливостями. В даній лабораторній роботі використовується новий схемний елемент – вольтметр. На вертикальній боковій панелі інструментів клікнути на ярлик Virtual instrument mood. Відкривається перелік приладів, в якому лівою кнопкою миші вибрати «DC Voltmeter» і клікнути на робочому полі в заданому місці.

ІІ Набір тексту програми та її перевірка в програмному середовищі AVR Studio

Алгоритм програми АЦП показаний на рисунку 3.3.

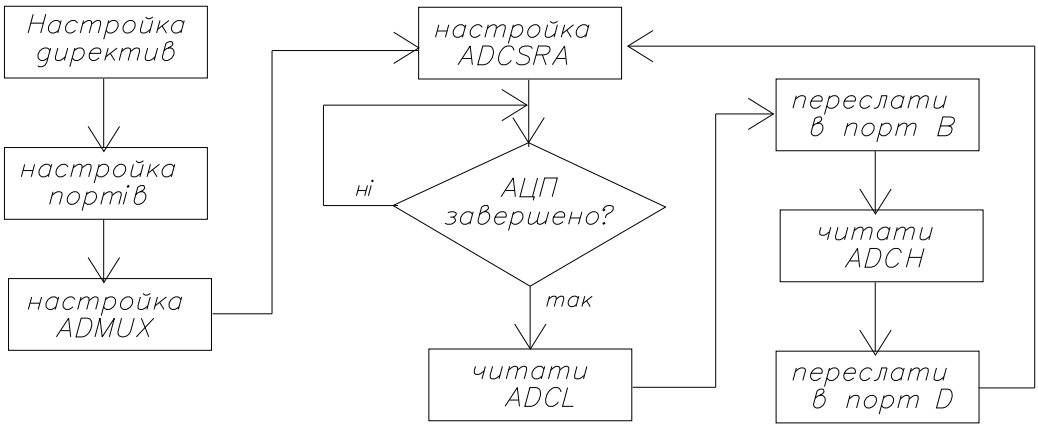

Рисунок 3.3 – Алгоритм програми аналого-цифрового перетворення

Для кожного окремого блоку алгоритму складаються відповідні команди. Для першого блоку алгоритму під назвою «Налаштування директив» перелік директив зведений в таблицю 3.1.

Таблиця 3.1 – Перелік директив Асемблера

Директиви	Коментарі
.device Tmega8	директива вказує, що програма складена для ATmega8
.include “m8def.inc”	директива підключає файл з бібліотеки AVR studio
.cseg	директива вказує про запис команд в ROM-пам’ять
.org 0x0000	директива вказує адресу 1-ї комірки ROM-пам’яті

Далі за алгоритмом програмується блок «налаштування портів». Зі схеми зовнішніх підключень мікроконтролера видно, що до портів D і B підключені світлодіоди, отже, порти потрібно налаштувати на вихід. До порту C підключений реостат, отже, порт потрібно налаштувати на вхід.

Необхідно відзначити, що при подачі живлення на мікроконтролер у всіх регістрах завантажені логічні нулі, тобто, при включенні живлення всі порти налаштовані на вхід. Отже порт C налаштований на вхід і спеціальної команди

програмування не потребує. Для програмування портів D і B на вихід в регістри напрямку ddrd і ddrb завантажуються логічні одиниці. Результати програмування портів на вихід зведені в таблицю 3.2.

Таблиця 3.2 – Програмування портів на вихід

Перелік команд	Коментарі
ser r16	завантаження одиниць в розряди регістра r16
out ddrd, r16	програмування порту D на вихід
out ddrb, r16	програмування порту B на вихід

Далі за алгоритмом необхідно програмувати блок «настройка ADMUX». Призначення розрядів регістра ADMUX: розряди 0, 1, 2, 3 – комбінація біт в розрядах програмує вибір каналу для вимірювань. АТмега8 може перетворити 6 каналів аналогових напруг. Для цього в мікроконтролер вбудований мультиплексор, який може перемикати вхід АЦП до будь-якого з 6-и виводів мікроконтролера, звичайно по черзі, а не одночасно.

Для програмування каналу для АЦП використовуємо таблицю 3.3.

Таблиця 3.3 – Комбінація біт в розрядах регістру ADMUX для вибору входів для перетворення

Розряд 3	Розряд 2	Розряд 1	Розряд 0	Виводи корпуса DIP-28	Розряди порта С	Кількість розрядів в коді
0	0	0	0	23(ADC0)	PC0	10
0	0	0	1	24(ADC1)	PC1	10
0	0	1	0	25(ADC2)	PC2	10
0	0	1	1	26(ADC3)	PC3	10
0	1	0	0	27(ADC4)*	PC4	8
0	1	0	1	28(ADC5)*	PC5	8

У розряд 4-й завантажуємо «0», так як не використовується.

Розряд 5-й – визначає два варіанти запису 10-тирозрядного двійкового числа в регістрі даних після завершення перетворення: при завантаженні «0» 8 молодших розрядів числа завантажуються в молодший регістр даних ADCL і два старших розряди завантажуються в старший регістр

даних ADCH. При завантаженні «1» 8 старших розрядів числа завантажуються в регістр ADCH і два молодших розряду – в ADCL. При зчитуванні 10-розрядного числа треба спочатку читати дані з регістра ADCL, а потім з ADCH.

Таблиця 3.4 – Завантаження числа в регістри даних при «1» в 5-м розряді регістра ADMUX

ADCH								ADCL							
1	1	0	0	1	1	0	1	0	1						

Якщо в 5-му розряді завантажити «0», то комбінація зміститься вправо.

Максимальна величина вхідної напруги АЦП задається величиною опорної напруги. Варіанти завдання опорного напруги визначають розряди 6, 7 регістра ADMUX. Розряди 7, 6 – комбінація біт визначає тип джерела опорної напруги (ДОН): 0–0 – підключається зовнішнє ДОН на U < 5 В до виводів AREF і «земля»; 0–1 – ДОН є напруга живлення, тобто, з'єднати між собою висновки AVCC, AREF, VCC і GND, AGND; 1–0 – резерв; 1–1 – підключає внутрішній ДОН на U = 2,56 В.

Таблиця 3.5 – Вибір розрядів регістра ADMUX

7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	1
Опорна напруга від джерела живлення		Зчитувати 8 старших розрядів числа з регістра ADCH і 2 молодших – з ADCL	Не використовується	Вхід АЦП підключений до порту PC1			

Команди для програмування регістру керування ADMUX зведені в таблицю 3.6.

Таблиця 3.6 – Програмування регістру керування ADMUX

Перелік команд	Коментарі
ldi r16, 0b01100001	завантаження коду
out ADMUX, r16	настройка регістра ADMUX

Далі за алгоритмом необхідно програмувати блок «настройка ADCSRA».

Призначення розрядів регістра ADCSRA: Комбінація біт в розрядах 0, 1, 2 задає коефіцієнт подільника частоти синхрогенератора мікроконтролера, щоб частота вимірів АЦП була в діапазоні 50-200 кГц. Біт в 3-му розряді дозволяє або забороняє переривання по закінченню вимірювання. 4-й розряд змінює біт по завершенню вимірювання. 5-й розряд програмує режим одноразового або безперервного перетворення. 6-й розряд запускає режим АЦП. 7-й розряд включає живлення АЦП.

Для зручності програмування регістра ADCSRA і щоб не помилитися при завантаженні біт в відповідні розряди використовується таблиця 3.7.

Таблиця 3.7 – Призначення розрядів регістра ADCSRA

7	6	5	4	3	2	1	0
1	1	1	0	0	1	0	0
Живлення АЦП включено	Запуск АЦП	Безперервне перетворення	Змінюється на «1» після завершення АЦП	Переривання заборонені	Коефіцієнт ділення частоти синхрогенератора = 16. Частота =1000/16=62,5 кГц		

Команди для програмування регістру керування ADCSRA зведені в таблицю 3.8.

Таблиця 3.8 – Програмування регістру керування ADCSRA

Перелік команд	Коментарі
adc1:	мітка для нескінченого циклу АЦП
ldi r16, 0b11100100	завантаження коду
out ADCSRA, r16	налаштування регістра ADCSRA

Далі за алгоритмом необхідно скласти програму опитування завершення АЦП.

Команди опитування стану кнопок зведені в таблицю 3.9.

Таблиця 3.9 – Команди опитування завершення АЦП та пересилки даних

Команди	Коментарі
opros:	мітка для перевірки завершення АЦП
sbis ADCSRA, 4	пропуск наступної команди, якщо АЦП завершено, тобто, в 4-му розряді регістра ADCSRA встановлена «1»
rjmp opros	перехід на мітку «opros», якщо АЦП не завершено, тобто, в 4-му розряді регістра ADCSRA встановлений «0»
in r16, ADCL	пересилка даних з молодшого сигнального регістра
out portb, r16	пересилка даних на вихідний порт portb
in r16, ADCH	пересилка даних зі старшого сигнального регістра
out portd, r16	пересилка даних на вихідний порт portd
rjmp adc1	нескінчений цикл

Команди програми набрати в текстовому редакторі Word. Набраний текст зберегти на диску D. Послідовність набору тексту програми наведена в тексті лабораторної роботи №1. Особливості набору: У віконці Project name клавіатурою набрати на англійській мові «lab3».

III Перевірка правильності роботи програми

Для перевірки роботи схеми необхідно в пам’ять мікроконтролера завантажити програму, отриману в AVR studio. Послідовність завантаження програми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1.

Для перевірки роботи схеми в нижньому лівому кутку робочого столу розташована панель керування симуляцією з кнопками «пуск», «пауза», «стоп».

Згідно таблиці 3.10 по вольтметру встановити величину напруги для свого варіанту та виконати відповідні розрахунки

Таблиця 3.10 – Розрахунок похибок перетворення

Напруга, В	Двійкове число АЦП	Кількість поділок	Ціна поділки	Розрахункове значення, В	Похибка абсолютна, В	Похибка відносна, %
0,5	0001100110	102	0,0049	0,4998	0,0002	0,04
1,45						
2,35						
3,55						
3,8						
4,65						

ПОРЯДОК ОФОРМЛЕННЯ ЗВІТУ

Звіт про виконану лабораторну роботу оформлюється в тому ж зошиті для конспекту лекцій та практичних занять. Вказати номер роботи, її назву, мету та порядок виконання. Намалювати реальну принципову електричну схему лабораторної роботи, переписати команди програми з коментарями, привести письмові відповіді на три контрольних запитання, вказані викладачем. Провести розрахунок похибки перетворення аналогічно таблиці 3.10.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Пояснити призначення команд в таблиці 3.1.
2. Пояснити у чому полягає програмування АЦП.
3. Пояснити у чому полягає зміст команд в таблиці 3.4.
4. Пояснити, яким чином в мікроконтролері реалізується можливість перетворення в цифровий код шести каналів аналогових напруг.
5. Дати характеристику розрядам 0, 1, 2, 3 регістра керування ADMUX.
6. Дати характеристику розрядам 0, 1, 2 регістра керування ADCSRA.
7. Дати характеристику термінам «аналогова шкала», «цифрова шкала» і «поділка шкали».
8. Пояснити призначення команди «rjmp adc1» в таблиці 3.9.
9. Дати характеристику регістрів ADCH ADCL.
10. Дати характеристику розрядам 0, 1, 2, регістра керування ADCSRA.
11. Дати характеристику розрядам 3, 4 регістра керування ADCSRA.
12. Дати характеристику розрядам 5, 6 регістра керування ADCSRA.
13. Дати характеристику розряду 5 регістра керування ADMUX.
14. Дати характеристику розрядам 6, 7 регістра керування ADMUX.
15. Пояснити призначення команд в таблиці 3.2.
16. Скласти формулу для розрахунку опору резистора R1.

ЛАБОРАТОРНА РОБОТА № 4

ПРОГРАМУВАННЯ МАКСИМАЛЬНОГО СТРУМОВОГО ЗАХИСТУ

Мета роботи – отримати практичні навички програмування максимального струмового захисту та перевірка правильності роботи в програмних середовищах AVR Studio та Proteus.

КОРОТКА ІНФОРМАЦІЯ ДО ВИКОНАННЯ РОБОТИ

Максимальний струмовий захист (МСЗ) згідно «Правил улаштування електроустановок» (ПУЕ) встановлюють для захисту ліній і трансформаторів від струмів перевантаження та в якості другого ступеню струмового захисту для дальнього та ближнього резервування захистів від коротких замикань. Ближнє резервування це спрацювання МСЗ в разі відмови захисту від коротких замикань. Дальнє резервування захисту від коротких замикань лінії 10 кВ це спрацювання МСЗ трансформатора 110/10 кВ в разі його відмови або несправності вакуумного вимикача.

Максимальний струмовий захист завжди використовують з витримкою часу при спрацюванні, отже, необхідно задіяти таймер, вбудований в мікроконтролер.

Для побудови МСЗ використовується аналого-цифровий перетворювач (АЦП), який перетворює напругу з датчика струму в двійкове число і далі порівнює з двійковим числом уставки спрацювання МСЗ.

ПРОГРАМА РОБОТИ

1. Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.
2. Вивчення тексту програми, набір та її перевірка в програмному середовищі AVR Studio.
3. Перевірка правильності роботи програми.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

І Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.

Схема лабораторної роботи показана на рисунку 4.1.

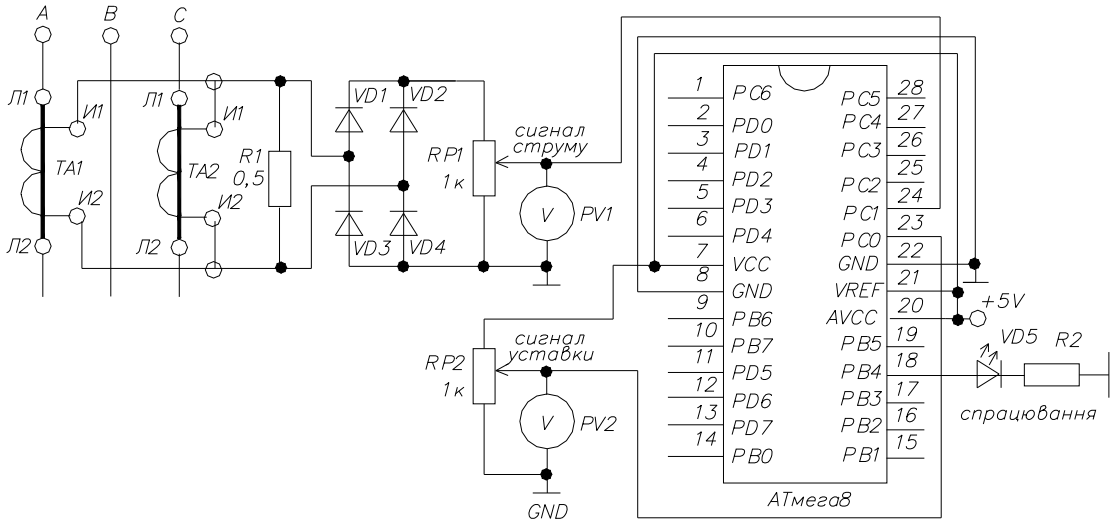

Рисунок 4.1 – Схема лабораторної роботи

На вхід АЦП підключений середній вивід змінного резистора RP1. Вивід 20 – це напруга живлення АЦП, вивід 21 – це вивід для підключення опорної напруги для АЦП, яка визначає максимальну вхідну напругу для перетворення аналогової напруги в цифрову. Зі схеми видно, що в якості опорної напруги використовується напруга живлення мікроконтролера

Аналоговий сигнал напруги від змінного резистора RP1 підключений до виводу 24, який є 1-м розрядом порту С. Величина аналогової напруги вимірюється вольтметром. Нижче видно світлодіод «signal MTZ» для сигналізації про спрацьовування МТЗ. При перевантаженні струм в лінії більше струму уставки, спрацьовує МТЗ і включається світлодіод «signal MTZ», який сигналізує про перевантаження. Схема лабораторної роботи може бути складена з реальних елементів або з віртуальних елементів в програмному середовищі Proteus показана на рисунку 4.2.

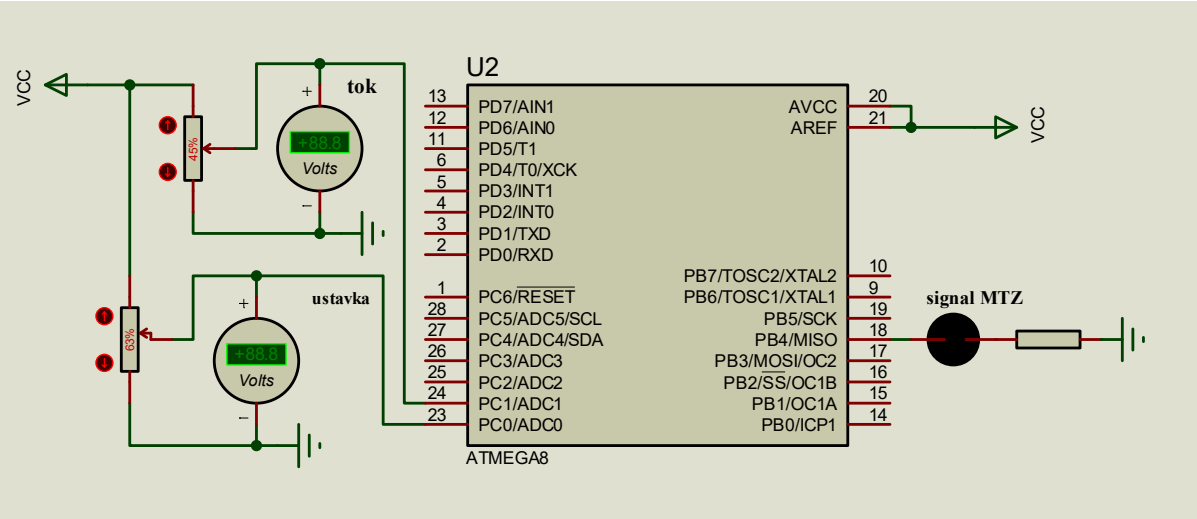

Рисунок 4.2 – Схема лабораторної роботи в програмному середовищі Proteus

Послідовність складання схеми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1 з наступними особливостями. В даній лабораторній роботі використовується новий схемний елемент «вольтметр». На вертикальній боковій панелі інструментів клікнути на ярлик «Virtual instrument mood». Відкривається перелік приладів, в якому лівою кнопкою «мишки» вибрати «DC Voltmeter» і клікнути на робочому полі в заданому місці.

II Набір тексту програми та її перевірка в програмному середовищі AVR Studio. Алгоритм підпрограми МСЗ представлений на рисунку 4.3.

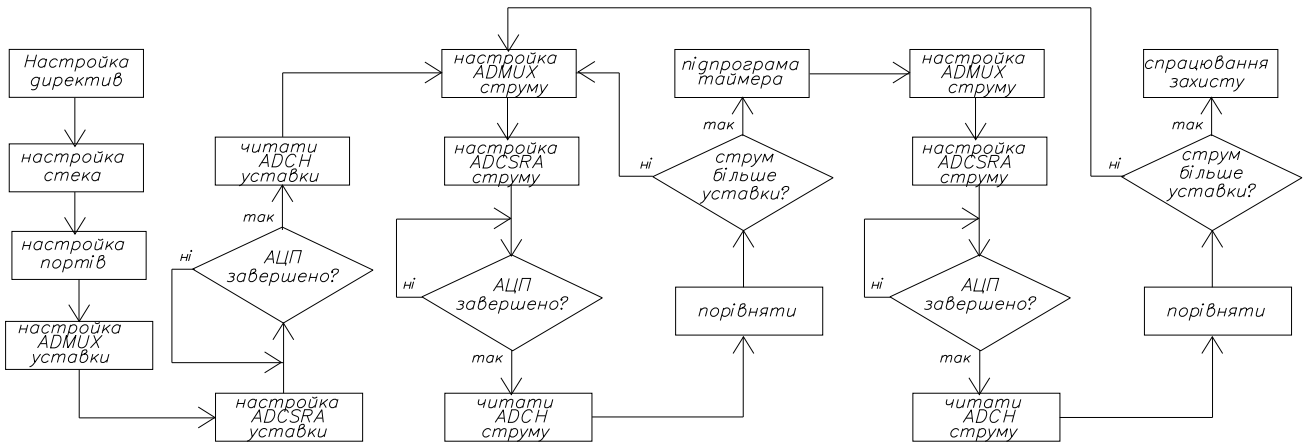

Рисунок 4.3 – Алгоритм підпрограми МСЗ

Для кожного окремого блоку алгоритму складаються відповідні команди.

Для першого блоку алгоритму під назвою «Настройка директив» перелік директив та програмування стека зведений в таблицю 4.1.

Таблиця 4.1 – Перелік директив Асемблера та програмування стека

Директиви	Коментарі
.device ATmega8	директива вказує, що програма складена для ATmega8
.include “m8def.inc”	директива підключає файл з бібліотеки AVR studio
.cseg	директива вказує про запис команд в ROM-пам’ять
.org 0x0000	директива вказує адресу 1-ї комірки ROM-пам’яті
ldi r16, high(RAMEND)	завантаження в r16 адреси останньої комірки RAM
out SPH, r16	пересилка адреси в старший регістр показчика стека
ldi r16, low(RAMEND)	завантаження адреси передостанньої комірки RAM
out SPL, r16	пересилка адреси в молодший регістр показчика стека

Далі за алгоритмом програмується блок «налаштування портів». Зі схеми зовнішніх підключень мікроконтролера видно, що до порту В підключений світлодіод для сигналізації спрацювання МСЗ, отже, порт потрібно налаштувати на вихід. В порту С використовується АЦП, отже, порт потрібно налаштувати на вхід. Необхідно відзначити, що при подачі живлення на мікроконтролер всі порти налаштовані на вхід, тобто, не потребують спеціальної команди програмування на вхід. Для програмування порту В на вихід в регістр напрямку ddrb завантажуються логічні «одиниці». Результати програмування портів на вихід зведені в таблицю 4.2.

Таблиця 4.2 – Програмування портів на вихід

Перелік команд	Коментарі
ser r16	завантаження «одиниць» в розряди регістра r16
out ddrb, r16	програмування порту В на «вихід»

Далі за алгоритмом необхідно програмувати блок «настройка ADMUX». Призначення розрядів регістра ADMUX: розряди 0, 1, 2, 3 – комбінація біт в розрядах програмує вибір каналу з 6 можливих. Для цього в мікроконтролер вбудований мультиплексор, який може програмно перемикати вхід АЦП до будь-якого з 6-и каналів з використанням таблиці 4.3.

Таблиця 4.3 – Комбінація біт в розрядах ADMUX для вибору каналів

Розряд 3	Розряд 2	Розряд 1	Розряд 0	Виводи корпусу	Розряди порту С	Кількість розрядів в коді
0	0	0	0	23(ADC0)	PC0	10
0	0	0	1	24(ADC1)	PC1	10
0	0	1	0	25(ADC2)	PC2	10
0	0	1	1	26(ADC3)	PC3	10
0	1	0	0	27(ADC4)*	PC4	8
0	1	0	1	28(ADC5)*	PC5	8

Вибір каналу виконується перед пуском АЦП. На час АЦП вибір каналів заборонений і можливий лише після завершення АЦП, коли в 4-й розряд регістра ADCSRA апаратно завантажується «1». Використання режиму перемикання каналів рекомендовано для однократного режиму АЦП.

У розряд 4-й завантажуємо «0», оскільки не використовується. Розряд 5-й – визначає два варіанти запису 10-тирозрядного двійкового числа в регістрі даних після завершення перетворення: при завантаженні «0» 8 молодших розрядів числа завантажуються в молодший регістр даних ADCL і два старших розряди завантажуються в старший регістр даних ADCH. При завантаженні «1» 8 старших розрядів числа завантажуються в регістр ADCH і два молодших розряду – в ADCL.

Якщо не зчитувати два молодших розряди числа з молодшого регістра ADCL, то абсолютна похибка вимірювання напруги дорівнює добутку ціни поділки на кількість поділок $\Delta = N_{МСЗ} \cdot Ц_{под}$. При цьому відносна похибка не перевищує допустимої до 10%, тому можна не враховувати два молодших розряди і зчитувати дані лише зі старшого розряду. Вибір типу джерела опорної напруги визначають розряди 6, 7 регістра ADMUX для вибору максимальної вхідної напруги. Вибір розрядів регістра ADMUX зведений в таблицю 4.4.

Таблиця 4.4 – Вибір розрядів регістра ADMUX уставки

7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	0
Опорна напруга від джерела живлення		Зчитувати 8 старших розрядів числа з регістра ADCH і 2 молодших – із ADCL	Не використо-вується	Вхід АЦП підключений до порту PC0			

Команди для програмування регістру ADMUX зведені в таблицю 4.5.

Таблиця 4.5 – Програмування регістру керування ADMUX уставки

Перелік команд	Коментарі
ldi r16, 0b01100000	загрузка коду
out ADMUX , r16	настройка регістра ADMUX

Далі за алгоритмом необхідно програмувати блок «настройка ADCSRA». Призначення розрядів регістра ADCSRA: комбінація біт в розрядах 0, 1, 2 задає коефіцієнт подільника частоти синхрогенератора, щоб частота вимірів АЦП була в діапазоні 50–200 кГц. Біт в 3-м розряді дозволяє або забороняє переривання по закінченню вимірювання. 4-й розряд встановлює «1» по завершенню вимірювання. 5-й розряд програмує режим одноразового або безперервного перетворення. 6-й розряд запускає режим АЦП. 7-й розряд включає живлення АЦП. Для зручності вибору бітів в розрядах регістра ADCSRA використовується таблиця 4.6.

Таблиця 4.6 – Призначення розрядів регістра ADCSRA уставки

7	6	5	4	3	2	1	0
1	1	0	0	0	1	0	0
Живлення АЦП включено	Запуск АЦП	Однократне перетворення	Встановлюється «1» після завершення АЦП	Переривання заборонені	Коефіцієнт ділення частоти синхрогенератора = 16. Частота =1000/16 = 62,5 кГц		

Команди для програмування регістру керування ADCSRA уставки зведені в таблицю 4.7.

Таблиця 4.7 – Програмування регістру керування ADCSRA уставки

Перелік команд	Коментарі
ldi r16, 0b11000100	завантаження коду
out ADCSRA, r16	настройка регістра ADCSRA

Далі за алгоритмом необхідно скласти програму опитування завершення АЦП уставки спрацювання, яке зведене в таблиця 4.8.

Таблиця 4.8 – Команди опитування завершення АЦП та пересилки даних

Команди	Коментарі
opros1:	мітка для перевірки завершення АЦП
sbis ADCSRA, 4	пропуск наступної команди, якщо АЦП завершено, тобто, в 4-му розряді регістра ADCSRA встановлена «1»
rjmp opros1	перехід на мітку «opros», якщо АЦП не завершено, тобто, в 4-му розряді регістра ADCSRA встановлений «0»
in r17, ADCH	двійковий код уставки спрацювання МСЗ

Далі за алгоритмом необхідно зчитувати двійковий код поточного струму, який поступає на вхід PC1. Програмування регістрів керування ADMUX і ADCSRA для аналогового сигналу струму виконуються аналогічно для уставки та зведено в таблицю 4.9.

Таблиця 4.9 – Програмування регістрів ADMUX і ADCSRA струму

Перелік команд	Коментарі
strum:	мітка для умовного переходу
ldi r16, 0b01100001	загрузка коду
out ADMUX, r16	настройка регістра ADMUX
ldi r16, 0b11000100	загрузка коду
out ADCSRA, r16	настройка регістра ADCSRA

Далі за алгоритмом необхідно скласти програму опитування завершення АЦП поточного струму, яке зведене в таблиця 4.10.

Таблиця 4.10 – Команди опитування завершення АЦП струму та зчитування

Команди	Коментарі
opros2:	мітка для перевірки завершення АЦП струму
sbis ADCSRA, 4	пропуск наступної команди, якщо АЦП завершено, тобто, в 4-му розряді регістра ADCSRA встановлена «1»
rjmp opros2	перехід на мітку «opros», якщо АЦП не завершено, тобто, в 4-му розряді регістра ADCSRA встановлений «0»
in r18, ADCH	двійковий код поточного струму в регістрі r18

Далі за алгоритмом необхідно порівняти код струму з кодом уставки. Якщо код струму більше коду уставки то необхідно викликати підпрограму витримку часу, якщо менше, то повторити вимірювання струму. Команди для цієї ділянки

алгоритму зведені в таблицю 4.11.

Таблиця 4.11 – Порівняння струму з уставкою

Команди	Коментарі
cp r18, r17	порівняти поточний струм з уставкою спрацювання
brlo strum	якщо r18-r17<0, то повторити вимірювання струму
rcall timer	якщо r18-r17>0, то виклик підпрограми витримки часу

Зміст команди «cp r18, r17» полягає у порівнянні шляхом віднімання з числа в регістрі r18 числа в регістрі r17. Якщо різниця менше «нуля», то число в регістрі r18 менше числа в регістрі r17, тобто, поточний струм менше струму уставки. Тоді згідно алгоритму потрібно повторити вимірювання. Для цього використовується команда «brlo strum» умовного переходу за умовою «менше». Якщо струм більше уставки, то викликається підпрограма часової затримки. Після часової затримки необхідно знову перевірити величину поточного струму, оскільки струм може зменшитися, наприклад, у випадку пуску високовольтного двигуна, коли пусковий струм виростає в 5-6 більше разів номінального, а після закінчення пуску струм зменшується. Якщо після витримки часу струм зменшився, то згідно алгоритму повторити вимірювання струму.

Таблиця 4.12 – Підпрограма таймера

Команди	Коментарі
timer:	мітка підпрограми таймера
push r16	загрузка в стек даних регістра r16 з основної програми
ldi r16, 0b00000101	настройка на коефіцієнт ділення частоти на 64
out TCCR1B, r16	пуск таймера с частотою на вході $10^6/64 = 15625$ Гц
ldi r16, 0x0000	скид лічильних регістрів таймера в «0»
out TCNT1H, r16	
out TCNT1L, r16	
wate:	мітка для переходу за умовою
in r16, TCNT1L	читати молодший байт полічених імпульсів
cpi r16, low(15625)	порівняти молодший байт с уставкою часу
brlo wate	якщо число в r16 менше уставки, йти на мітку wate,
in r16, TCNT1H	якщо рівне або більше, то читати дані зі старшого
cpi r16, high(15625)	порівняти зі старшим байтом уставки
brlo wate	якщо число в r16 менше уставки, йти на мітку wate,
pop r16	пересилка даних r16 зі стека в основну програму
ret	вихід з підпрограми таймера

Якщо струм після витримки часу залишився більше струму уставки, то необхідно ввімкнути світлодіод для сигналізації спрацювання МСЗ

Таблиця 4.13 – Команди для кінцевої ділянки алгоритму

Команди	Коментарі
opros2:	мітка для перевірки завершення АЦП струму
sbis ADCSRA, 4	пропуск наступної команди, якщо АЦП завершено, тобто, в 4-му розряді регістра ADCSRA встановлена «1»
rjmp opros2	перехід на мітку «opros», якщо АЦП не завершено, тобто, в 4-му розряді регістра ADCSRA встановлений «0»
in r18, ADCH	двійковий код поточного струму
cp r18, r17	порівняти поточний струм з уставкою спрацювання
brlo strum	якщо r17-r16 < 0, то повторити вимірювання струму
sbi portb, 4	включити світлодіод для сигналізації спрацювання МСЗ

Команди програми набрати в текстовому редакторі Word. Набраний текст зберегти на диску D. Послідовність набору тексту програми наведена в тексті лабораторної роботи №1. Особливості набору: У віконці «Project name» клавіатурою набрати на англійській мові «lab4».

III Перевірка правильності роботи програми

Для перевірки роботи схеми необхідно в пам’ять мікроконтролера завантажити програму, отриману в «AVR studio». Послідовність завантаження програми в програмному середовищі «Proteus» наведена в тексті лабораторної роботи №1. Для перевірки роботи схеми в нижньому лівому кутку робочого столу розташована панель керування симуляцією з кнопками «пуск», «пауза», «стоп».

ПОРЯДОК ОФОРМЛЕННЯ ЗВІТУ

Звіт про виконану лабораторну роботу оформлюється в тому ж зошиті для конспекту лекцій та практичних занять. Вказати номер роботи, її назву, мету та порядок виконання. Намалювати реальну принципову електричну схему лабораторної роботи, переписати команди програми з коментарями, привести письмові відповіді на три контрольних запитання, вказані викладачем. Провести

розрахунок похибки спрацювання МСЗ для свого варіанту аналогічно таблиці 4.12.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. За допомогою векторних діаграм струмів вторинних обмоток трансформаторів струму визначити фазовий зсув суми струмів.

2. Намалювати графік напруги на виході випрямляча.

3. Пояснити призначення виводів мікросхеми №№ 20, 21, 22.

4. Скласти формулу для розрахунку опору резистора R2.

5. Яким чином вплине на похибку вимірювання струму збільшення величини опору резистора R1.

6. Пояснити у чому полягає зміст команд в таблиці 4.11.

7. Пояснити значення терміну «ближнє резервування».

8. Пояснити значення терміну «дальнє резервування».

9. Пояснити значення команд «pop r16» і «push r16» в таблиці 4.11.

10. Пояснити призначення команди «rjmp opros2» в таблиці 4.12.

11. Пояснити зміст команди «cp r18, r17» в таблиці 4.11.

12. Пояснити зміст команди «brlo strum» в таблиці 4.11.

13. Пояснити, яким чином відбувається програмне перемикання входу АЦП до аналогових каналів струму уставки та поточного струму.

14. Пояснити принцип роботи таймера.

15. Пояснити призначення стеку та прокоментувати команди для його програмування.

ЛАБОРАТОРНА РОБОТА № 5

ПРОГРАМУВАННЯ ДІСПЛЕЮ

Мета роботи – отримати практичні навички програмування дисплею та перевірка правильності роботи в програмних середовищах AVR Studio та Proteus.

КОРОТКА ІНФОРМАЦІЯ ДО ВИКОНАННЯ РОБОТИ

В електроенергетиці існує необхідність показувати тільки цифри, то для цього використовують світлодіодну матрицю, яка складається з семи пікселей. Така матриця називається «світлодіодний семисегментний індикатор». Індикатор має сім світлодіодних сегментів, які позначені буквами: a, b, c, d, e, f, g та восьмий сегмент – крапка позначена буквою h. Для керування багаторозрядними дисплеями використовують принцип динамічної індикації, який полягає в почерговому вмиканні індикаторів с частотою більшою 50 Гц. Кожний сегмент індикатора споживає струм 6 мА.

ПРОГРАМА РОБОТИ

1. Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.
2. Вивчення тексту програми, набір та її перевірка в програмному середовищі AVR Studio.
3. Перевірка правильності роботи програми.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

І Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.

Схема лабораторної роботи показана на рисунку 5.1.

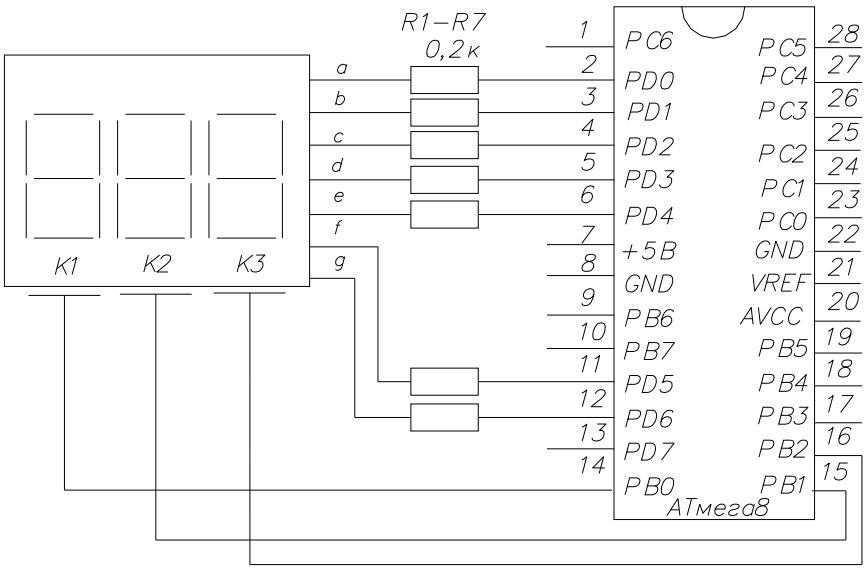

Рисунок 5.1 – Схема лабораторної роботи

Схема містить мікроконтролер, трирозрядний дисплей, який складається з трьох семисегментних індикаторів та струмобмежуючі резистори. Аноди сегментів всіх індикаторів об’єднані та підключені до порту d, запрограмованого на вихід. Кожен з катодів сегментів індикаторів підключені до порту b, запрограмованого на вихід. Для включення цифр на індикаторах на аноди сегментів подається комбінація «1» та «0», а на катоди почергово подається логічний «0».

Схема лабораторної роботи може бути складена з реальних елементів або з віртуальних елементів в програмному середовищі Proteus, яка показана на рисунку 5.2.

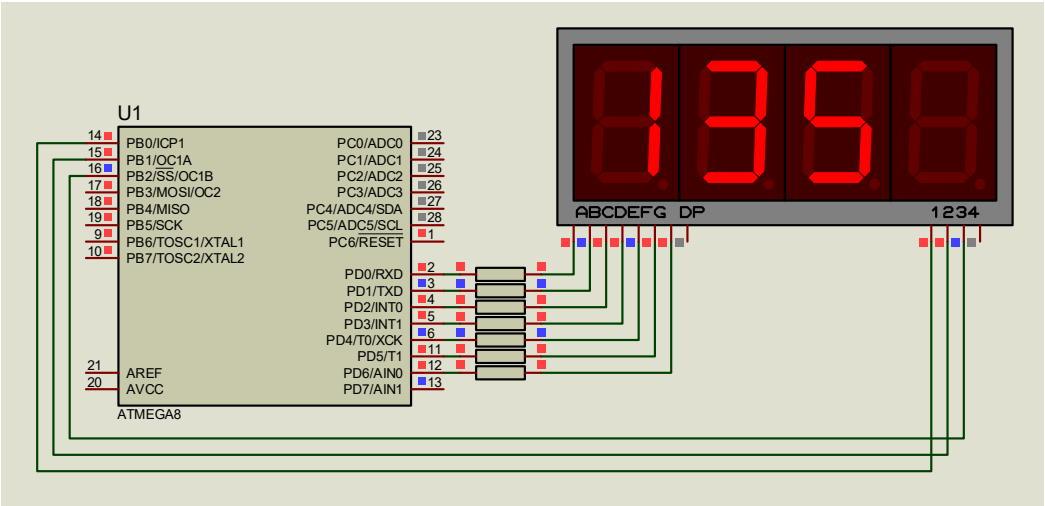

Рисунок 5.2 – Схема лабораторної роботи

Послідовність складання схеми в програмному середовищі Proteus наведена в тексті лабораторної роботи №1 з наступними особливостями. В даній лабораторній роботі використовується новий схемний елемент «чотирирозрядний дисплей». Для його включення в бібліотеку елементів схеми необхідно у вертикальному віконці клікнути на букву P, і далі клікнути на слова Category і на Optoelectronics. У віконці Sub-Category клікнути на 7-Segment Display, і далі на «7SEG-MPX4-CC». Необхідно зазначити, що нумерація розрядів дисплею зліва направо.

II Набір тексту програми та її перевірка в програмному середовищі AVR Studio

Алгоритм програми керування дисплеєм показаний на рисунку 5.3.

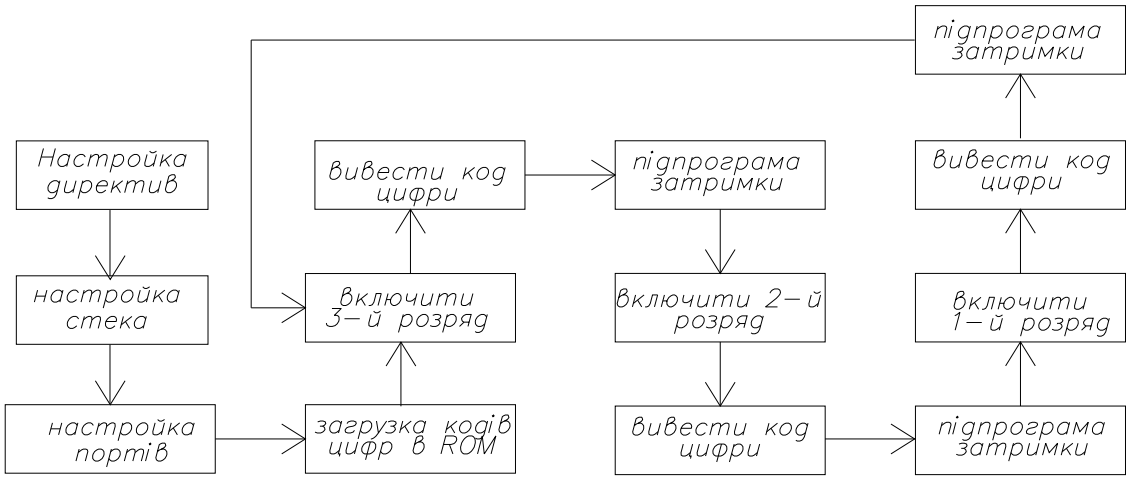

Рисунок 5.3 – Алгоритм програми керування 3-розрядним дисплеєм

Для кожного окремого блоку алгоритму складаються відповідні команди. Для першого блоку алгоритму під назвою «Настройка директив» перелік директив та програмування стека зведений в таблицю 5.1.

Таблиця 5.1 – Перелік директив Асемблера та програмування стека

Директиви	Коментарі
.device Tmega8	директива вказує, що програма складена для ATmega8
.include “m8def.inc”	директива підключає файл з бібліотеки AVR studio
.cseg	директива вказує про запис команд в ROM-пам’ять
.org 0x0000	директива вказує адресу 1-ї комірки ROM-пам’яті
ldi r16, high(RAMEND)	завантаження в r16 адреси останньої комірки RAM
out SPH, r16	пересилка адреси в старший регістр показчика стека
ldi r16, low(RAMEND)	завантаження адреси передостанньої комірки RAM
out SPL, r16	пересилка адреси в молодший регістр показчика стека

Далі за алгоритмом програмується блок «налаштування портів». Зі схеми зовнішніх підключень мікроконтролера видно, що до портів D і B підключені аноди і катоди сегментів індикаторів, отже, порти потрібно налаштувати на вихід. Для програмування портів D і B на вихід в регістри напрямку ddrd і ddrb завантажуються логічні одиниці. Результати програмування портів на вихід зведені в таблицю 5.2.

Таблиця 5.2 – Програмування портів на вихід

Перелік команд	Коментарі
ser r16	завантаження одиниць в розряди регістра r16
out ddrd , r16	програмування порту D на вихід
out ddrb, r16	програмування порту B на вихід

Далі за алгоритмом необхідно завантажити таблицю кодів цифр в ROM-пам’ять мікроконтролера для їх виводу на дисплей. Для цього використовується директива згідно таблиці 5.3.

Таблиця 5.3 – Завантаження кодів цифр в ROM-пам’ять

Перелік команд	Коментарі
table:	мітка таблиці кодів десяткових цифр
.db 0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f	
ldi zh, high(table*2)	завантаження адреси 1-й комірки таблиці в регістр zh
ldi zl, low(table*2)	завантаження адреси 1-й комірки таблиці в регістр zl

Далі за алгоритмом необхідно включити 3-й розряд дисплею згідно таблиці 5.4 та вивести семисегментний код цифри «1» на 3-й розряд дисплею згідно таблиці 5.5.

Таблиця 5.4 – Включення 3-го розряду дисплею

Команди	Коментарі
push zl	загрузка початкової адреси регістра zl з кодами цифр в стек
ldi r16,0b11111011	загрузка коду для включення 3-го розряду дисплея
out portb, r16	включити 3-й розряд дисплея

Таблиця 5.5 – Вивід семисегментного коду цифри «1» на 3-й розряд дисплею

Команди	Коментарі
ldi r16, 1	завантаження цифри «1» для виводу на дисплей
add zl, r16	Визначення адреси комірки з кодом цифри «1»
lpm	пересилка коду цифри «1» з комірки ROM в регістр r0
out portd, r0	видати семисегментний код цифри «1» на дисплей
rcall timer	Виклик підпрограми затримки
pop zl	Пересилка зі стеку початкової адреси таблиці кодів

Сутність команди «add zl,r16» полягає у тому, що до адреси комірки в ROM-пам’яті з семисегментним кодом нуля додається цифра 1, яка дає адресу комірки з семисегментним кодом одиниці. Для пересилки семисегментного коду одиниці з комірки в ROM-пам’яті в робочий регістр використовується команда lpm, яка пересилає дані з комірки в робочий регістр r0.

Команди підпрограми затримки складені в кінці основної програми. Аналогічні команди виконуються для 2-го та 1-го розрядів дисплея.

Таблиця 5.6 – Вивід семисегментних кодів цифри «3» і «5»

Команди	Коментарі
push zl	завантаження адреси zl в стек
ldi r16, 0b11111101	завантаження коду
out portb, r16	включити 2-й розряд дисплея
ldi r16, 3	загрузка цифри «3»
add zl, r16	адреса комірки з кодом цифри «3»
lpm	пересилка кода цифри «3» з комірки ROM в регістр r0
out portd, r0	видати семисегментний код цифри «3» на дисплей
rcall timer	виклик підпрограми затримки
pop zl	пересилка зі стеку початкової адреси таблиці кодів
	вивід семисегментного коду цифри «5»
push zl	завантаження адреса zl в стек
ldi r16, 0b11111110	завантаження коду
out portb, r16	включити 1-й розряд дисплея
ldi r16, 3	завантаження цифри «5»
add zl, r16	адреса комірки з кодом цифри «5»
lpm	пересилка кода цифри «5» з комірки ROM в регістр r0
out portd, r0	видати семисегментний код цифри «3» на дисплей
rcall timer	виклик підпрограми затримки
pop zl	пересилка зі стеку початкової адреси таблиці кодів

Далі за алгоритмом необхідно викликати підпрограму затримки на горіння цифри в 3-му розряді. Алгоритм підпрограми затримки показаний на рисунку 5.3.

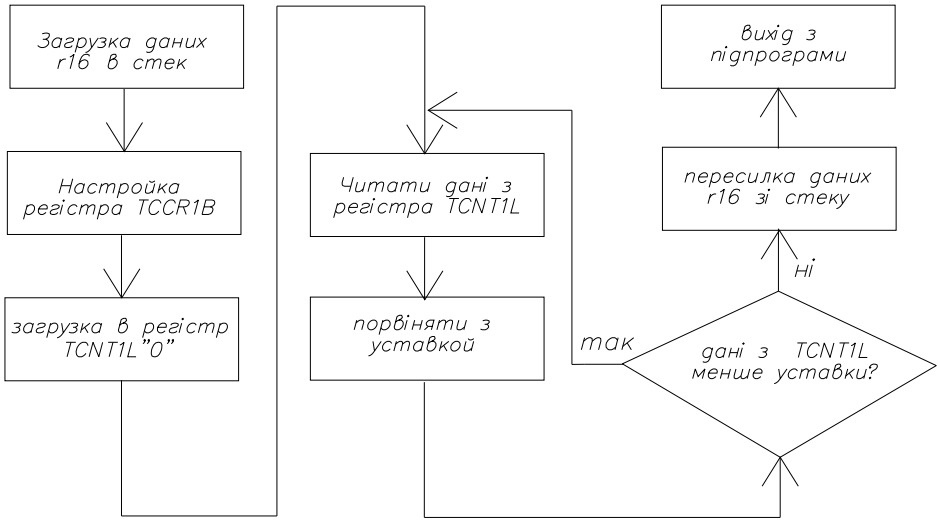

Рисунок 5.3 – Алгоритм підпрограми затримки

Таблиця 5.7 – Команди підпрограми таймера

Команди	Коментарі
timer:	мітка підпрограми таймера
push r16	завантаження в стек даних r16 з основної програми
ldi r16, 0b00000101	настройка TCCR1B на коефіцієнт ділення частоти 64
out TCCR1B, r16	пуск таймера с частотою імпульсів на вході 15625 Гц
ldi r16, 0x0000	завантаження «0» в лічильний регістр
out TCNT1L, r16	
in r16, TCNT1L	зчитати кількість порахованих імпульсів
cpi r16, 15;	порівняти з уставкою часу 15
brlo wate	якщо число в r16 менше уставки, йти на мітку wate,
pop r16	пересилка даних r16 зі стека в основну програму
ret	якщо рівна або більше, вийти з підпрограми таймера

Команди програми набрати в текстовому редакторі Word. Набраний текст зберегти на диску D. Послідовність набору тексту програми наведена в тексті лабораторної роботи № 1. Особливості набору: у віконці Project name клавіатурою набрати на англійській мові «lab5».

III Перевірка правильності роботи програми

Для перевірки роботи схеми необхідно в пам’ять мікроконтролера завантажити програму, отриману в AVR studio. Послідовність завантаження програми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1.

Для перевірки роботи схеми в нижньому лівому кутку робочого столу розташована панель керування симуляцією з кнопками «пуск», «пауза», «стоп».

ПОРЯДОК ОФОРМЛЕННЯ ЗВІТУ

Звіт про виконану лабораторну роботу оформлюється в тому ж зошиті для конспекту лекцій та практичних занять. Вказати номер роботи, її назву, мету та порядок виконання. Намалювати реальну принципову електричну схему

лабораторної роботи, переписати команди програми з коментарями, привести письмові відповіді на три контрольних запитання, вказані викладачем.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Дати технічну характеристику світлодіодному семисегментному індикатору.

2. Скласти формулу для розрахунку опору резисторів R1–R7.

3. Прокоментувати команди в таблиці 5.2.

4. Пояснити призначення стеку та прокоментувати команди для його програмування.

5. Прокоментувати команди в таблиці 5.3.

6. Пояснити принцип формування цифр семи сегментним індикатором, скласти код цифри «9».

7. Пояснити сутність команди «add zl,r16» в таблиці 5.5.

8. Пояснити сутність директиви .db.

9. Пояснити сутність команди lpm в таблиці 5.5.

10. Пояснити алгоритм роботи таймера.

11. Пояснити призначення розрядів регістру TCCR1B для програмування таймера.

12. Прокоментувати команду «out TCCR1B, r16» в таблиці 5.7.

13. Написати послідовність завантаження програми в ROM-пам’ять мікроконтролера програмному середовищі Proteus.

14. Пояснити призначення регістру TCNT1L в роботі таймера.

15. Пояснити призначення команди «ret» в таблиці 5.7.

ЛАБОРАТОРНА РОБОТА № 6

ПРОГРАМУВАННЯ ПРИЙОМОПЕРЕДАВАЧА

Мета роботи – отримати практичні навички програмування прийомопередавача та перевірка правильності роботи в програмних середовищах AVR Studio та Proteus.

КОРОТКА ІНФОРМАЦІЯ ДО ВИКОНАННЯ РОБОТИ

Операції вводу та виводу даних з мікроконтролера можна проводити за допомогою клавіатури, дисплею та комп'ютера. Мікроконтролер містить периферійний пристрій під назвою прийомопередавач, призначений для зв'язку між мікроконтролерами та комп'ютером.

ПРОГРАМА РОБОТИ

1. Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus.
2. Вивчення тексту програми, набір та її перевірка в програмному середовищі AVR Studio.
3. Перевірка правильності роботи програми.

ПОРЯДОК ВИКОНАННЯ РОБОТИ

І Вивчення електричної схеми лабораторної роботи та її складання в програмному середовищі Proteus. Схема лабораторної роботи показана на рисунку 6.1.

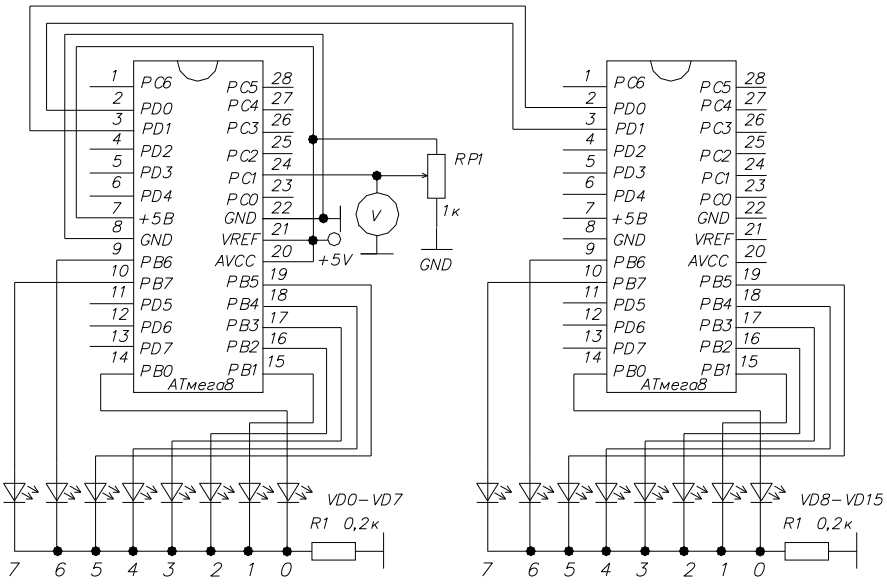

Рисунок 6.1 – Схема лабораторної роботи

Схема містить два мікроконтролера, з’єднані між собою двома проводами. По першому проводу перший мікроконтролер передає дані, а другий їх приймає, а по другому проводу другий мікроконтролер передає, а перший приймає. На першому мікроконтролері задіяний аналого-цифровий перетворювач. В результаті перетворення АЦП видає двійкове 8-мирозрядне число, яке з використанням інтерфейсу UART передається на другий мікроконтролер.

Схема лабораторної роботи може бути складена з реальних елементів або з віртуальних елементів в програмному середовищі Proteus показана на рисунку 6.2.

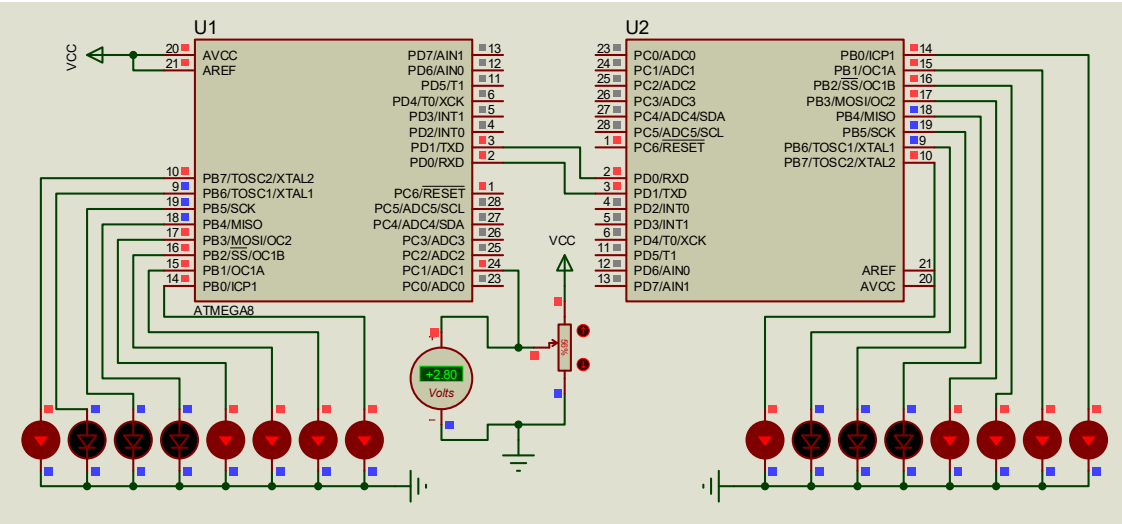

Рисунок 6.2 – Схема лабораторної роботи

Послідовність складання схеми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1.

II Набір тексту програми та її перевірка в програмному середовищі AVR Studio.

Алгоритм програми керування першим мікроконтролером для передачі даних показаний на рисунку 6.3. Для кожного мікроконтролера створюються окремі файли в AVR Studio, тобто в МК завантажуються відповідні «.hex» файли.

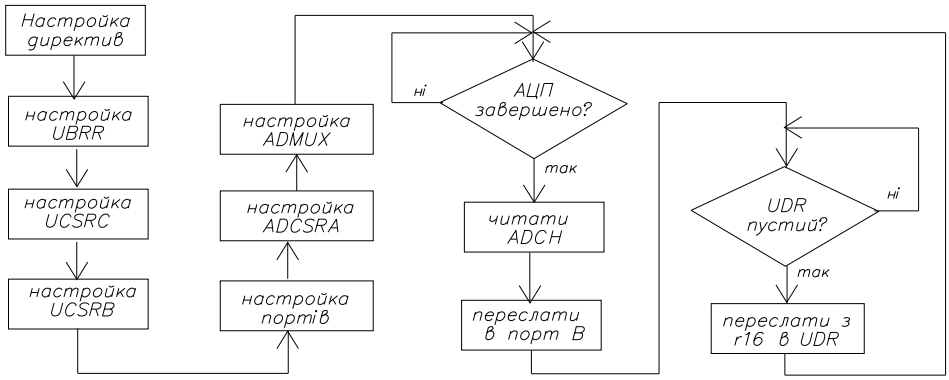

Рисунок 6.3 – Алгоритм програми мікроконтролера для передачі даних

Для кожного окремого блоку алгоритму складаються відповідні команди. Для першого блоку алгоритму під назвою «Настройка директив» перелік директив зведений в таблицю 6.1.

Таблиця 6.1 – Перелік директив Асемблера

Директиви	Коментарі
.device Tmega8	програма складена для ATmega8
.include “m8def.inc”	директива підключає файл з бібліотеки AVR studio
.cseg	директива вказує про запис команд в ROM-пам’ять
.org 0x0000	директива вказує адресу 1-ї комірки ROM-пам’яті
.equ FC = 8000000	дана назва частоти синхрогенератора в герцах
.equ BAUD= 19200	дана назву швидкості передачі біт за секунду
.equ cod=(FC/(BAUD*16)-1)	директива дала назву числу для загрузки в UBRR

Далі за алгоритмом програмуються блоки для програмування регістрів керування для режиму передачі даних.

Таблиця 6.2 – Програмування регістра UBRR

Перелік команд	Коментарі
ldi r16, low (cod)	
out UBRRL, r16	завантаження молодшого байта кодового числа
ldi r16, high(cod)	
out UBRRH, r16	завантаження старшого байта кодового числа

Таблиця 6.3 – Програмування регістрів UCSRC і UCSRB

Перелік команд	Коментарі
ldi r16, 0b10000110	
out UCSRC, r16	налаштування регістра UCSRC на формат кадра
ldi r16, 0b00011000	
out UCSRB, r16	налаштування регістра UCSRB на дозвіл передачі і прийому

Далі за алгоритмом програмується блок «налаштування портів». Зі схеми зовнішніх підключень мікроконтролера видно, що до портів D і B підключені світлодіоди, отже, порти потрібно налаштувати на вихід. Для програмування портів D і B на вихід в регістри напрямку ddrd і ddrb завантажуються логічні одиниці. Результати програмування портів на вихід зведені в таблицю 6.4.

Таблиця 6.4 – Програмування портів на вихід

Перелік команд	Коментарі
ser r16	завантаження одиниць в розряди регістра r16
out ddrd, r16	програмування порту D на вихід
out ddrb, r16	програмування порту B на вихід

Далі за алгоритмом необхідно регістри керування АЦП.

Таблиця 6.5 – Програмування регістрів ADCSRA і ADMUX

Перелік команд	Коментарі
ldi r16, 0b11100100	
out ADCSRA, R16	налаштування регістра ADCSRA для керування АЦП
ldi r16, 0b01100001	
out ADMUX, R16	налаштування регістра ADMUX для керування АЦП

Таблиця 6.6 – Програмування запуску АЦП

Перелік команд	Коментарі
adccod:	мітка для повторного запуску АЦП
opros:	метка для перевірки завершення АЦП
sbis ADCSRA, 4	пропуск команди, якщо в 4-м розряді «1» – АЦ завершено
rjmp opros	перехід на мітку opros, якщо АЦ не завершено
in r16, ADCH	читати дані з регістра даних АЦП
out portb, r16	пересилка двійкового коду в portb

Далі за алгоритмом команди для передачі даних

Таблиця 6.7 – Програмування запуску АЦП

Перелік команд	Коментарі
wate1:	мітка для повернення
sbis UCSRA,5	пропуск наступної команди, якщо в 5-м розряді «1»,
rjmp wate1	в регістрі UDR є дані для передачі
out UDR, r16	посилка байту в регістр UDR і передача в лінію
rjmp adccod	повторний запуск АЦП

Блок-схема алгоритму для мікроконтролера для прийому даних зображена на рисунку 6.4.

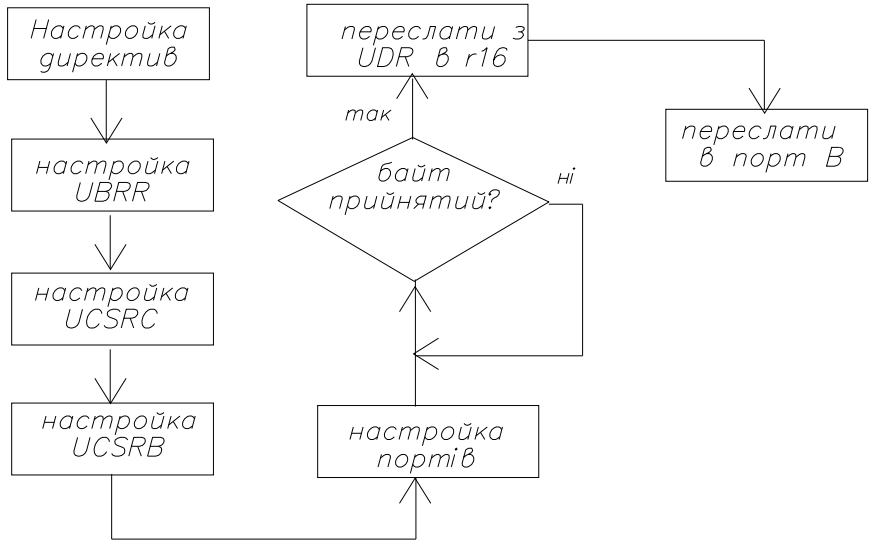

Рисунок 6.4 – Алгоритм програми мікроконтролера для прийому даних

Таблиця 6.8 – Програма для прийому байта даних

Директиви	Коментарі
.device Tmega8	програма складена для ATmega8
.include “m8def.inc”	директива підключає файл з бібліотеки AVR studio
.cseg	директива вказує про запис команд в ROM-пам’ять
.org 0x0000	директива вказує адресу 1-ї комірки ROM-пам’яті
.equ FC = 8000000	дана назва частоти синхрогенератора в герцах
.equ BAUD= 19200	дана назву швидкості передачі біт за секунду
.equ cod=(FC/(BAUD*16)-1)	директива дала назву числу для загрузки в UBRR
ldi r16, low(cod)	
out UBRRL, r16	завантаження молодшого байта кодового числа
ldi r16, high(cod)	
out UBRRH, r16	завантаження старшого байта кодового числа
ldi r16, 0b10000110	настройка регістра UCSRC на формат кадра
out UCSRC, r16	
ldi r16, 0b00011000	
out UCSRB, r16	дозвіл на передачу і прийом даних
ser r16	завантаження одиниць в r16
out ddrb, r16	налаштування порта b на вихід
wate:	мітка для повернення
sbis UCSRA, 7	пропуск наступної команди, якщо в 7-м розряді «1»
rjmp wate	очікування прийому байта в UDR
in r16, UDR	завантаження прийнятого байта в R16
out portb, r16	пересилка прийнятого байта в portb
rjmp wate	цикл

Команди програми набрати в текстовому редакторі Word. Набраний текст зберегти на диску D. Послідовність набору тексту програми наведена в тексті лабораторної роботи № 1. Особливості набору: у віконці Project name клавіатурою набрати на англійській мові «lab6».

III Перевірка правильності роботи програми

Для перевірки роботи схеми необхідно в пам’ять мікроконтролера завантажити програму, отриману в AVR studio. Послідовність завантаження програми в програмному середовищі Proteus наведена в тексті лабораторної роботи № 1.

Для перевірки роботи схеми в нижньому лівому кутку робочого столу розташована панель керування симуляцією з кнопками «пуск», «пауза», «стоп».

ПОРЯДОК ОФОРМЛЕННЯ ЗВІТУ

Звіт про виконану лабораторну роботу оформлюється в тому ж зошиті для конспекту лекцій та практичних занять. Вказати номер роботи, її назву, мету та порядок виконання. Намалювати реальну принципову електричну схему лабораторної роботи, переписати команди програми з коментарями, привести письмові відповіді на три контрольних запитання, вказані викладачем.

КОНТРОЛЬНІ ЗАПИТАННЯ

1. Пояснити чому розбіжність частот синхрогенераторів при прийомі і передачі даних не повинна перевищувати 1,5 %.
2. Описати формат кадра для передачі по інтерфейсу UART.
3. Пояснити у чому полягає програмування інтерфейсу UART.
4. Пояснити яким чином програмується швидкість передачі та прийому.
5. Пояснити у чому полягає програмування регістра UBRR.
6. Пояснити призначення регістра UCSRA.
7. Пояснити призначення регістра UCSRC.
8. Пояснити призначення регістра UCSRB.
9. Пояснити призначення регістра UDR.
10. За допомогою чого реалізується зв’язок МК з комп’ютером.
11. Опишіть фрагмент основної програми для передачі та прийому даних.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Фурман І. О. Мікроелектронні засоби керування : підручник для студентів ВНЗ / І. О. Фурман, М. Л. Малиновський, В. Г. Джулгаков. – Харків : Факт, 2007. – 486 с.

2. Ревич Ю. В. Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера / Ю. В. Ревич – 2-е изд., испр. – СПб. : БХВ-Петербург, 2011. – 352 с. : ил.

3. Баранов В. Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы / В. Н. Баранов. – М. : Издательский дом «Додэка-XXI», 2004. – 288 с. : ил.

4. Евстифеев А. В. Микроконтроллери AVR семейства Mega. Руководство пользователя / А. В. Евстифеев – М. : Издательский дом «Додека-xxi», 2007. – 592 с.

5. Белов А. В. Создаем устройства на микроконтроллерах / А. В. Белов. – СПб. : Наука и Техника, 2007. – 304 с.

6. Методичні рекомендації до виконання розрахунково-графічної роботи з навчальної дисципліни «Мікроконтролери в електроенергетиці» (для студентів 3 курсу зі скороченим терміном навчання, 4 курсу денної та заочної форм навчання спеціальності 141 – Електроенергетика, електротехніка та електромеханіка) / Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова ; уклад. Ю. В. Ковальова. – Харків : ХНУМГ ім. О. М. Бекетова, 2019. – 58 с.

Виробничо-практичне видання

Методичні рекомендації

до виконання лабораторних робіт
з навчальної дисципліни

«МІКРОКОНТРОЛЕРИ В ЕЛЕКТРОЕНЕРГЕТИЦІ»

(для студентів 3 курсу зі скороченим терміном навчання, 4 курсу денної та заочної форм навчання спеціальності 141 – Електроенергетика, електротехніка та електромеханіка)

Укладач **КОВАЛЬОВА** Юлія Вікторівна

Відповідальний за випуск *Д. М. Калюжний*

За авторською редакцією

Комп'ютерне верстання *Ю. В. Ковальова*

План 2019, поз. 207М

Підп. до друку 12.03.2020. Формат 60 × 84/16.
Друк на ризографі. Ум. друк. арк. 3,7
Тираж 50 пр. Зам. № .

Видавець і виготовлювач:
Харківський національний університет
міського господарства імені О. М. Бекетова,
вул. Маршала Бажанова, 17, Харків, 61002.
Електронна адреса: rectorat@kname.edu.ua
Свідоцтво суб'єкта видавничої справи:
ДК № 5328 від 11.04.2017.