МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт з дисципліни

«Мікроелектронні та мікропроцесорні пристрої та системи»

освітніх програм «Електричні та електронні апарати,
Електромеханічне обладнання енергоємних виробництв»

для студентів спеціальності 141 – Електроенергетика, електротехніка
та електромеханіка всіх форм навчання

2022

Методичні вказівки до виконання лабораторних робіт з дисципліни «Мікроелектронні та мікропроцесорні пристрої та системи» освітніх програм «Електричні та електронні апарати, Електромеханічне обладнання енергоємних виробництв» для студентів спеціальності 141 – Електроенергетика, електротехніка та електромеханіка всіх форм навчання. /Укл.: М.О. Поляков, – Запоріжжя: Національний університет «Запорізька політехніка», 2022.– 35 с.

Укладач: М.О. Поляков, доцент, д. т. н.

Рецензент: В.В. Василевський, ст. викл., к. т. н.

Відповідальний
за випуск: М.О. Поляков, доцент, д. т. н.

Затверджено
на засіданні кафедри
«Електричні та
електронні апарати»
Протокол № 7
від «30» березня 2022

Затверджено НМК ЕТФ
Протокол № 6
від 02.06. 2022 р.

ЗМІСТ

ВСТУП

Метою виконання лабораторних робіт є закріплення на практиці теоретичного матеріалу з дисципліни «Мікроелектронні та мікропроцесорні пристрої та системи». Для дослідження цих систем використовується пакети середовищ програмування, трансляторів, симуляторів промислових контролерів та мікроконтролерів.

Методичні вказівки містять опис 6 лабораторних робіт за темами програмування промислових контролерів мовою LD, плат Ардуїно мовою С, взаємодії мікроконтролеру I8051 з об'єктом управління та оператором, взаємодії мікроконтролеру AVR із об'єктом управління, використання симулятора EdSim51. При розробці лабораторних робіт використані матеріали робіт [8-11, 18].

Студент зобов'язаний вивчити теоретичний матеріал [1-20], виконати практичні завдання та зробити звіт з лабораторної роботи, який буде мати основні розділи:

- назва лабораторної роботи;
- її мета;
- схеми експериментів, таблиці, графіки, розрахунки, висновки,, блок – схеми алгоритмів, тексти програм на відповідних мовах програмування.

Звіт має бути оформлений у відповідності до стандарту і захищений.

ЛАБОРАТОРНА РОБОТА № 1
Програмування логічних функцій мовою LD

Мета: Навчитись програмувати системи управління на базі промислових контролерів

Короткі теоретичні дані:

Промисловий контролер це мікроконтролерний устрій орієнтований на управління обладнанням. Він складається з процесору, модулів введення, виведення та мережних модулів. Стандарт IEC 61131-3 встановлює мови програмування промислових контролерів. Ladder diagram (LD) – це мова драбинних діаграм. Програмування вивчається на прикладі програмованих промислових контролерів (PLC) американської компанії Rockwell Automation.

Рекомендована література [1,12], сайт виробника контролерів www.ab.com [15].

Хід роботи

Експеримент 1

1.Запустити пакети RSLogix500, RSLinx, RSEmulate 500.

2.Створити у пакеті RSLogix500 проект для процесора 1747-L542B вузла 1, драйвера EMUL500. У розділі I/O Configuration менеджера проекту додати до складу контролеру проекту один модуль введення та один модуль виведення.

3.Вивчити за допомогою довідки у меню HELP інструкції XIC та OTE.

4.Розробити та завантажити у емулятор контролера програму, яка містить інструкції XIC та OTE. Проглянути зміст файлів даних, які використовуються у програмі.

5.Виконати завантажену в емулятор програму для кількох значень даних. Записати здобуті результати.

6.Зберегти розроблений проект на диску комп’ютера.

Експеримент 2. Вивчення інструкцій мови програмування LD

1.У середовищі програмування RSLogix500 використовуючи вбудовану допомогу (функціональне меню/HeIp/Instruction List) вивчити склад, умовні графічні позначення, призначення, параметри та прикладі використовування бітових інструкцій, інструкцій таймерів, лічильників та порівняння.

2.Розробити та завантажити в емулятор контролеру програму, що реалізує логічну функцію y = f(x1,x2,x3) , де x1, x2, x3 – вхідні змінні, що надходять до контролера через входи дискретного модуля введення; y – вихідна змінна, значення якої формується контролером на виході дискретного модуля виведення. Виконати програму для усіх комбінацій вхідних змінних. За результатами виконання скласти таблицю істинності логічної функції. Розробити блок-схему алгоритму програми, що реалізує логічну функцію.

3.Виконати попередній пункт щодо функції, яка задана викладачем.

Контрольні запитання

1. Призначення та функціональні можливості пакетів RSLogix, RSLinx, RSEmulate.
2. Основні етапи створення проекту у пакеті RSLogix.
3. Як виконати конфігурування системи зв’язку з контролером.
4. Елементи менеджера проекту.
5. Як виконати конфігурування складу контролера.
6. Назвіть основні принципи введення у програму інструкцій, гілок та рангів.
7. Призначення команд XIC та OTE.

ЛАБОРАТОРНА РОБОТА № 2

Програмування плат Ардуіно мовою С

Мета: Навчитись програмувати мікроконтролерні системи на базі плат Ардуіно (Arduino) мовою С/С++.

Короткі теоретичні дані: Плати Ардуіно це поширений, доступний мікроконтролерний засіб для управління електромеханічним обладнанням. Вони містять контакти для під’єднання давачів та(або) виконавчих механізмів. У простішому варіанті це світлодіоди, вимикачі, Ці та інші електричні та радіо компоненти монтують на спеціальних платах. Зв'язок плати Arduino із системою програмування виконується за допомогою USB кабеля. Програмне забезпечення для програмування мікропроцесорів плат Arduino мовою С є у вільному доступі в мережі Інтернет.

Рекомендована література [1,12], сайт розробника плат [19], довідка про мову програмування [20].

Для мікроконтролерів широкого призначення існують програмні середовища, що здійснюють програмування, налагодження програми, проектування апаратних компонент тощо.

Так, наприклад, для програмування контролерів Arduino використовується середовище розробки Arduino, яка складається з безкоштовної програмної оболонки (IDE) для написання програм, їх компіляції та програмування апаратури.

Мова програмування Arduino є стандартним C++ з особливостями, які полегшують непідготовленим користувачам написання працюючої програми.

Програми для Arduino називають скетчі (від англ. Sketch), які складаютья в текстовому редакторі. Після компіляції програма зберігається в файлах з розширенням ino. Перед компіляцією ці файли обробляються препроцесором Arduino, у ході чого здійснюється виявлення помилок у разі наявності.

Також існує можливість створювати і підключати до проекту стандартні файли C++. Обов'язкову в C++ функцію main () препроцесор Arduino створює сам, вставляючи туди необхідні дії.

Програміст повинен написати дві обов'язкові для Arduino функції setup () і loop (). Перша викликається одноразово при старті, друга виконується в нескінченному циклі. В тексті своєї програми (скетчу) програміст може використовувати стандартні бібліотеки.

Менеджер проекту Arduino IDE має нестандартний механізм додавання бібліотек. Бібліотеки у вигляді вихідних текстів на стандартному C++ додаються в спеціальну папку в робочому каталозі IDE. При цьому назва бібліотеки буде додано до списку бібліотек в меню IDE. Програміст зазначає потрібні бібліотеки і вони вносяться до списку компіляції.

Arduino IDE не пропонує ніяких налаштувань компілятора і мінімізує інші настройки, що спрощує початок роботи для новачків і зменшує ризик виникнення помилок.

Найпростіша Arduino-програма складається з двох функцій:

- setup (): функція викликається одноразово при старті мікроконтролера.

- loop (): функція викликається після setup () в нескінченному циклі весь час роботи мікроконтролера.

На рис. 1 наведено повний текст однієї з найпростіших програм (скетчу) миготіння світлодіодом, підключеного до виходу 13 Arduino, з

періодом 1 секунди, де для управління цифровим виходом використовується функція digitalWrite (). Ця функція має два аргументи: номер виходу, на який виводиться дискретний сигнал, і значення сигналів HIGH (високий рівень) або LOW (низький рівень).

```
    void setup () {
    pinMode (13, OUTPUT); // Визначення вивода 13 як вихід
    }
void loop () {
    digitalWrite (13, HIGH); // Включення _вивода 13, параметр HIGH
    // - високого признак високого логічного рівня
    delay (1000); // Цикл затримки на 1000 мс – 1 секунда
    digitalWrite (13, LOW); // Виключення вивода 13, параметр LOW
    delay (1000); // Цикл затримки на 1000 мс – 1 секунда }
```

Рисунок 2.1 – Текст програми (скетчу) миготіння світло діодом

Всі використовувані в прикладі функції є бібліотечними. У комплекті Arduino IDE є велика кількість прикладів програм.

При виконанні з реальною платою Arduino скетчу рис. 1 спостерігати миготіння вбудованого світлодіода. Зовнішній світло діод та послідовний з ним резистор приєднати о одного боку до піна 13, а з іншого – до вивода GND (Загальний). Опір резистора розрахуємо за формулою:

$$R = (E - U_d) / I_d$$

де E – напруга живлення; U_d – напруга на діоді; I_d – струм через діод.

У випадку, коли відсутні реальна плата Ардуіно, зовнішні електро- радіоелементи та пристрої використовується програма-симулятор. Наприклад, UnoArduSim [22]. Вигляд екрана цієї програми наведено на рис. 2.

Хід роботи

Експеримент 1. Міготіння світло діодом

Завдання 1. У середовище Arduino IDE ввести та виконати скетч рис. 1 із вбудованим світло діодом.

Завдання 2. Змінити частоту миготіння.

Завдання 3. Додати до схеми зовнішній світло діод та резистор.

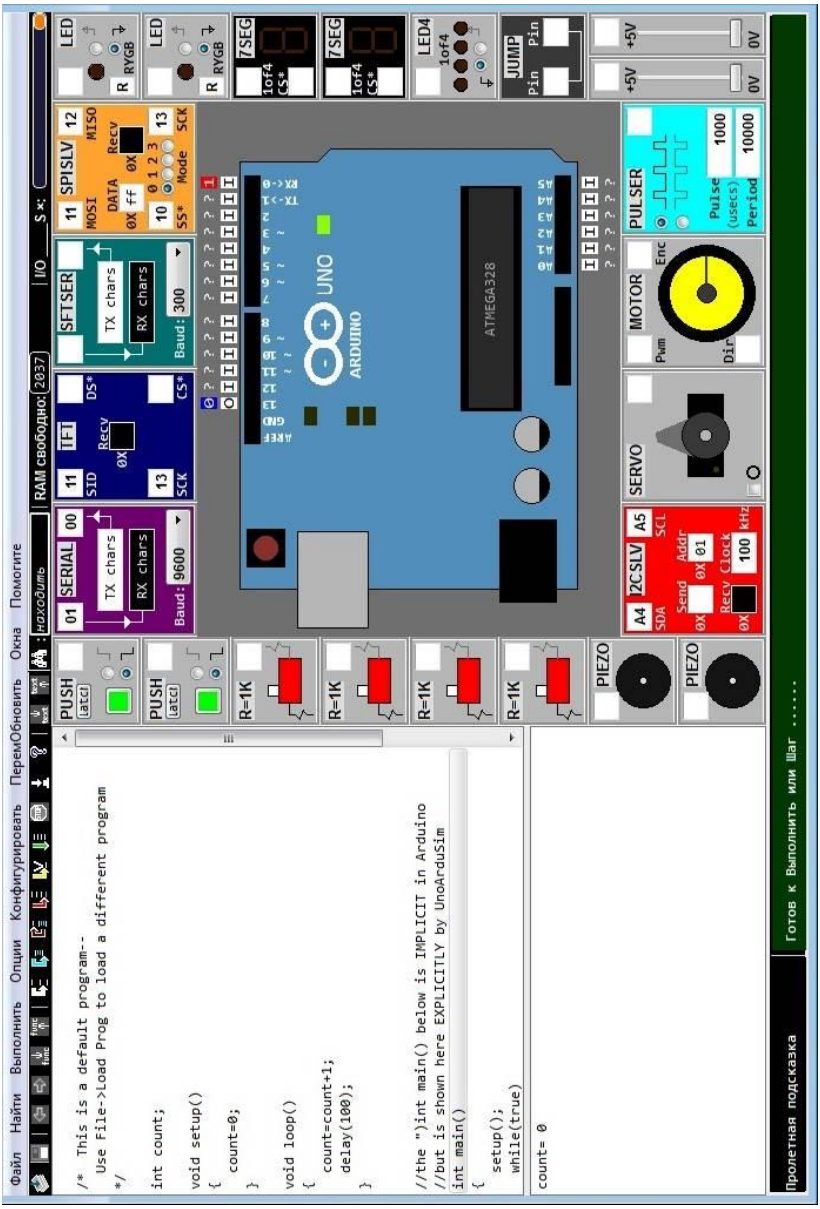

Рисунок 2.2. – Екран симулятора UnoArduSim

Завдання 4. Виконати завдання 1-3 у середовищі симулятора UnoArduSim.

Експеримент 2. Керування двигуном постійного струму

Завдання 1. У середовищі симулятора UnoArduSim ввести та виконати скетч, що керує обертанням двигуна із заданою швидкістю.

Завдання 2. Змінити напрям обертання двигуна.

Завдання 3.За допомогою моделі потенціометра задавати значення змінної *x*. Якщо x=500 то двигун повинен обертатись «вправо», інакше – «вліво»

Завдання 4. Скласти програму для керування швидкості двигуна постійного струму за допомогою моделі потенціометра.

Експеримент 3. Скласти програму вимірювання відстані за допомогою ультразвукового датчика HCSR04 з таким підключенням: Trig - пін 2, Echo - пін 3. Ультразуковий датчик керує переміщенням візка. Візок їде прямо з максимальною швидкістю, якщо відстань до перешкоди більше 40 см. Візок їде прямо з швидкістю 0,5 від максимальної, якщо відстань до перешкоди більше 20 см. Якщо відстань до перешкоди менше ніж 20 см, візок зупиняється. При зникненні перешкоди рух продовжується. Двигун підключений до виходу 5.

Контрольні питання

1. Системні характеристики плат Arduino.
2. Інтерфейс середовища програмування плат Arduino.
3. Структура програми управління.
4. Як запрограмувати лінію порта як вхід або вихід.
5. Що таке ШІМ.
6. Параметри АЦП плати Arduino

ЛАБОРАТОРНА РОБОТА № 3
Програмування взаємодії мікроконтролеру I8051 з об’єктом управління

Мета Навчитися програмувати мовою асемблер завдання взаємодії мікроконтролеру I8051 з об'єктом управління

Короткі теоретичні дані: Для програмування необхідно знати структуру мікроконтролера, призначення його зовнішніх виводів,

систему інструкцій, мову асемблер, інтерфейс середовища програмування та методи побудови програм взаємодії [2-11, 13,16-18].

Хід роботи

1.**Вивчення короткого посібника до "PROVIEW"** від Franklin Software Inc. (32-бітна версія 3.3 для Windows):

1.1.Скачати програму з WWW.FSINC.COM

1.2.Відкрити програмного забезпечення з меню СТАРТ – ПРОГРАМИ - Franklin Software - PROVIEW 32.

1.3.Запустити новий проект: на панелі завдань у верхній частині екрана натисніть кнопку PROJECT, повідомте програмі ім'я та каталог вашого нового проекту. У вікні NAME та вибір буде TYPE "8051". У нижній частині екрана відкриється вікно PROJECT.

1.4.Додати файл проекту. Виберіть ASSEMBLER FILES для типу вибору файлу. (Не забудьте зберегти та назвати новий файл).

1.5.Натисніть File, а потім New, щоб почати додавання нового файлу. Введіть приклад коду що перемикає з певною частотою виходи порта P1 мікроконтролера 8051 за допомогою констант 55H & AAH

```
        ORG 0
BACK:   MOV A, #55H       ;load A with 55h
        MOV P1,A          ;send 55H to port 1
        LCALL DELAY    ;time delay subroutine
        MOV A , #0AAH  ;load A with AA (hex)
        MOV P1,A          ;send AAH to port 1
        LCALL DELAY
        SJMP BACK       ;keep toggling P1
;-----------------------------------------
        ORG 300H        ;put time delay at address 300h
DELAY: MOV R5, #200;    R5=200 , the counter
HERE:   MOV R3,#250
AGAIN:  DJNZ R3, AGAIN ;stay here until R3 becomes 0
        DJNZ R5,HERE
        RET
        END
```

1.6. Складання програми: Якщо вам потрібен HEX-файл (для запису в ROM), то встановіть асемблер для створення файлу hex, як показано в наступних трьох кроках. Інакше перейдіть до кроку 4.

1. По-перше, натисніть OPTIONS - PROJECT. (переконайтеся, що дві мітки проекту не заплутані)

2. Щоб встановити асемблер для створення шістнадцяткового файлу, двічі клацніть L51, потім Linker. Після цього відкриються налаштування і ви зможете вибрати "INTEL HEX" у розділі Misc :.

3. Натисніть OK, щоб зберегти налаштування.

4. Після встановлення правильних налаштувань натисніть PROJECT та MAKE або "F9", щоб зібрати та створити свій HEX-файл. (примітка: ваш файл hex буде розміщено в тому ж каталозі, в якому існує ваш проект і файл)

5. Також зауважте, що якщо у вашому коді існують будь-які “ПОМИЛКИ”, асемблер помітить помилки в іншому діалоговому вікні внизу вашої робочої області.

1.7.Налагодження та відстеження зібраної програми:

1. Після збирання та створення всіх файлів ви можете скористатися «Віртуальним відладчиком» для імітації вашої програми.

2. Для цього натисніть кнопку DEBUG, а потім START. Виберіть пункт Віртуальна машина (Simulator) і процесор (8051) під мікроконтролером. Примітка: переконайтеся, що ваш курсор знаходиться у верхній частині коду, тому що, коли ви налагоджуєте програму, іноді вона почнеться з точки, де знаходиться курсор.

3. З'явиться вікно з усіма регістрами та портами.

4. Тепер для відстеження коду використовуйте клавішу F7. Це буде одним кроком через програму, дозволяючи вам вивчити і змінити регістри.

1.8.Закриття PROJECT та збереження з супровідними файлами

1. У той час як проект і всі пов'язані з ним файли відкриті, натисніть кнопку FILE, а потім SAVE ALL. Це збереже проект і всі файли в рамках проекту.

2. Далі натисніть PROJECT, а потім CLOSE, щоб закрити проект і всі файли, що стосуються проекту.

3. Ви знаєте, що готові "записати" свій шістнадцятковий файл у пам'ять про 8051.

4. Шістнадцятковий файл, як зазначено раніше, буде міститися в каталозі, в якому ви розмістили свою роботу.

2.Розробити та відладити програму [9] **відповідно варіанту**

1. Розробити програму підрахунку позитивних чисел у масиві
2. Розробити програму підрахунку негативних чисел у масиві
3. Розробити програму підрахунку кількості нульових елементів у масиві
4. Розробити програму підрахунку кількості елементів масиву, відмінних від нуля
5. Розробити програму підрахунку парних чисел у масиві
6. Розробити програму підрахунку непарних чисел у масиві
7. Розробити програму підрахунку нульових бітів у байтах масиву
8. Розробити програму підрахунку одиничних бітів у байтах масиву
9. Розробити програму пошуку кількості елементів у масиві, рівних першому елементу
10. Розробити програму сортування за зростанням масиву
11. Розробити програму сортування за убуванням масиву
12. Розробити програму заповнення масиву числами в арифметичній прогресії
13. Розробити програму заповнення масиву нулями
14. Розробити програму, що реалізує суму всіх елементів масиву
15. Розробити програму, що реалізує різницю всіх елементів масиву
16. Розробити програму пошуку найбільшого числа в масиві
17. Розробити програму пошуку найменшого числа в масиві
18. Розробити програму обчислення середнього арифметичного масиву (кількість елементів прийняти рівною 8)
19. Розробити програму додавання елементів 2-х масивів. Результат затягти у 1-й масив
20. Розробити програму переміщення блоку пам'яті на 16 байт уперед
21. Розробити програму порівняння двох масивів. Виділити змінну для результату й дорівняти її до 1, якщо масиви однакові, інакше вона дорівнюватимемо 0

22. Розробити програму логічного зрушення всіх бітів у масиві вліво
23. Розробити програму логічного зрушення всіх бітів у масиві вправо
24. Розробити програму обчислення суми окремо позитивних і негативних елементів масиву
25. Розробити програму обчислення суми окремо парних і непарних елементів масиву
26. Розробити програму перетворення числа із двійкової форми на двійково-десяткову
27. Розробити програму заповнення елементів масиву за зростанням
28. Розробити програму заповнення елементів масиву за убуванням
29. Розробити програму знаходження суми найбільшого й найменшого елементів масиву
30. Розробити програму знаходження різниці найбільшого й найменшого елементів масиву

Контрольні питання

1. Склад мікроконтролеру I8051.
2. Призначення та конфігування зовнішніх входів та виходів.
3. Групи інструкцій мікроконтролеру I8051.
4. Структура строки асемблерної програми.
5. Елементи середовища програмування “PROVIEW” від Franklin Software Inc.

ЛАБОРАТОРНА РОБОТА № 4
Програмування взаємодії мікроконтролеру I8051 з оператором

Мета: дослідити методи програмування завдань взаімодії. мікроконтролеру I8051 з оператором

Короткі теоретичні дані:

У сучасній електронній техніці найбільш поширені такі типи індикаторів: світлодіодні (LED – light emission diode) і рідкокристалічні індикатори (LCD – liquid crystal display). Поширеність LCD обумовлена в порівнянні з іншими типами індикаторів майже ідеальними електричними характеристиками [2-11, 13,16-18].

Світлодіодні індикатори мають низьку напругу живлення (1,5…3,5 В), що зручно, проте їх споживаний струм досить великий (2 - 20 мА), і це практично "ставить хрест" на їх використанні в сучасній мікропотужній радіоелектронній апаратурі. LCD при робочій напрузі 3-5 В споживають малі струми - долі мікроампера. Керування LCD здійснюється змінною напругою, але для сучасної техніки це – не проблема. На відміну від решти індикаторів вони практично нечутливі до електричних перевантажень. І ще одна особливість: LED-індикатори випромінюють світло, а LCD – поглинають.

LED-індикатори набули поширення завдяки зручності, наочності і невисокій вартості. Цифрові і знакові LED-індикатори є впорядкованим набором окремих світлодіодів (сегментів індикатора), розміщених в одному корпусі і, зазвичай, сполучених між собою в певну електричну схему (рис. 4.1).

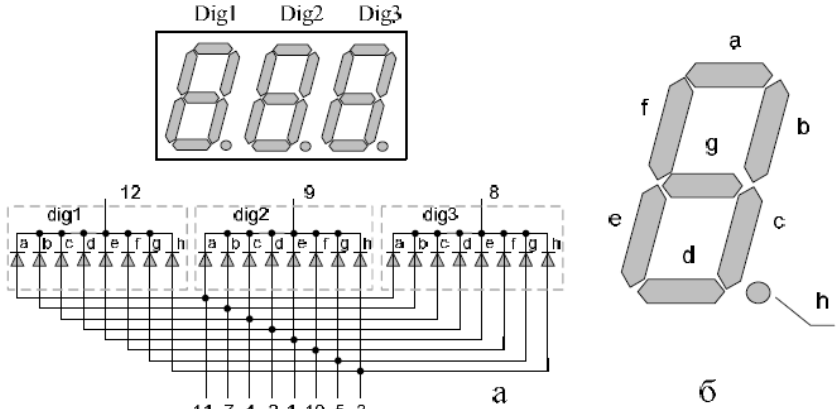

Рисунок 4.1 – Будова семисегментних LED-індикаторів

Залежно від схеми включення світлодіодів в індикаторі розрізняють індикатори із загальним катодом (рис. 4.1,а) і загальним

анодом. За способом включення індикаторів в електричну схему можна виділити два режими: статична і динамічна індикація.

LCD-індикатори конструктивно виконуються як сегментні, наприклад (ИЖКЦ1-4/14), або матричні. Останні внаслідок необхідності роботи на змінному струмі, зазвичай виконуються у вигляді готового модуля на друкованій платі із спеціальним контролером (наприклад HD44780 для знакосинтезуючих і S6B0108 для матричних).

Для забезпечення роботи багаторозрядних індикаторів або декількох однорозрядних індикаторів використовують динамічну індикацію, суть якої полягає в тому, що розряди індикатора працюють не одночасно, а по черзі. Якщо перемикання індикаторів здійснюється з великою швидкістю, то людина не помічає мерехтіння, і картинка виглядає як статична. Індикатори підключають з частотою 50÷70 Гц, що достатньо для того, щоб не помічати мерехтіння індикаторів.

Загальну структурну схему динамічної індикації наведено на рис. 4.2.

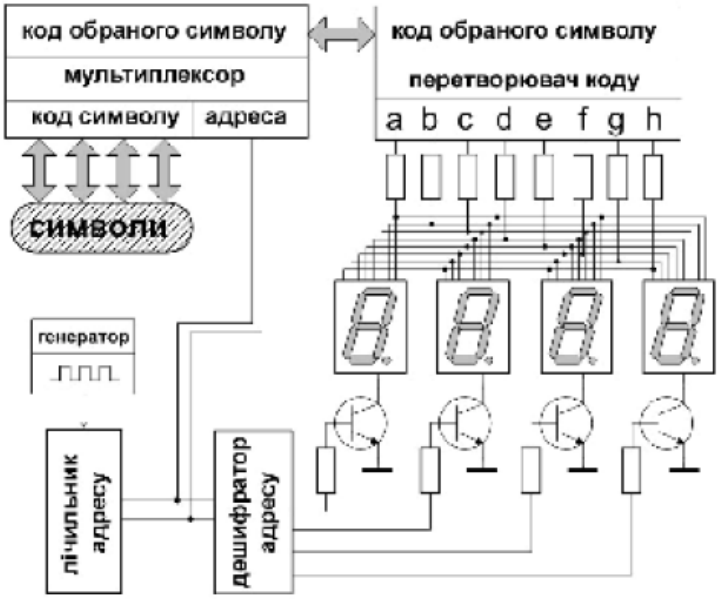

Рисунок 4.2 – Структурна схема блоку динамічної індикації на 4 знаки

Інформація поступає не на декілька перетворювачів, а на один, загальний для всіх, але порціями. Цей перетворювач коду своїми виходами підключений до всіх елементів відразу. У системі динамічної індикації мов би працює швидкодіючий безконтактний перемикач на багато положень. У одному з його положень всі виводи (сегменти) одного з розрядів підключені до відповідних виходів дешифратора. І в цей же момент через транзисторні ключі поступає сигнал управління засвіченням елементів того знакомісця, яке відповідає цьому розряду.

Завдання до лабораторної роботи

1 Розробити принципову схему і алгоритм, та написати програму для пристрою на мікроконтролері 80С51, яка б дозволяла:

- генерувати задану послідовність імпульсів заданої сквапності $S = (20 + N\text{var} \times 2)\%$
- змінювати у заданих межах скважність при натисненні на клавіші $S = \pm (10 + N\text{var})\%$
- відображати змінюваний параметр на семисегментних дисплеях.

Розробити принципову схему і алгоритм, та написати програму для пристрою на мікроконтролері 80С51, яка б дозволяла за допомогою динамічної індикації відображати дату. Студент бере дату свого народження.

2 Провести моделювання роботи приладу.

3 Оформити звіт за вимогами. Звіт повинен містити:

- титульну сторінку, тему і мету роботи;
- необхідний теоретичний матеріал;
- розрахунки і пояснення;
- електричні принципові схеми включення МПС;.
- блок-схему алгоритму роботи програми;
- лістинг програми на асемблері із поясненнями;
- результати моделювання: *print screen* екрану у потрібні моменти часу;
- висновки по роботі;
- відповіді на контрольні запитання.

Приклад виконання завдання

1 Створити проект як в прикладі та використовуючи інструменти програми у вкладці **Shematic Capture** зібрати схему як на рис.4.3:

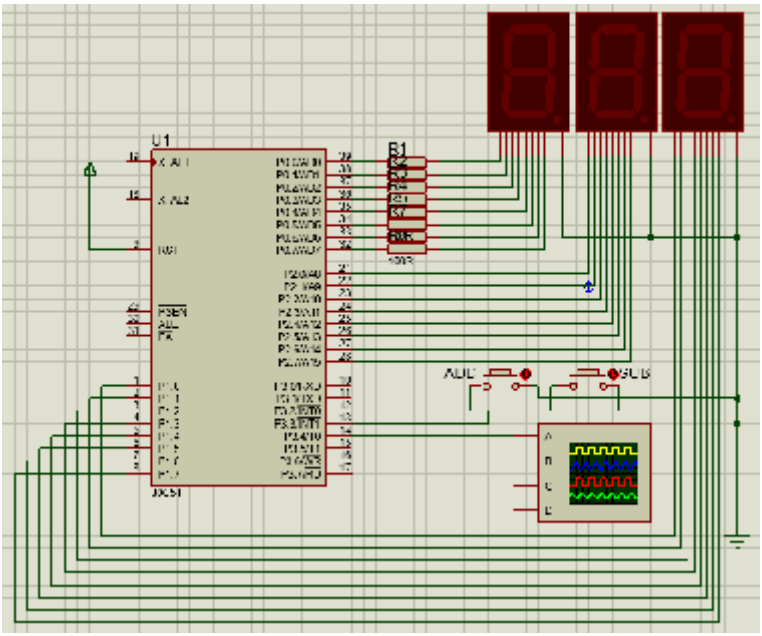

Рисунок 4.3 – Схема підключення мікро контролера

Лістинг прикладу програми для 1 завдання:

```
#include "io8051.h"                ; Include register definition file
                ASEGN CODE_SEG02:CODE,0
                jmp Start
                    ASEGN CODE_SEG02:CODE,0x0B
                jmp timer0 ;Вектор обробки перерривання таймера 0
                ASEGN CODE_SEG02:CODE,0x03
                jmp Interrupt0 ;Вектор обробки зовнішнього
                ;переривання 0
                ASEGN CODE_SEG02:CODE,0x1B
                jmp timer1 ;Вектор обробки переривання таймера 1
                ASEGN CODE_SEG02:CODE,0x13
                jmp Interrupt1 ;Вектор обробки зовнішнього
                ; переривання 0
;============ ================================
                RSEG CODE_SEG:CODE ; Switch to this code segment.
               ;Register bank 0 by default
```

```
Start:          ;мітка, що вказує початок програми
        call init
loop:
        jmp loop ;нескінченний цикл
init:
        MOV TL0, 0x00 ;задання періоду
        MOV TH0, 0x00 ;імпульсі
        MOV ACC, #255 ;зміна
       SUBB A, R7 ;скважності
       MOV B, 0xA0 ;імпульсів за
       MOV TL1, B ;допомогою
       MOV TH1, ACC ;коеффіцієнта
       clr TMOD.0 ;16-бітний таймер/лічильник,
       setb TMOD.1 ;TL0 та TH0 ввімкнені послідовно
       clr TMOD.2 ;таймер працює від внутрішнього джерела
                  ;сигналів синхронізації
       clr TMOD.3 ;робота таймера дозволяється якщо
                  ;встановлений керуючий біт TCON_TR0
       clr TMOD.4 ;16-бітний таймер/лічильник,
       setb TMOD.5 ;TL1 та TH1 ввімкнені послідовно
       clr TMOD.6 ;таймер працює від внутрішнього джерела
                  ;сигналів синхронізації
       clr TMOD.7 ;робота таймера дозволяється якщо
       ;встановлений керуючий біт TCON_TR1
       setb TCON_TR0 ;запуск таймера 0
       setb TCON_TR1 ;запуск таймера 1
       setb IE_ET0 ;дозвіл переривання по таймеру 0
       setb IE_ET1 ;дозвіл переривання по таймеру 1
;================================================
       setb TCON.0 ;перепад сигналу 1-0
       setb IE_EX0 ;дозвіл переривання по зовнігшньому
                   ;виводу 0
       setb TCON.2 ;перепад сигналу 1-0
       setb IE_EX1 ;дозвіл переривання по зовнігшньому
                   ;виводу 1
;================================================
       setb IE_EA ;глобальний дозвіл препивань
;================================================
```

```
            mov R7, #95
            call print
            ret
;=====================================
timer0:
            MOV TL0, 0x00 ;задання періоду
            MOV TH0, 0x00 ;імпульсі
            MOV ACC, #255 ;зміна
            SUBB A, R7 ;скважності
            MOV B, 0xA0 ;імпульсів за
            MOV TL1, B ;допомогою
            MOV TH1, ACC ;коеффіцієнта
            setb P3.4 ;початок імпульсу
            reti
timer1:
            clr P3.4 ;скидання імпульсу
            reti
Interrupt0:             ;збільшення значення в R7 на одиницю
            mov a, R7 ;перевірка
            SUBB a, #199 ;на
            JZ IN0rel00 ;вихід
            inc R7 ;з інтервалу
IN0rel00:
            call print ;оновлення
            reti
Interrupt1:             ;зменшення значення в R7 на одиницю
            mov a, R7 ;перевірка
            SUBB a, #0 ;на
            JZ IN1rel00 ;вихід
            dec R7 ;з інтервалу
IN1rel00:
            call print ;оновлення
            reti
print:
            MOV a, R7 ;виймання значення з R7
            call DA_corr ;двійково-десяткова корекція
            mov dptr, #code ;показник на кодуючу таблицю
            MOV a, R0 ;кодування і
```

```
            movc a, @a+dptr ;відображення 3-го
            MOV P0, a ;розряду
            MOV a, R1 ;кодування і
            movc a, @a+dptr ;відображення 2-го
            MOV P2, a ;розряду
            MOV a, R2 ;кодування і
            movc a, @a+dptr ;відображення 1-го
            MOV P1, a ;розряду
            ret
DA_corr:
            MOV R0, #0 ;очищення регітсрів
            MOV R1, #0 ;для збереження
            MOV R2, #0 ;результату
DA_rel00:
            clr C ;очищення біту переносу
            MOV R5, a
            SUBB a, #100
            JC DA_rel01
            inc R0
            jmp DA_rel00
DA_rel01:
            MOV a, R5
DA_rel02:
            clr C ;очищення біту переносу
            MOV R5, a
            SUBB a, #10
            JC DA_rel03
            inc R1
            jmp DA_rel02
            DA_rel03:
            MOV a, R5
            MOV R2, a
            ret
;======== ====================================
code: db 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F,
0x6F, 0x00
;======= = ====================================
END
```

Контрольні запитання

1 Структура статичної індикації, особливості, переваги і недоліки.

2 Структура динамічної індикації, особливості, переваги і недоліки.

3 Якою має бути частота оновлення для динамічної індикації?

4 Ефект брязкоту контактів, способи його усунення.

5 Організація індикації восьмирозрядного індикатора

ЛАБОРАТОРНА РОБОТА № 5
Програмування взаємодії мікроконтролеру AVR із об'єктом управління

Мета роботи: вивчити команди асемблеру, навчитися організовувати цикли затримки при виводі інформації на порти та візуалізувати отримані результати на світлодіодах у програмі Proteus VSM.

Короткі теоретичні дані:

AVR Studio – це інтегроване середовище розробки (IDE, Integrated Development Environment), яке дуже зручне для відлагодження AVR-додатків в операційних системах Windows 9x/Me/NT/2000/XP. Це середовище пропонує інтерфейс програмного імітатору та внутрішньо-схемного емулятору для восьмирозрядних мікроконтролерів AVR RISC. Окрім того, AVR Studio підтримує набір розробника STK500, який дозволяє програмувати AVR-пристрої, а також новий вбудований емулятор JTAG. ПакетAVRStudio безкоштовно розповсюджується фірмою Atmel.

Програма AVRStudio дозволяє писати програми на мові асемблер, ід'єднувати зовнішній компілятор для мови С, відлагоджувати текст написаної програми, використовуючи вбудований програмний стимулятор або під'єднувати внутрішньо-схемний емулятор.

В середовищі AVRStudio користувач має можливість контролювати стан кожного елементу за допомогою наступних вікон візуалізації (рис. 5.1):

– **Watch window** – відображає значення та адреси визначених змінних;

– **Register Window** – відображає стан регістрів загального призначення;

Memory window – відображає зміст пам'яті програм, пам'яті даних, регістрів вводу/виводу та зміст енергонезалежної пам'яті EEPROM.

– **I/O window** – відображає зміст регістру стану, таймерів, EEPROM регістрів, портів вводу/виводу і т. і.

– **Processor window** – відображає важливу інформацію про виконання програми, в тому числі лічильник команд (**Program Counter**), покажчик стеку (**Stack Pointer**), прапори регістру стану (Flags), лічильник циклів (**Cycle Counter**) і т. і.

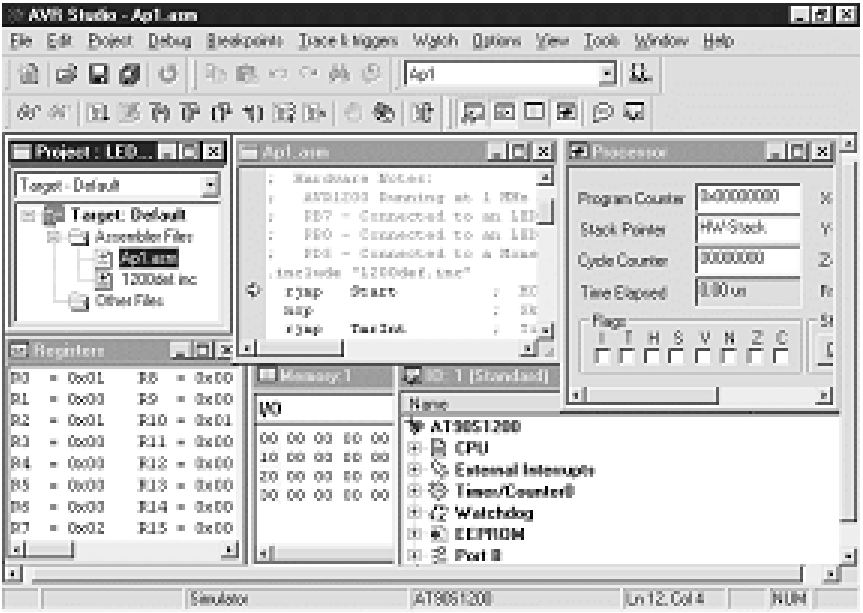

Рисунок. 5.1. – AVR Studio в процесі відладки вихідного коду програми

Під налагодженням розуміють покрокове виконання з контролем вмісту регістрів мікроконтролеру (перевірка на низькому

рівні) та змінних (перевірка на програмному рівні). Для переходу в режим відладки використовують наступні команди меню **Debug:**

– **Run, Auto Step** – перехід в режим відладки виконується, якщо зустрічається точка переривання.

– **Step Into** (клавіша <F11>) – виконує поточну команду з заходженням в підпрограми (всі вікна відновлюються).

2.1.**Step Over** (клавіша <F10>) – виконує поточну команду з виходом з підпрограми (всі вікна відновлюються).

– **Step Out** (комбінація клавіш <Shift+F11>) – запускає програму та виконує її до тих пір, поки не зустрінеться закінчення поточної підпрограми; якщо хід виконання знаходиться в області основної програми, то програма буде виконуватись до тих пір, поки не буде зупинена користувачем за допомогою команди **Break** або зустріне точку переривання.

– **Run To Cursor** (комбінація клавіш <Ctrl+F10>) – запускає програму, яка виконується до тих пір, поки не зустрінеться курсор у вікні вихідного коду; якщо зустрічається точка зупинки, то виконання програми не зупиняється; якщо позиція курсору не досягається ніколи, то програма виконується до тих пір, поки не буде зупинена командою **Break.**

Proteus VSM – це програма-симулятор радіотехнічних схем та мікроконтролерних пристроїв, що підтримує мікроконтролери PIC, 8051, AVR, HC11, ARM7/LPC2000 та інші [21].

Proteus VSM складається з двох основних модулів:

– ISIS – графічний редактор принципових схем, який використовується для вводу розроблених проектів з подальшою імітацією і передачею для розробки друкованих плат в ARES. Після налагодження пристрою можна відразу розвести друковану плату в ARES, який підтримує авто розміщення і трасування за вже існуючою схемою.

– ARES – графічний редактор друкованих плат з вбудованим менеджером бібліотек і автотрасувальника ELECTRA, автоматичною розстановкою компонентів на друкованій платі.

Для інсталяції Proteus VSM необхідно:

1. Запустити інсталяційний файл.

2. У відкритому вікні вибрати компоненти, які необхідно інсталювати.

3. Виконати настройки, необхідні для інсталяції.

Основні пункти меню Proteus ISIS: File, View, Edit, Library, Tools, Design, Graph, Source, Debug, Template, System, Help.

Хід роботи

Створення проекту AVR Studio:

1. Запустити AVR Studio.
2. В меню Project обрати команду New. У вікні створення нового проекту необхідно вказати назву проекту (Project name) і його розміщення (Location). Новий проект зручно створювати в новій папці. Зверніть увагу, що назва проекту та його розміщення не повинні мати кириличні символи.
3. Далі обирається тип проекту: – **AVR Assembler**. Використовує вбудований макроасемблер AVR Studio. – **AVR GCC**. Використовує зовнішній компілятор C, що має інтерфейс командної строки.
4. Обрати тип **AVR Assembler**. Після цього у меню **Debug Platform** вибрати AVR Simulator, у меню **Device** – **ATmega16**. Натистути кнопку Finish.
5. У вікні, що відкрилось, набрати фрагмент програми, наведений нижче.

```
include "m16def.inc"      ;Підключити файл опису імен регістрів
                          ;вводу/виводу
                          ;Ініціалізація
          ldi  r31, LOW(RAMEND ;Ініціалізація покажчику
                               ;стека, яка
          out  spl, r31        ;потрібна для використання
                               ; підпрограм
          ldi  r31, HIGH(RAMEND)
          out  sph, r31
          ldi  r31, 0xff       ;Програмування порту B на вивід
          out ddrb, r31
                               ;Безкінечний цикл
Loop:     ldi  r16,0xff        ;записати у регістр r16 всі одиниці
                               ;тобто запалити всі світлодіоди
          out  portb, r16      ;вивести зміст r16 в порт B
          rcall  wait          ;виклик підпрограми затримки
          ldi  r16,0x00        ;записати у регістр r16 всі нулі
                               ;тобто загасити всі світлодіоди
```

```
                out  portb, r16     ;вивести зміст r16 в порт B
                rcall  wait         ;виклик підпрограми затримки
                rjmp  Loop          ;повернення на мітку Loop
                                    ;підпрограма затримки wait:
                ldi r19,5           ;зовнішній цикл виконується 5 раз
label:          ldi r17, 25         ;перший внутрішній цикл
                                    ;виконується 255 раз
label1:         ldi r18, 255        ;другий внутрішній цикл
                                    ;виконується 255 раз
label2:         dec r18             ;зменшити r18 на одиницю
                brne label2         ;поки r18 не буде дорівнювати нулю
                dec r17             ;зменшити r17 на одиницю
                brne label1         ;поки r17 не буде дорівнювати нулю
                dec r19             ;зменшити r19 на одиницю
                brne label          ;поки r19 не буде дорівнювати нулю
                                    ;підпрограма забезпечує затримку
                                    ;близно на 5×255×255 тактів
                                    ;контролеру
                ret                 ;повернення з підпрограми
```

6. Далі необхідно відтранслювати програму, тобто перевірити її на правильне написання. Для цього потрібно вибрати пункт Build в меню Build. У нижньому вікні, з'явиться інформація про кількість слів коду та даних, про наявність помилок і т. і.

Створення проекту Proteus ISIS:

1. Запустити Proteus ISIS. Найбільший простір відведено під вікно редагування **EDIT WINDOW**. В ньому відбуваються всі основні процеси створення, редагування та налагодження схеми пристрою. Зліва вгорі знаходиться вікно попереднього перегляду **Overview Window** за допомогою якого можна переміщуватись по вікну редагування. Під вікном попереднього перегляду знаходиться **Object Selector** – список обраних в даних момент компонентів, символів та інших елементів. Виділений в списку об'єкт відображається у вікні попереднього перегляду. всі можливі функції та інструменти Proteus ISIS доступні через меню, розташоване вгорі основного вікна програми, через піктограми, що знаходяться під меню і в лівому куті основного вікна та через "гарячі клавіші". Внизу головного вікна

розташовані: панель управління інтерактивною симуляцією (кнопки **ПУСК-ПОКРОКОВИЙ РЕЖИМ-ПАУЗА-СТОП)**, рядок статусу (у ньому відображаються помилки, підказки, поточний стан процесу симуляції і т. і.) і координати курсору, що відображаються в дюймах. Для маніпулювання об'єктами їх потрібно виділити за допомогою правої кнопки миші. Для виділення групи можна утримуючи CTRL послідовно натискати праву кнопку на всіх об'єктах або утримуючи праву кнопку протягнути область виділення по необхідним об'єктам. Повторне натискання правої кнопки миші по виділеному об'єкту видаляє його (видалити об'єкти можна ще натиснувши на кнопку **DELETE**).

2. Відкрийте бібліотеку компонентів. Для цього натисніть кнопку **P** на клавіатурі, по піктограмі **Pick Devices**, вибрати елемент **Pick Devices** в меню **Library** або двічі натиснувши ліву кнопку у полі вибору компонентів **Object Selector.** Компоненти можна вибирати за категоріями **Category**, підкатегоріями **Sub-categories**, за виробником **Manufacturer** або ж шукати за ключовими словами **Keywords.** Для підтвердження вибору компонента натисніть лівою кнопкою по рядку з назвою компонента. Натисніть кнопку OK і помістіть компонент на схему натиснувши два рази ліву кнопку.

3. Зберіть схему, що зображена на рис. 5.2.

З'єднуйте компоненти за допомогою лівої кнопки миші. Якщо необхідно розгорніть їх за допомогою правої кнопки миші. Елементи типу "земля" вибираються в режимі **INTER SHEET TERMINAL** в лівій частині екрана.

4. Додати hex-файл з програмою в проект. Для цього натисніть правою кнопкою мишки на контролері. Оберіть пункт **Edit Properties**. У меню **Program File** виберіть файл з розширенням *.hex у папці, яка створювалась для проекту в AVR Studio. У рядку **PROCESSOR CLOCK FREQUENCY** виставити тактову частоту процесора. Зберегти проект.

5. Натиснути кнопку ПУСК на панелі управління інтерактивною симуляцією. Подивиться на результат роботи схеми.

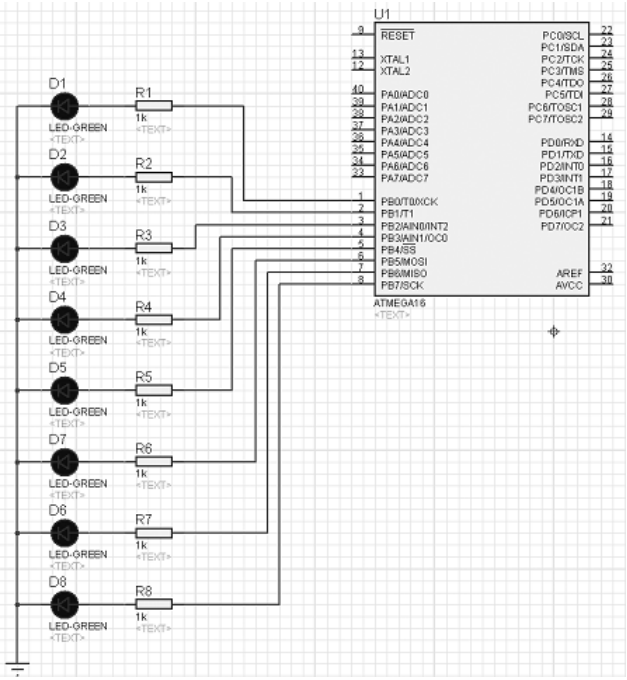

Рисунок. 5.2. – Приклад створення схеми у програмі Proteus VSM

6 Скласти програму, яка із використанням циклу затримки, наведеного в пункті 5, забезпечує перемикання світлодіодів відповідно з алгоритмом на порт що вказані у таблиці 5.1. та затримкою з часом затримки яку визначено викладачем,

Таблиця 5.1 – Варіанти завдань

Варіант	Алгоритм	Порт
1	Послідовно, починаючи з молодшого розряду	B
2	Послідовно, з країв до середини	C
3	Послідовно, з середини до країв	D
4	Послідовно, парні розряди	D

	порту	
5	Послідовно, непарні розряди порту	C
6	Послідовно, 1, 2, 3, 6, 7, 8 розряд	B
7	Послідовно, 1, 4, 5, 6, 7, 8 розряд	A
8	Послідовно, 2, 3, 6, 7 розряд	A
9	Послідовно, 1, 2, 7, 8 розряд	B
10	Послідовно, 1, 4, 5, 8 розряд	C
11	Послідовно, 1+2, 3+4, 5+6, 7+8 розряд	D
12	Послідовно, 2+3+4, 5+6+7 розряд	D
13	Послідовно, 1+4, 5+8 розряд	C
14	Послідовно, 1+8, 4+5 розряд	B

Контрольні питання

1. Особливості інтегрованого середовища розробки AVR Studio.
2. Основні пункти меню AVR Studio.
3. Створення проекту в AVR Studio.
4. Створення проекту в Proteus ISIS.

Лабораторна робота № 6
Програмування мікроконтролеру I8051 з використанням симулятора EdSim51

Мета роботи: вивчити команди асемблеру мікроконтролеру I8051, інтерфейс користувача симулятора EdSim51, схеми підключення периферійних пристроїв

Короткі теоретичні дані: симулятор можливо завантажити з [23]. Опис інтерфейсу наведено в [24]. Окно симулятора наведено на рис. 6.1. Воно містить панелі мікроконтролера, програм, сигналів на виводах портів та вікно периферійних пристроїв. Панель мікроконтролера

дає доступ до регістрів і памяті програм та даних. На панелі програм розміщені наступні кнопки:

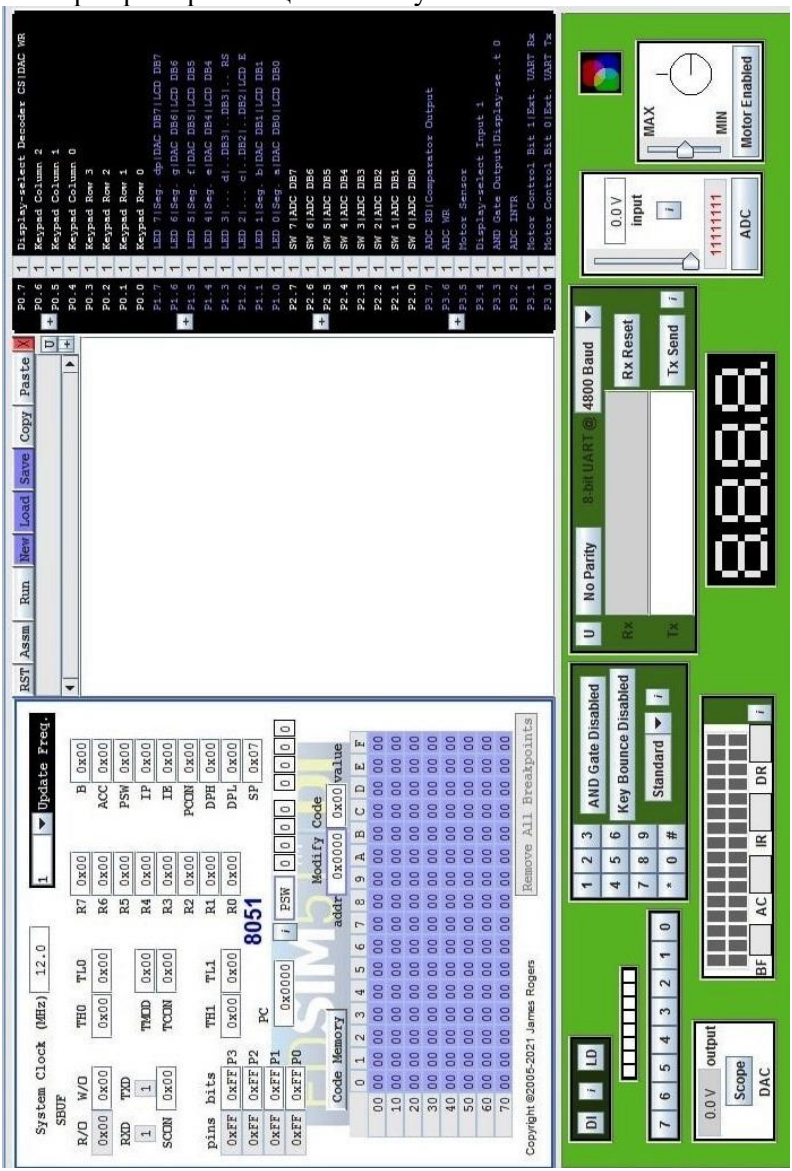

Рисунок 6.1 – Окно симулятора EdSim51

- RST – рестарт програми;
- Assm – трансляція програми;
- RUN, STEP – пуск, покрокове виконання;
- NEW, LOAD, SAVE – операції з файлами;
- COPY, Paste, EXIT – копіювати, вставити, вихід.

Панель периферійних пристроїв включає: АЦП; компаратор; 4 знаковий 7-сегментний дисплей; послідовний порт; клавіатуру; банк вітло діодів; двигун постійного струму; банк перемикачів; ЦАП; дисплей 2 рядки по 16 символів

Схему підключення цих пристроїв до портів I8051 можливо проглянути натиснув кнопку LD на панелі периферійних пристроїв.

Хід роботи

Експеримент 1. Знайомство з інтерфейсом симулятора.
Розробка простої програми.

Створити програму, що записує у резидентну пам'ять даних у комірки з адресами 41 та 42 число 1C3FH. Приклад програми, що наведено в [13, стор. 113] ввести у панелі програм та натиснути кнопку Assm. Далі натиснути кнопку STEP та спостерігати зміни у панелях.

Завдання 1. Прокоментувати зміст комірок пам’яті програм.

Завдання 2. Записати зміни, що відбулися після кожного кроку програми.

Завдання 3. Зберегти програму у файлі.

Завдання 4. Розробити та продемонструвати програму, яка обчислює логічну функцію. Вираз функції та прив’язку її змінних до пінов мікроконтролера визначає викладач.

Експеримент 2. Розробка програми що керує периферійним пристроєм.

Створити програму що виконує послідовну індикацію цифр від 0 до 9 на правий 7-сегментний індикатор у панелі периферійних пристроїв. Ввести та виконати програму з наступним кодом:

```
mov 30h, #11000000b
mov 31h, #11111001b
mov 32h, #10100100b
mov 33h, #10110000b
mov 34h, #10011001b
mov 35h, #10010010b
mov 36h, #10000010b
mov 37h, #11111000b
mov 38h, #10000000b
mov 39h, #10010000b
mov 3ah, #0
clr p3.4
clr p3.3
start: mov r0,#30h
loop: mov a,@r0
jz start
mov p1, a
inc r0
jmp loop
```

Рисунок 6.2 – Програма індикації

Завдання 1. Розробити блок-схему алгоритму програми індикації.

Завдання 2. Модернізувати програму, щоб вона відображала усі шістнадцятирічні цифри.

Завдання 3. Модернізувати програму, щоб вона відображала цифри на іншому індикаторі.

Завдання 4. Змінити час індикації однієї цифри.

Завдання 5. Навести схему підключення індикаторів до мікроконтролера.

Контрольні запитання:

1. Призначення панелей симулятора.

2. За допомогою яких пінів мікроконтролера керується двигун постійного струму. Навести схему підключення.

3. Що треба зробити, щоб шістнадцятирічну цифру «С» вивести на певний 7-сегментний індикатор.

4. Як перевести код з пам'яті програм на мову асемблер.

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна

1. Віддалений та віртуальний інструментарій в інжинірингу: монографія / за заг. ред. Карстена Хенке – Запоріжжя: Дике поле, 2015, с. 86-152. Поляков М.О., Ларіонова Т.Ю. Системи керування електричними машинами та апаратами. //В кн. Віддалений та віртуальний інструментарій в інжинірингу: монографія / за заг. ред. Карстена Хенке – Запоріжжя: Дике поле, 2015, с. 86-152.

2. Будіщев М. С. Електротехніка, електроніка та мікропроцесорна техніка: підручник / М. С. Будіщев. – Львів : Афіша, 2001. – 424 с.

3. Костинюк Л. Д. Мікропроцесорні засоби та системи / Л. Д. Костинюк, Я. С. Паранчук. – Львів : Львівська політехніка, 2001. – 200 с.

4. Локазюк В. М. Мікропроцесори та мікроЕОМ у виробничих системах / В. М. Локазюк. – Київ : Академія, 2002. – 368 с.- (Альма-матер).

5. Мікропроцесорна техніка: підручник для студентів вищ. навч. закладів / Ю. І. Якименко, Т. О. Терещенко, Є. І. Сокол та ін. ; за ред. Т. О. Терещенко.- 2-ге вид., перероб. та доп. – Київ : Політехніка: Кондор, 2015. – 594 с.

6. Мілих В. І. Електротехніка, електроніка та мікропроцесорна техніка [Текст]: підручник для студ. вищих навч. закладів / В. І. Мілих, О. О. Шавьолкін ; за ред. В. І. Мілих.- 2-ге вид. – Київ : Каравела, 2008. – 688 с.

7. Спеціалізовані мікроконтролерні системи. Теорія і практика: Підручник / Є. І. Сокол, І. Ф. Домнін, О. М. Рисований та ін. – Харків: НТУ "ХПІ", 2007. – 252 с.

8. Методичні вказівки до лабораторних робіт з дисципліни „Мікропроцесорна техніка" для студентей спеціальності 152 „Метрологія

та інформаційно-вимірювальна техніка“ 153 „Мікро- та наносистемна техніка“ денної й заочної форм навчання / Укл.: В.І.Рева. – Запоріжжя: ЗНТУ, 2019. – 114 с.

9. Методичні вказівки до самостійної роботи з дисципліни «Мікропроцесорні пристрої» для студентів спеціальності 7.092203 «Електромеханічні системи автоматизації» усіх форм навчання /укл.: А. М. Наливайко, Д. С. Пономарьов. – Краматорськ: ДДМА, 2008. – 132 с.

10. Методичні вказівки з виконання лабораторних робіт дисципліни. «Мікропроцесорна техніка» для студентів спеціальності 141 – ЕЛЕКТРОЕНЕРГЕТИКА, ЕЛЕКТРОТЕХНІКА ТА ЕЛЕКТРОМЕХАНІКА денної форми навчання. /Укл: В.В. Осадчий, О.С. Назарова, Е.М. Кулинич - Запоріжжя: ЗНТУ, 2016. – 46 с.

11. Методичні вказівки до виконання лабораторних робіт "Основи проектування спеціалізованих мікроконтролерних та вбудованих комп'ютерних систем для комплексів суднової і промислової автоматизації" : Частина 1 / Д. О. Жук, М. В. Джангиров, І. Ю. Жук. – Миколаїв : Видавництво НУК, 2011. – 70 с.

Допоміжна

12. Poliakov, M., Larionova, T. CONTROL SYSTEMS WITH PROGRAMMABLE LOGIC CONTROLLERS , pp. 101-165. Remote and virtual tools in engineering: textbook / general editorship Dr.Ing.Karsten Henke. – Zaporizhzhya: Dike Pole, 2016. – 250 p.

13. Сташин В.В. и др. Проектирование цифровых устройств на однокристальных микроконтроллерах/ В.В. Сташин, А.В. Урусов, О.Ф. Мологонцева. - М.: Энергоатомиздат, 1990. - 224 с.

14. Шпак Ю.А. Программирование на языке Си для AVR и PIC микроконтроллеров.- МК-Пресс, 2-е издание. 2011.- 400 с.

Інформаційні ресурси

15. Офіційний сайт компанії Rockwell Automation www.ab.com .

16. Intel MCS-51 . https://uk.wikipedia.org/wiki/Intel_MCS-51

17. Грищук Ю.С. Мікропроцесорні пристрої: навчальний посібник. – Харків: НТУ «ХПІ», 2005.- 280с. Електронний документ. Режим доступу http://web.kpi.kharkov.ua/ea/wp-content/uploads/sites/25/2013/04/Mikroprotsesorni-pristroyi.pdf

18. Мікропроцесорна техніка: Практикум [Електронний ресурс]: навч. посіб. для студ. спеціальності 163 «Біомедична інженерія» та 152 «Метрологія та інформаційно-вимірювальна техніка»/ В.В. Шликов; КПІ ім. Ігоря Сікорського. – Електронні текстові данні (1 файл: 3,1 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2018. – 144 с.. Режим доступу http://ela.kpi.ua/bitstream/123456789/24694/3/Shlykov_microprotsessor_techni%D0%BA%D0%B0_praktykum.pdf .

19. Офіційний сайт проекту Arduino / [Електронний ресурс] – Режим доступу: https://www.arduino.cc/.

20. Справочник языка Ардуино / [Електронний ресурс]. – Режим доступу: https://doc.arduino.ua/ru/prog/ .

21. Краткий учебный курс PROTEUS [Електронний ресурс]/ Русское руководство для начинающих. – Режим доступу: http://proteus123.narod.ru , вільний. – Загл. з екрана. – Мова рос.

22. Офіційний сайт UnoArduSim [Електронний ресурс]/ - Режим доступу: **https://www.sites.google.com/site/unoardusim/services.**

23. Офіційний сайт EdSim51. **EdSim51 - The 8051 Simulator for Teachers and Students** [Електронний ресурс]/ - Режим доступу:

https://www.edsim51.com

24.Гурский А. Л. Цифровые и микропроцессорные устройства систем телекоммуникаций: лаб. практикум для студ. спец. I-45 01 03 «Сети телекоммуникаций», I-98 01 02 «Защита информации в телекоммуникациях» всех форм обуч./А.Л. Гурский, И.С. Терех. – Минск: БГУИР, 2008. -56 с.: ил. / [Електронний ресурс]. – Режим доступу: https://ru.ua1lib.org/book/3638106/9d0ecb