МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ І.С. Скарга-Бандурова
«____»_____ 2020 р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА
ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

«Програмна реалізація тестування мікроконтролерів AVR»

Освітній ступінь «бакалавр»
Спеціальність 123 « Комп'ютерна інженерія»

Керівник проекту: _____ (підпис) Кардашук В. С. (ініціали, прізвище)

Консультант з охорони праці: _____ (підпис) Критська Я. О. (ініціали, прізвище)

Здобувач вищої освіти: _____ (підпис) Бережний П. А. (ініціали, прізвище)

Група: КІ-16 бд

Сєвєродонецьк 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітній ступінь бакалавр
Спеціальність 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ:

Т.в.о. завідувача кафедри КНІ
_____ С.О. Сафонова
«_____» _____ 2020 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ

Бережному Павлу Андрійовичу
(прізвище, ім'я, по-батькові)

1. Тема проекту (роботи): «Програмна реалізація тестування мікроконтролерів AVR» затверджена наказом по університету № 73/15.15 від «30» квітня 2020 р.

2. Строк здачі студентом закінченого проекту (роботи): 10.06.2020 р.

3. Вихідні дані проекту (роботи): матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити): огляд існуючих засобів програмного налаштування для мікроконтролерів AVR, постановка задачі на розробку діагностичних програм для мікроконтролерів AVR, розробка та тестування діагностичних програм, завантаження розроблених програм в пам'ять мікроконтролера AVR та перевірка їх виконання, розгляд питань та розроблення рекомендацій з охорони праці.

5. Перелік графічного матеріалу (з точною назвою обов'язкових креслень):

Електронні плакати

6. Консультанти роботи, з вказівкою розділів, що до них відносяться

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина	Кардашук В.С., к.т.н., доц.		
Охорона праці	Критська Я.О., ст.викл.		

7. Дата видачі завдання _____

Керівник _____ Кардашук В. С.
(підпис)

Завдання до виконання прийняв _____ Бережний П. А.
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітки
1.	Отримання завдання, збір матеріалів	18.05.20 – 24.05.20	
2.	Огляд існуючих програмних засобів налаштування мікроконтролерів AVR фірми Atmel	25.05.20 –28.05.20	
3.	Постановка завдання на розробку діагностичних програм	29.05.20 – 30.05.20	
4.	Розробка алгоритмів програмування діагностичних операцій	31.05.20 – 01.06.20	
5.	Розробка програми завантаження діагностичних програм в пам'ять мікроконтролера AVR	02.06.20 – 08. 06.20	
6.	Оформлення пояснювальної записки	08.06.20 – 09.06.20	
7.	Підготовка та подання роботи до захисту	09.06.20 – 10.06.20	

Здобувач вищої освіти _____ Бережний П.А.
(підпис) (ініціали, прізвище)

Керівник _____ Кардашук В.С.
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту бакалавра: 94 сторінки, 25 рисунків, 11 таблиць, 28 джерел посилань, 6 додатків на 22 сторінках.

У дипломному проекті розроблені діагностичні програми для мікроконтролерів AVR фірми Atmel.

Проведений огляд та аналіз програмних засобів налаштування для мікроконтролерів AVR фірми Atmel. За результатами дослідження сформульовані мета та завдання дипломного проекту. Здійснена постановка задачі на розробку діагностичних тестів портів вводу-виводу та арифметичних операцій.

Визначені шляхи реалізації поставленого завдання, розроблені алгоритми. Розроблена програма загрузки діагностичних програм в пам'ять мікроконтролера AVR.

У якості інструментального засобу налаштування мікроконтролерів обрано AVR Studio.

Проведена перевірка застосування діагностичних програм.

Розглянуті питання та сформульовані рекомендації щодо охорони праці в умовах виробництва.

МІКРОКОНТРОЛЕР, ПРОГРАМНЕ СЕРЕДОВИЩЕ, ЗАСОБИ НАЛАШТУВАННЯ, АЛГОРИТМ, ДІАГНОСТИЧНА ПРОГРАМА, КОМПІЛЯЦІЯ, ПОРТИ ВВОДУ-ВИВОДУ.

Умови отримання дипломного проекту:

СНУ ім. Володимира Даля, пр. Центральний 59а, м. Сєвєродонецьк, 93406.

ЗМІСТ

ВСТУП

Швидкий розвиток компонентів елементної бази для побудови мікроконтролерних (МК) та мікропроцесорних (МП) систем керування різноманітними пристроями та технологічними процесами сприяє науково-технічному розвитку країни, є основою удосконалення архітектури таких систем, якісного підвищення їх продуктивності і надійності.

Номенклатура та область застосування таких систем постійно розширюється. На сучасному етапі науково-технічного розвитку їх впровадження охоплює практично всі види виробничої та наукової діяльності.

Застосування МП та МК у науково-технічних рішеннях вимагає від спеціалістів досконалого володіння сучасними методами проектування МП та МК систем, вміння використовувати їх при практичному вирішенні інженерних задач.

МК в повсякденному житті застосовуються як в складній побутовій техніці, так і у супутникових навігаційних системах. До сфери застосування МК входить управління пристроями різного призначення за допомогою дискретних сигналів і багато іншого. Можна сказати, що без МК в даний час не обходиться практично жодний сучасний електронний пристрій.

Впровадження МК в усі сфери життєдіяльності ставить перед розробниками електронної техніки завдання із забезпечення діагностування МК.

Мета дипломної роботи – розроблення діагностичних програм для МК AVR фірми ATMEL з застосуванням програмного середовища налаштування AVR Studio 4.

1 ІНСТРУМЕНТАЛЬНІ ТА ПРОГРАМНІ ЗАСОБИ НАЛАШТУВАННЯ МІКРОКОНТРОЛЕРІВ

1.1 Мікроконтролери AVR

Мікроконтролери AVR – найобширніша виробнича лінії серед інших мікроконтролерів корпорації Atmel, яка представила перший 8-розрядний мікроконтролер 1993 році і з тих пір безперервно удосконалює технологію [1].

Сама ідея створення нового RISC-ядра народилася в 1994 році в Норвегії. У 1995 році два його винахідника Альф Боген (Alf-Egil Bogen) і Вегард Воллен (Vegard Wollen) запропонували корпорації Atmel випускати новий 8 розрядний RISC-мікроконтролер як стандартний виріб і забезпечити його Flash-пам’яттю програм на кристалі.

Керівництво Atmel Corp. ухвалило рішення інвестувати даний проект. У 1996 році був заснований дослідницький центр в місті Тронхейм (Норвегія). Обидва винахідники стали директорами нового центру, а мікроконтролерне ядро було запатентоване і отримало назву AVR (Alf - Egil Bogen + Vegard Wollen + RISC). Перший дослідний кристал 90S1200 був випущений на межі 1996-1997 років.

Прогрес технології з новим ядром полягає у зниженні питомого енергоспоживання (мА/МГц), розширення діапазону напруги живлення (до 1,8 В) для продовження ресурсів батарейних систем, збільшенні швидкодії до 16 млн. операцій за секунду (конвеєризація), використанням часових емуляторів і відладчиків, реалізації функції самопрограмування, удосконалення і розширення кількості периферійних модулів, спеціалізованих пристроїв (радіочастотний передавач, USB-контролер, драйвер рідинно-кристалевого індикатора РКІ, програмована логіка, контролер DVD, пристрої захисту даних тощо [2].

Успіх AVR-мікроконтролерів пояснюється можливістю простого виконання проекту з досягненням необхідного результату у найкоротші

терміни, цьому сприяє доступність великого числа інструментальних засобів проектування, що поставляються, як безпосередньо корпорацією Atmel, так і сторонніми виробниками. Провідні сторонні виробники випускають повний спектр компіляторів, програматорів, асемблерів, відладчиків, роз'ємів і адаптерів. Відмінною рисою інструментальних засобів від Atmel є їх невисока вартість.

Іншою особливістю AVR-мікроконтролерів, яка сприяла їх популяризації, є використання RISC-архітектури, що характеризуються потужним набором інструкцій (118-133), кожна з яких має довжину в одне слово (16 біт) і більшість яких виконуються за один машинний цикл. Це означає, що при рівній частоті тактового генератора вони забезпечують продуктивність в 12 (6) разів більше продуктивності попередніх мікроконтролерів на основі CISC-архитектури (наприклад, MCS51).

З іншого боку, у рамках одного застосування із заданою швидкодією, AVR-мікроконтролер може тактуватися в 12 разів меншою тактовою частотою, забезпечуючи однакову швидкодію, але при цьому споживаючи набагато меншу потужність. Таким чином, AVR-мікроконтролери представляють ширші можливості з оптимізації відношення продуктивності до енергоспоживання, що особливо важливо при розробці додатків з батарейним живленням.

Мікроконтролери забезпечують продуктивність до 16 млн. оп. у секунду і підтримують FLASH-пам’ять (ППЗП) програм різної ємкості: 1…256 Кбайт. Велика швидкодія обумовлена дворівневим конвеєром при виконанні команд. Під час першого машинного циклу відбувається вибірка команди з пам'яті програм і її декодування, а під час другого циклу ця команда виконується, також паралельно відбувається вибірка і декодування другої команди і так далі

AVR-архітектура оптимізована під мову високого рівня Сі, а більшість представників сімейства megaAVR містять 8-канальний 10 розрядний АЦП,

а також сумісний з IEEE 1149.1 інтерфейс JTAG або debugWIRE для вбудованого налагодження **(**рис 1.1).

Крім того, всі мікроконтролери megaAVR з флеш-пам’яттю ємкістю 16 кбайт і більш можуть програмуватися через інтерфейс JTAG. Арифметико-логічний пристрій (АЛП), що виконує всі обчислення, безпосередньо підключений до 32 робочих регістрів, об'єднаних у реєстровий файл. Завдяки цьому АЛП виконує одну операцію (читання вмісту регістрів, виконання операції і запису результату назад у реєстровий файл) за один машинний цикл [3].

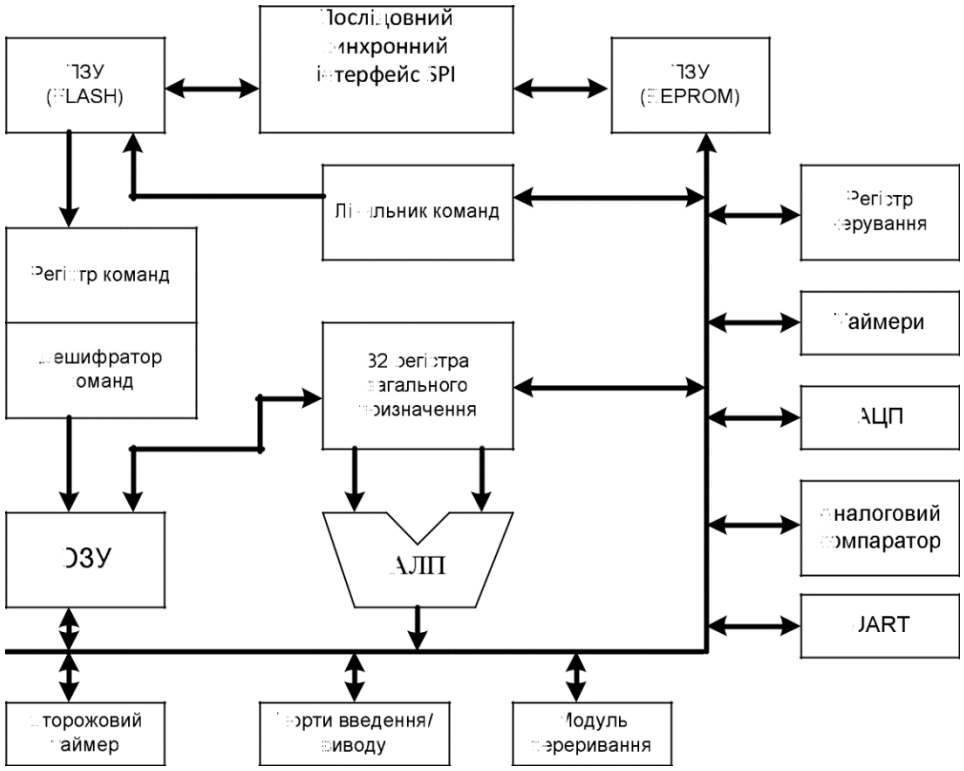


Рисунок 1.1 – Архітектура ядра мікроконтролерів AVR

У мікроконтролерах AVR практично всі команди (за винятком команд, у яких одним з операндів є 16-ти розрядна адреса) займають одну комірку пам'яті програм.

Організація пам'яті мікроконтролерів AVR сімейства Classic виконана по Гарвардській архітектурі, у якій розділені не тільки адресні простори

пам'яті програм і пам'яті даних, але також і шини доступу до них. Причому пам'ять даних складається із трьох областей: реєстрова пам'ять, статичне ОЗП і пам'ять на основі EEPROM. У зв'язку з тим, що реєстрова пам'ять перебуває в адресному просторі ОЗУ, про ці дві області пам'яті звичайно говорять як про одну. Кожна з областей (ОЗП і EEPROM) розташована у своєму адресному просторі.

В рамках базової RISC-архітектури AVR-мікроконтролери розділяються на три сімейства:

– Classic AVR – базова лінія мікроконтролерів;

– Mega AVR – мікроконтролери для складних застосувань, що вимагають великого об'єму пам'яті програм і даних;

– Tiny AVR – (маленькі) недорогі мікроконтролери (до 8 виводів).

Мікроконтролери AVR фірми Atmel є 8-ми розрядними мікроконтролерами що мають пам'ять програм (FLASH), пам'ять даних EEPROM (електрично стирається, перепрограмовується статичний запам’ятовуючий пристрій ЕСППЗП) і різноманітні периферійні пристрої, склад яких змінюється від моделі до моделі. Мікроконтролери виготовляються за К-МОП-технологією, яка у поєднанні з вдосконаленою RISC-архітектурою дозволяє досягти якнайкращого співвідношення показників швидкодія/енергоспоживання та має мале споживання від джерела живлення.

Мікроконтролери AVR мають наступні характерні особливості [4]:

– продуктивність, що наближається до 1 MIPS/МГц;

– вдосконалена AVR RISC архітектура;

– роздільні шини пам'яті команд і даних (гарвардська архітектура), 32 регістри загального призначення;

– система двобайтових команд складається з 121 базових операцій;

– Flash ПЗП програм, з можливістю внутрісистемного перепрограмування і завантаження через SPI послідовний канал, 1000 циклів стирання/запис EEPROM даних, з можливістю внутрісистемного

завантаження через SPI послідовний канал, 100000 циклів стирання/запис. Блокування режиму програмування;

– вбудовані аналоговий компаратор, сторожовий таймер, порти SPI і UART, таймери/лічильники;

– повністю статичні прилади працюють при тактовій частоті від 0 Гц до 20 Мгц;

– діапазон напруги живлення від – 1,8 В до 6,0 В.

Додаткові функції:

Скидання по включенню живлення (установка у нульовий стан);

Вбудований годинник реального часу з окремим кварцовим резонатором;

Три режими енергозбереження:

- активний режим 6.4 мА;
- режим холостого ходу пасивний (idle) 1.9 мА;
- режим низького споживання (стоповий) <1 мкА;

Зв'язок з навколишнім середовищем через 32 програмовані лінії введення/виведення, конфігуруванні у чотирьох портах (A, B, C, D).

1.2 Постановка задачі розроблення

Для виконання поставленої задачі у дипломному проекті необхідно:

1. Провести аналіз інтерфейсу програми AVR Studio 4.

2. Розглянути можливості програми AVR Studio 4.

3. Розробити діагностичні програми тестування команд МК, портів вводу-виводу, таймера, команд зсуву.

4. Здійснити компіляцію, налаштування та завантаження діагностичних програм в пам'ять МК AVR.

1.3 Перелік джерел посилань до розділу 1

1. Петров И. В. М. Программируемые микроконтроллеры. Библиотека инженера. – М.: Постмаркет, 2010. – 488с.

2. Гребнев В.В. Микроконтроллеры семейства AVR фирмы Atmel / Гребнев В.В. – М.: ИП РадиоСофт, 2012. – 176 с.: ил.

3. Королев Н. AVR: Программирование в среде AVR Studio / Николай Королев, Дмитрий Королев / Компоненты и технологии. - № 3 – 2016. С. 15 – 20.

4. Трамперт В. AVR–RISC микроконтроллеры.: Пер. с нем. – К.: «МК–Пресс», 2016. –464 с. , ил.

2 ІНТЕГРОВАНІ СЕРЕДОВИЩА НАЛАШТУВАННЯ

2.1 Емулятори та симулятори

Найефективніший спосіб налагодження програм для МК - застосування спеціалізованих професійних інструментальних налагоджувальних засобів, до яких слід віднести [1]:

- внутрисхемні емулятори (ВСЕ) - програмно апаратний засіб, здатний замінити собою віртуальний процесор у реальному пристрої;
- програмні симулятори - програмне засіб здатний імітувати роботу МК і його пам'яті;
- монітори налагодження - спеціальна програма, завантажується в пам ’ ять, що налагоджується системи.
- плати розвитку (Evaluation Boards - оціночні плати) - своєрідні конструктори для макетування прикладних систем;
- емулятори ПЗУ - програмно-апаратний засіб, що дозволяє замінити ПЗУ пристрою на ОЗУ, яке можна завантажити програму з комп’ютера через один із стандартних каналів зв'язку.

Крім цього існують і комбіновані пристрої та набори.

Симулятор складається з модуля налаштування, моделі процесора і пам'яті. Більш досконалі пристрої містять у своєму складі моделі вбудованих периферійних пристроїв (таймерів, портів, АЦП і систем переривань).

Симулятор повинен уміти завантажувати файли програм у всіх популярних форматах, максимально повно відображає інформацію про стан ресурсів МК, а також надавати можливості по симуляції виконання завантажується програми в різних режимах. В процесі налагодження модель виконує програму, і на екрані монітора комп'ютера відображається поточний стан моделі.

Завантаживши програму в симулятор, користувач може запустити її покроковому або безперервному режимі, задавати умовні або безумовні точки зупину, контролювати і вільно змінювати вміст комірок пам'яті і регістрів МК.

Симулятор дозволяє швидко перевірити логіку виконання програми, правильність виконання арифметичних операцій.

Залежно від класу використовуваного відладчика деякі моделі симуляторів підтримують високорівневе налагодження програм.

Симулятор може містити й ряд додаткових програмних засобів, наприклад інтерфейс зовнішнього середовища. Наявність такого інтерфейсу дозволяє створювати і гнучко використовувати модель зовнішнього середовища МК, функціонуючу і впливає на програму по заданому алгоритму.

В реальному системі МК «зазвичай займається» зчитуванням інформації з підключених до нього пристроїв (датчиків), обробкою і її видачі керуючих сигналів на виконавчі пристрої. Для того щоб у простому симуляторі змоделювати роботу датчика, потрібно вручну змінювати поточний стан моделі периферійного пристрою, до якого в реальному системи підключено датчик. Але існує ряд сучасних програмних розробок симуляторів, в яких щоб імітувати зовнішні умови і ситуації, зазвичай використовується спеціальний файл вхідних впливів. Цей файл визначає послідовність вхідних сигналів, що надходять на імітації пристрій.

Наприклад, для мікроконтролерів AVR цей вхідний файл програмного симулятора може виглядати наступним чином:

000000000:00, 000000006:F1, 000000015:18, 000000109:1C, 000000203:61 000000250:10, 000000344:1F 000000391:71, 999999999:ff, де кожен рядок містить цикл: дані , що надходять у вказаний порт.

В деяких моделях симуляторів ця проблема імітації зовнішніх сигналів вирішено таким чином, що симулятор має вбудований засіб для створення моделей підключених до МК зовнішніх пристроїв, включаючи кошти графічного відображення інформації.

Очевидна особливість програмних симуляторів в тому, що завантажені в них програми виконуються в масштабі часу, відмінному від реального. Однак низька ціна, можливість налагодження навіть при відсутності макета

пристрою роблять програмні симулятори вельми привабливим засобом налагодження. Необхідно також відзначити, що існує цілий клас помилок, які можна виявити лише за допомогою симулятора.

2.2 Інтегровані середовища розробки

Ідея єдності програмного і апаратного забезпечення систем на базі МК є дуже важливою. Об'єднання інструментальних засобів розробки програмного забезпечення з інструментальними засобами розробки апаратного забезпечення може стати важливою перевагою при розробці пристроїв [2].

Істотно спрощують і прискорюють процес розробки та налагодження мікропроцесорних систем, так звані інтегровані середовища розробки. Вони поєднують у собі текстовий редактор для написання вихідних текстів, транслятори з асемблера і С, компілятор, налагодження, довідкову інформацію по МК та інші засоби, необхідні розробнику.

Налаштування трансляторів, компілятора та інших компонентів здійснюється не методом вказівки ключів в командному рядку, а у вигляді діалогових вікон, де потрібно тільки розставити «галочки» в потрібних місцях

Поява інтегрованих середовищ розробки програм ще більше підвищило ефективність створення програм для МК, дозволило розробникам зосередитися на суті розв'язуваної задачі і відволіктися від конкретних деталей її реалізації. Інтегровані пакети для розробки програм випускають кілька фірм, пакети різних виробників схожі між собою по функціям, але розрізняються надаваними сервісними можливостями, зручністю роботи і якістю генерованого машинного коду.

При традиційному підході початковий етап написання програм будується таким чином. Вихідний текст набирають за допомогою будь-якого текстового редактора. По завершенню набору робота з текстовим редактором

припиняється і запускається крос-компілятор. Як правило, нова програм містить синтаксичні помилки, і компілятор повідомляє про них на консоль оператора. Потім знову запускається текстовий редактор, і оператор шукає і усуває виявлені помилки. При цьому повідомлення про їхній характер, виведені компілятором вже не видно, оскільки екран зайнятий текстовим редактором.

Цей цикл може повторюватися не один раз. І якщо програма щодо складна, збирається з різних частин, підлягає редагуванню або модернізації, то навіть цей початковий етап може зажадати дуже багато сил і часу програміста.

Уникнути рутинної роботи і тим самим істотно підвищити продуктивність праці програміста дозволяє з'явилися і швидко завойовують популярність так звані інтегровані середовища (оболонки) розробки (Integrated Development Environment - IDE).

Як правило, гарна інтегроване середовище об'єднує наявні засоби налагодження (внутрішній схемний емулятор, програмний симулятор і програматор) і забезпечує роботу програміста з текстами програм в стилі діалогових вікон. Інтегроване середовище дозволяє:

- використовувати вбудований файловий текстовий редактор, спеціально орієнтований на роботу з вихідними текстами програм;

- спостерігати одночасно в багатовіконному режимі діагностику виявлених при компіляції помилок і вихідний текст програми доступний редагування;

- вести паралельну роботу над декількома проектами. Менеджер проектів дозволяє використовувати будь-який проект в якості шаблону для новостворюваного. Опції використовуваних компіляторів і список вихідних файлів проекту встановлюються в діалогових меню і зберігаються в рамках проекту, усуваючи необхідність роботи з незручними bat - файлами:

- піддавати рекомпіляції, тільки редаговані модулі;

- завантажувати програму в наявні засоби налагодження і працювати з ними без виходу з оболонки;
- підключати до оболонці практично будь-які програмні засоби.

Останнім часом функції інтегрованих середовищ розробки стає приналежністю програмних інтерфейсів найбільш «просунутих» емуляторів і відладчик симуляторів. Такі функціональні можливості в поєднанні з дружнім інтерфейсом істотно прискорюють роботу програміста. Таким чином, вибираючи інструментальні засоби налагодження, доцільно враховувати наступний комплекс показників: перелік підтримуваних МК, обмеження на ресурси емульованих/симульованих МК, можливість символьного налагодження, перелік підтримуваних компіляторів і, нарешті, сервісні можливості.

2.3 Мова Асемблера

Перш ніж почати розробку якого пристрою на база МК дуже важливо познайомитися з основами програмування на мові Асемблера. При створенні додатків для МК слід не тільки освоїти цей метод програмування, але і навчитися добре розуміти, як крок за кроком виконується ваша програма, і що при цьому відбувається в пристрій.

Система команд асемблера Atmel Studio містить 121 команду [3].

Розробнику програмного забезпечення надаються наступні типи асемблерних команд:

– арифметичні і логічні;
– переходів;
– пересилання даних;
– побітові;
– тестування бітів.

Щоб процес вивчення мови, написання і налагодження програм на Асемблері був більш простим і зрозумілим, існує кілька прийомів. По-перше

- використання візуалізації процедур виконання команд процесором. По-друге - застосування методів структурного програмування, щоб зробити програми більш простими для читання і розуміння.

Візуалізацію виконання команд найкраще здійснювати, використовуючи структурну схему МК, на якій наголошується проходження даних при виконанні кожної команди. В результаті забезпечується гарне візуальне представлення процесу виконання команд.

2.4 Мови програмування високого рівня

Для програмування МК можна використовувати різні мови високого рівня. Термін «мова високого рівня» служить для позначення мов, що використовуються для написання легко читаються програм, які конвертуються (компілюються) в мова асемблера, а потім перетворюються в об'єктний код (біти і байти) для їх виконання мікроконтролером.

Перелічимо основні характеристики мов високого рівня [4]:

- наявність вбудованих функцій (наприклад, консольний ввід/вивід) з підключаються бібліотеками;
- різноманітні типи даних (8-, 16-, 32-бітні і з плаваючою точкою);
- виконання арифметичних операцій з використанням стека;
- використання локальних і глобальних змінних, покажчиків і структур даних;
- розподіл пам'яті;
- доступ до апаратних регістрів;
- символічна інформація для симулятора/емулятора.

Реалізація цих характеристик може бути проблематичним для вбудованих МК, які володіють наступними особливостями:

- обмежений обсяг пам'яті програм ROM і пам'яті даних RAM;
- відсутність BIOS або операційної системи;

- зміна функції контакту введення/виведення (коли вихід може використовуватися як цифро-аналоговий/послідовний вхід/вихід).

Таким чином, використання асемблера необхідно, якщо до розміру і швидкодії генерується коду пред'являються дуже жорсткі вимоги. В даний час таких випадків стає все менше і менше, тому що практично завжди можна взяти більш «швидкий» МК з великим об'ємом пам'яті. Крім того, сучасні пакети крос коштів дозволяють легко писати змішані програми, де частина модулів написана на мови С найбільш критичні до швидкодії частини - на асемблері. Компілятори мови С дозволяють також вставляти у вихідні тексти асемблерні інструкції.

При розробці програмного забезпечення для МК існує кілька правил, які слід виконати, щоб обсяг ресурсів, що використовуються не перевищив доступної межі.

- використовувати тільки один вид інтерфейсу з апаратними засобами (зовнішніми пристроями). Застосування різних інтерфейсів створює проблеми, якщо буде потрібно підключати інші типи зовнішніх пристроїв;

- ідентифікувати глобальні змінні, специфічні для підпрограм, і не використовувати їх де-небудь ще в коді;

- використовувати скрізь, де можливо, локальні змінні (це можна реалізувати тільки в мовах високого рівня);

- якщо передбачається наявність тимчасово використовуваних змінних, то програма повинна забезпечити їх унікальне використання.

Дотримання цих правил при розробці прикладних програм позбавить вас надалі від проблем, пов'язаних з усуненням важко виявляються нестійких помилок у програмі

За останні роки МК AVR набули великої популярності, залучаючи розробників досить вигідним співвідношенням показників "ціна/швидкодія/енергоспоживання», зручними режимами програмування, доступністю програмно-апаратних засобів підтримки і широкою номенклатурою кристалів. МК цієї серії є зручним інструментом для

створення сучасних високопродуктивних і економічних контролерів багатоцільового призначення.

МК сімейства AVR фірми ATMEL володіють низьким рівнем споживання, невисокою вартістю при досить високих функціональних можливостях, високим швидкодією і можливістю багатократного перезапису програм. Хоча аналогічні за характеристиками МК випускаються багатьма фірмами, за загальним комплексом властивостей сімейство AVR одне з найбільш ефективних у класі недорогих 8-розрядних МК.

У МК AVR є дві особливості, які відрізняють це сімейство від інших МК. По-перше, система команд і архітектура ядра AVR розроблялася спільно з фірмою-розробником компіляторів мов програмування високого рівня IAR Systems. У результаті з'явилася можливість створення AVR-програм на мові С без великої втрати в продуктивності в порівнянні з програмами написаними на мові асемблера. По-друге, одним з істотних переваг МК AVR стало застосування конвеєра. У результаті для МК AVR не існує поняття машинного циклу: більшість команд виконується за один такт. Для порівняння зазначимо, що МК сімейства PIC, які користуються великою популярністю, виконують команду за 4 такту, а класичні 8051 - взагалі за 12 тактів.

2.5 Інтегроване середовище налаштування AVR Studio 4

Для програмування МК сімейства AVR існує багато засобів розробка, проте, найбільш популярним, поза сумнівом, є програмний пакет AVR Studio 4. Є ряд причин такої популярності. Це безкоштовний програмний пакет, що розроблений фірмою "ATMEL", який об'єднує в собі текстовий редактор та програмний емулятор МК різних типів [5].

Пакет AVR Studio 4 використовується також спільно з апаратними засобами налаштування фірми "ATMEL".

Сучасна версія Atmel Studio (раніше AVR Studio) - заснована на Visual Studio безкоштовне інтегроване середовище для розробки додатків для 8- і 32-бітних мікроконтролерів сімейства AVR і 32-бітних мікроконтролерів сімейства ARM від компанії Atmel, що працює в операційних системах Windows.

Atmel Studio містить компілятор GNU C / C ++ і емулятор, що дозволяє налагодити виконання програми без завантаження в мікроконтролер.

Поточна версія (Atmel Studio 7) підтримує всі МК, які випускаються на сьогоднішній день фірмою Atmel – мікроконтролери архітектур AVR, AVR32, ARM та засоби розробки тощо.

Atmel Studio містить в собі менеджер проектів, редактор вихідного коду, інструменти віртуальної симуляції і внутрішньосхемного налагодження, дозволяє писати програми на асемблері або на C / C ++.

Програмне середовище AVR Studio надає користувачу можливість повністю контролювати виконання програм з використання симулятора, який підтримує всі типи МК AVR. Середовище налаштування підтримує виконання програм у вигляді асемблерного тексту формату AVR Assembler, IAR Systems Assembler та у форматі мови програмування С компілятора фірмі IAR Systems ICCA90 C Compiler. З AVR Studio також сумісні всі програматори та засоби налаштування, що підтримують МК фірми **“**ATMEL**”**.

2.6 Висновки до розділу 2

У другому розділі дипломного проекту проведений огляд інструментальних та програмних засобів розроблення та налаштування мікроконтролерів AVR фірми Atmel.

За результатами дослідження сформульовані мета та завдання дипломного проекту. Здійснена постановка задачі на розроблення діагностичних тестів команд МК.

2.7 Перелік джерел посилань до розділу 2

1. Хартов В.Я. Микроконтроллеры AVR. Практикум для начинающих. – М.: Изд–во МГТУ им. Н.Э. Баумана, 2012. – 240 с.: ил.

2. Белов А.В. Самоучитель разработчика устройств на микроконтроллерах AVR / Белов А.В. – СПб.: Наука и техника, 2008. – 544 с.: ил.

3. Малахов В. П. Мікроконтролери: Навч. посіб. для студентів бакалаврської підготовки за напрямом 6.050102 – комп'ютерна інженерія / В. П. Малахов, Д. П. Яковлев. – О.: Наука і техніка, 2008. – 224 с. – Бібліогр. 5 назв.

4. Гумеров Р.И. Практикум по микропроцессорам. Часть первая: микроконтроллеры AVR. Руководство. – Казань: КГУ, 2009. – 37 с.

5. Навчальний посібник з дисципліни «Проектування мікропроцесорних систем» Розділ «Програмування мікроконтролерів родини AVR». [Електронний ресурс]. Режим доступу https://acts.kpi.ua/app/uploads/2017/05/ПОСІБНИК-ПРОГРАМУВАННЯ-AVR.pdf (дата звернення: 14. 05. 2020).

3 ДІАГНОСТИКА МІКРОКОНТРОЛЕРІВ ATMEL

3.1 Команди та коментарі AVR-асемблера

Особливості мнемонічною запису більшості команд в AVR-асемблері такі, як і в інших асемблерах. Спочатку йде власне команда (в AVR команди бувають від 2 до 4 букв), потім через пробіл або знак табуляції слідують команди. Деякі команди операндів не мають (наприклад, lpm, reti), в інших – один операнд (наприклад, inc r16).

Якщо команда має два операнди, то спочатку вказують приймач, потім джерело (т. зв. прямий польський запис). Між джерелом і приймачем обов'язково повинна стояти кома (з будь-яким числом пробілів до або після неї, або взагалі без них) [1].

Наприклад: sub r16,r17 або sub r16, r17

Наведений вираз означає, що з вмісту регістра r16 потрібно відняти вміст регістра r17 і результат зберегти в регістрі r16.

Після крапки з комою в кінці виразу (якщо необхідно) наводиться коментар до програми. Фраза коментарів обмежена 120 символами. Якщо необхідно продовжити коментар на наступний рядок, то цей рядок потрібно знову почати з крапки з комою.

Приклад використання коментарів до програми:

```
ldi r16, r17          ; Завантажити вміст регістра r17 в регістр r16
; Підпрограма ініціалізації
```

Загальне правило для використання пропусків і знаків табуляції таке: не можна розбивати ідентифікатор на частини і навпаки, не можна зливати різні ідентифікатори між собою. Між найменуванням інструкцій, змінних, регістрів і т. п. повинен бути хоча б один пропуск або знак табуляції.

AVR-асемблер не розрізняє регістр букв, в тому числі і в привласнених програмістом іменах змінних, констант, міток і т. п.

Наприклад: JMP, Jmp, jmp

Форма вищенаведених записів буде однаково правильною для всіх трьох команд.

Команда повинна займати окремий рядок. Розбивати команду на частини розривом рядка не можна. Окрім команди єдиний рядок може містити коментарі.

Мітка (label) – ідентифікатор довільної довжини використовується в програмі і закінчується двокрапкою без пропусків перед ним.

Наприклад: mitka:, Init:, loop:

Мітку можна розташовувати в окремому рядку. Крім простої вказівки на адресу переходу для команд розгалуження, мітка служить також вказівкою на адресу підпрограми і заодно є її ім'ям.

В деякі команди можна включати вирази і числові значення. Числа за замовчуванням вважаються десятковими, наприклад, 23. Числа у шістнадцятеричній системі можна записувати як в мові Pascal ($0A) або як в мові C (0x0A). Форма запису асемблера архітектури x86 (0Ah) не допускається. Двійкові числа записуються як в мові C: 0b01010101.

Наприклад: ldi r30, c1 + c2 ;c1 і c2 – мітки

В команди можна включати алгебраїчні і логічні вирази [2].

У виразах допустимі всі арифметичні і логічні команди, включаючи команди порівняння. Однак дії у виразах можуть проводитися тільки над константами, а не над вмістом регістрів, які на момент компіляції невідомі.

Крім власне команд, в асемблерній програмі можуть зустрічатися директиви компілятора. Найуживаніші, які є практично в кожній програмі, це def (definition), equ (equvalent) і include.

Перші два призначені для визначення імен змінних користувача і констант відповідно. Директива асемблера починається з точки.

Приклад:

```
.equ max_value = $11      ; змінній max_value присвоєно значення 17
.def temp = r16           ; в регістрі r16 зберігається змінна temp
.def counter = r05        ; в регістрі r05 зберігається змінна counter
```

.def digital= 0b01010101 ;змінній digital присвоєно двійкове

;значення 0b01010101

Ці визначення з метою структурування програми розташовують на початку тексту. Ніяких перевірок, крім синтаксичних, в цьому місці програми не проводиться, тому можна оголосити два різних імені для одного регістра і вони будуть сприйматися як синоніми.

Всі прийняті в технічних описах фірми Atmel найменуваннях та інші необхідні константи вводяться так само і збираються в спеціальні файли з розширенням inc. Такі файли додаються до кожної моделі МК (наприклад, 2313def.inc – для моделі AT90S2313, tn2313def.inc – для моделі Atiny2313 і т. п.).

Сам асемблер "не підозрює" про існування таких речей, як PortA або DDRC, а "знає" тільки числові адреси відповідних регістрів. Відповідність між цими мнемонічними позначеннями та адресами і встановлюється за допомогою inc-файлів, причому для різних моделей МК ці адреси можуть відрізнятися. Для того щоб включити ці відповідності до тексту програми і служить директива include.

Наприклад: .include "2313def.inc"

В даному випадку підключається файл з визначеннями констант і адрес для МК AT90S2313. Подібним рядком повинен починатись текст будь-якої AVR-програми.

Директива device також використовується на початку програми і окремо вказує асемблеру на конкретну модель МК, що застосовується.

Наприклад: .device AT90S2313

Якщо inc-файли вказують асемблеру на справжні адреси регістрів для конкретної моделі МК, то ця директива не дозволить використовувати команди, які не підтримуються даною моделлю.

Розглянемо директиву .db, яка дозволяє зберігати константи у Flash-пам'яті програм, так і в EEPROM. Для того, щоб показати куди саме будуть записуватися дані спільно з директивою .db можна вказати директиву .eseg

(для EEPROM). Якщо ця директива відсутня, то без спеціальних вказівок дані по директиві .db будуть збережені в пам'яті програм.

Команди логічних операцій складають важливу частину функціональності будь-якого МК. Значна частина функцій МК здійснюється саме через логічні команди з регістрами.

Логічні команди застосовуються тільки до РЗП. У цій групі представлені всі стандартні логічні команди: побітове and (логічне множення), or (логічне додавання) і eor (мажоритарна операція), а також переведення в зворотний код com і в додатковий neg. За допомогою цих операцій можна утворювати будь-які інші логічні функції. Відзначимо, що дві команди, які працюють з константами andi і ori, застосовуються лише до старших РЗП, починаючи з r16. Найбільш часто команди логічних операцій виконують маскування окремих бітів або їх груп.

Наприклад:

and temp, 0b00001111

Команда and дозволить залишити молодшу тетраду змінної temp без змін, а старшу – обнулити.

Наприклад: ori temp, 0b00001111

Команда ori дозволить залишити старшу тетраду змінної temp без змін, а молодшу – встановити в одиничне значення.

Команду eor можна назвати «елементом порівняння»,що дозволяє зафіксувати ті біти, які збігаються (або не збігаються) в обох операціях (біти, які збігаються, встановляться в «0», які не збігаються – в «1»). Команда eor придатна для елементарного шифрування даних.

Крім перерахованих стандартних логічних операцій, до цієї групи команд часто відносять також і команди clr (очистити (скинути) всі біти), ser (встановити всі біти), tst (перевірка на негативне або нульове значення – ця команда поміщена в групу команд порівняння) і дуже корисну команду swap, яка міняє місцями тетради одного байта (ця команда поміщена в групу команд роботи з бітами).

3.2 Застосування команд логічних операцій

Для виконання програмних проектів в інструментальному середовищі налаштування МК AVR для команд логічних операцій потрібно виконати наступні дії [5]:

1) Створити новий проект.
2) Обрати МК, наприклад, 90S8515 або аналогічний за функціями.
3) Призначити операнду 1 та операнду 2 двійкові значення.
4) Загрузити операнд 1 в регістр R16 або інший.
5) Загрузити операнд 2 в регістр R17 або інший.
6) Виконати команду логічної операції.
7) Вивести результат виконання команд в Port D або інший.
8) Очистити регістри R16 та R17.
9) Зберегти результати виконання завдання.

До групи команд логічних операцій належать наступні команди наведені в таблиці 3.1.

Таблиця 3.1 – Команди логічних операцій

Мнемоніка команди	Операція
and	логічне множення
andi	логічне множення з безпосереднім значенням
or	логічне додавання
ori	логічне додавання з безпосереднім значенням
eor	мажоритарна операції (додавання по модулю 2)

Для реалізації програми виконання, наприклад, команди and порядок виконання виглядає так, як зазначено на початку п. 2.2.

Для реалізації діагностичних програм виконання інших логічних команд порядок виконання буде аналогічним.

Для реалізації перевірки виконання логічних команд необхідно обрати операнди, що наведені в таблиці 3.2

Таблиця 3.2 – Операнди для виконання логічних команд

Операнд 1	Операнд 2	Операнд 1	Операнд 2
01	34	AA	55
AF	DC	DD	BB
DD	45	EB	21
44	21	04	FA
17	68	51	17
CD	ED	18	3C
12	54	6F	55
23	78	AC	29
99	7C	76	F9
92	4F	D6	E4

Логічний зміст виконання команди and наведено в таблиці 3.3.

Таблиця 3.3 – Таблиця істиності виконання команди and

Операнд 1	Операнд 2	Результат
0	0	0
0	1	0
1	0	0
1	1	1

Наведемо алгоритм програми, що реалізує виконання логічної команди множення (команда and).

Необхідно результат виконання команди and видати у порт D МК.

У програмі використовуються два двійкових операнди 0b01010101 та 0b00110100.

Попередньо регістр R18 встановлюється в одиничний стан, та підготовлюється регістр DDRD на виведення результату операції.

В регістр r16 командою ldi завантажується перший операнд, в регістр r17 – другий операнд. Після виконання команди and результат записується в регістр r16 та виводиться в порт PORTD.

В подальшому регістри очищуються скиданням всіх бітів в 0, таким чином, підготовлюємо для завантаження нових значень.

Правильність виконання команд (стан лічильника, портів, байта стану) відслідковуємо за допомогою інструментальних панелей AVR Studio.

Текст програми виконання команди and

```
.include "8515def.inc"
.equ oper1=0b01010101
.equ oper2=0b00110100
;Логічне множення
loop:
    ser r18              ;Установка бітів регістра r18 в 1
   out DDRD,r18          ;Підготовка порту D на виведення
   ldi r16, oper1        ;Завантаження oper1 в r16
    ldi r17, oper2       ; Завантаження oper2 в r17
    and r16,r17          ;Логічне множення r16 та r17. Результат – в r16
    out PORTD, r16       ;Виведення в порт D
    clr r16              ;Готуємо r16 для завантаження нових значень
   clr r17               ;Готуємо r17 для завантаження нових значень
    out PORTD, r16       ; Готуємо порт D для виводу нових значень
    rjmp loop                    ;
```

Для реалізації виконання команди and обрано МК AT90S8515.

На рисунку 3.1 зображено завантаження проекту у робоче вікно програми.

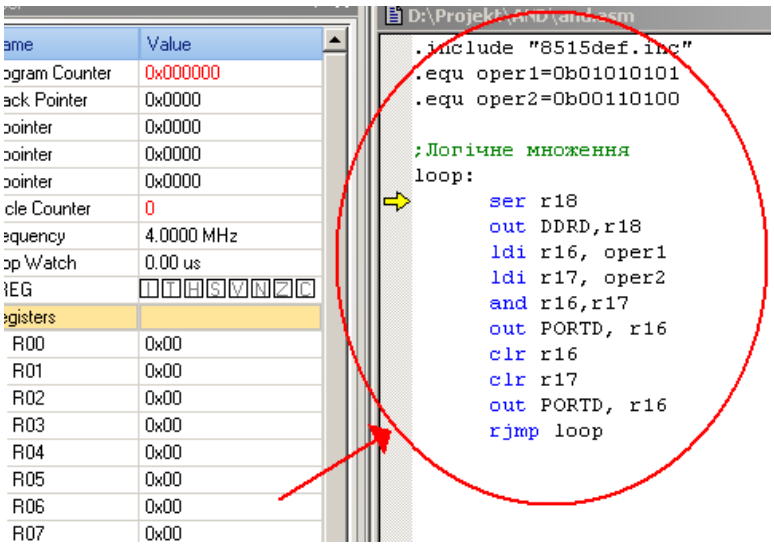


Рисунок 3.1 – Завантаження проекту у робоче вікно програми

На рисунку 3.2 зображено привласнення іменам змінних користувача числових значень.

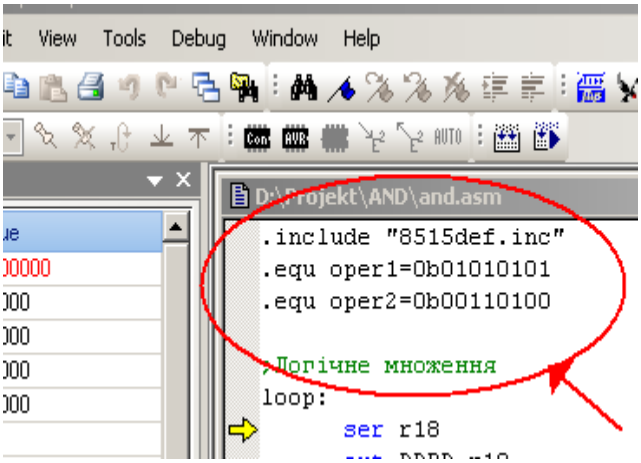


Рисунок 3.2 – Призначення іменам змінних користувача числових значень

На рисунку 3.3 зображено результат виконання команди завантаження регістрів r16 та r17.

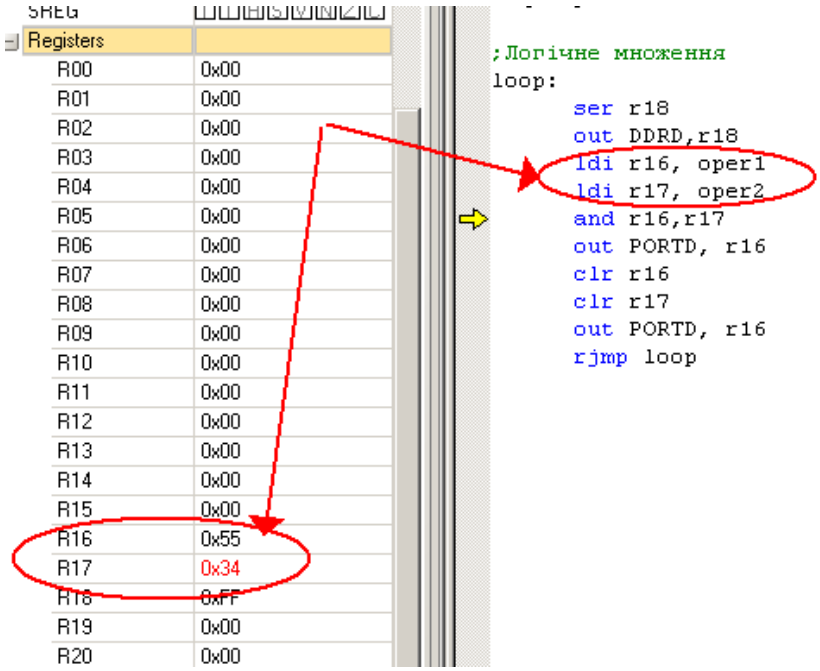


Рисунок 3.3 – Результат виконання команди завантаження регістрів r16 та r17

На рисунку 3.4 зображено результат виконання логічної команди and над вмістом регістрів r16 та r17.

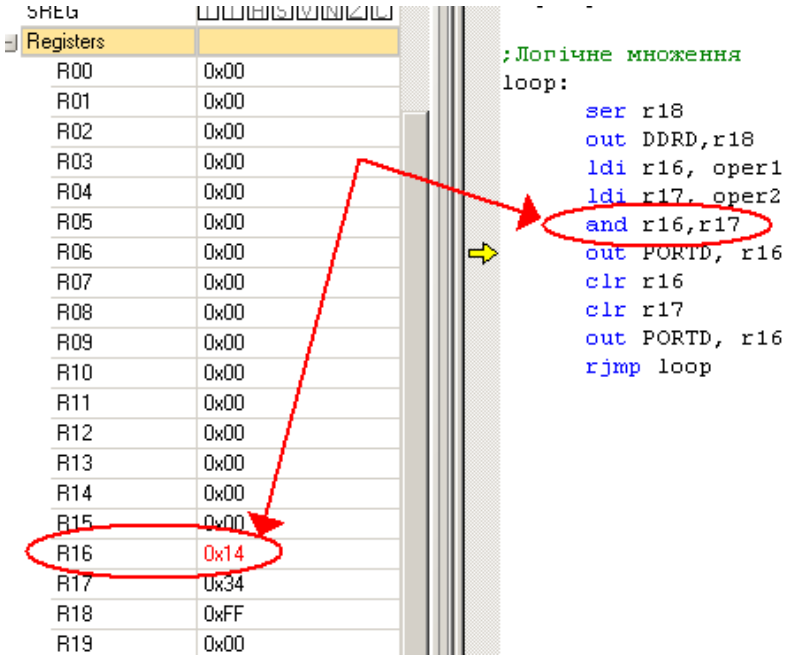


Рисунок 3.4 – Результат виконання логічної команди and

На рисунку 3.5 зображено виведення результату команди and в Port D.

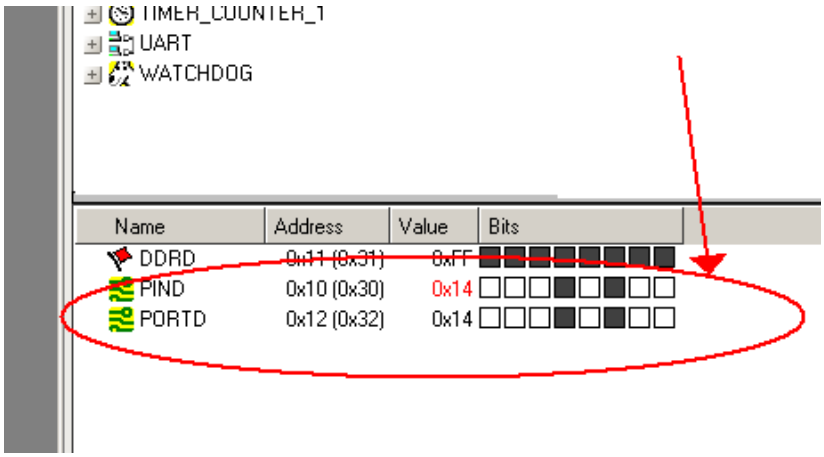


Рисунок 3.5 – Виведення результату команди and в PORTD

За результатами виконання прикладу проходить виконання інших команд логічних операцій, що наведенні вище.

Слід відмітити, що безпосередньо значення операндів можна завантажувати тільки в регістри з R16 з R31.

Регістри з R1 з R15 доступні тільки в командах завантаження через регістри з R16 з R31.

Діагностика виконання наступних команд відбувається по тому ж самому алгоритму, що і виконання команди and.

* **Примітка.** В регістри R0-R15 не можна безпосередньо завантажити значення змінних. В такому випадку необхідно використати тимчасовий регістр з діапазону R16-R31, а потім застосувати команду пересилання (mov) між регістрами. Наприклад, mov R1, R16.

3.3 Діагностика портів вводу-виводу

Оскільки, основна робота мікроконтролера пов’язана з портами вводу-виводу, тому їхньому тестуванню приділяються особлива увага. Мікроконтролери AVR в переважній своїй більшості використовуються для керування різноманітними пристроями і отримують дані зовні через свої порти, кількість яких визначається моделлю МК [3].

МК AVR мають широкі можливості щодо вводу-виводу даних.

МК серії ATx8515 мають чотири паралельних 8-розрядних порти Px (x=A, B, C, D) і один 3-розрядний порт PE (у моделі ATmega8515). Всі лінії портів можуть програмуватися на введення або на виведення даних незалежно один від одного і підключатися через внутрішні опори номіналом 35...120 кОм до шини живлення VCC.

На рисунку 3.6 наведений загальний вигляд мікросхем МК серії ATx8515 [6].

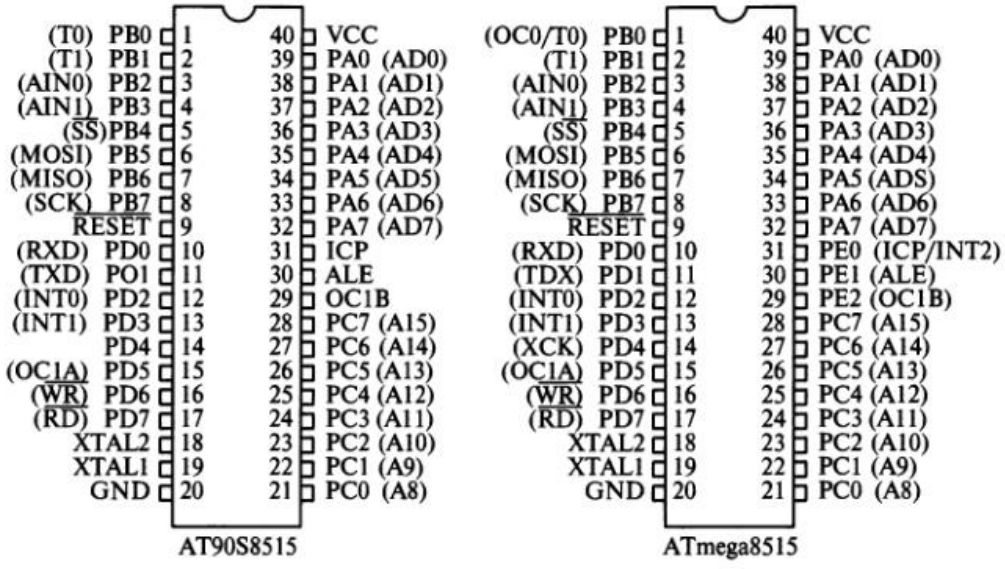


Рисунок 3.6 – Загальний вигляд мікросхем МК серії ATx8515

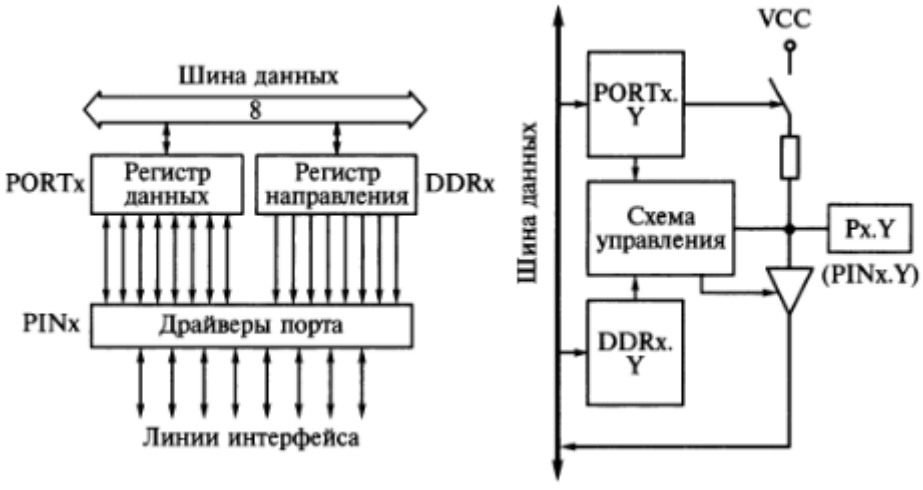
До складу кожного порту Px входять 3 регістра з іменами DDRx, PORTx і PINx. У МК ATx8515 регістр PINx не має апаратної реалізації. Це ім'я використовується для читання ліній інтерфейсу.

Стан розрядів DDRx.Y визначає напрям передачі бітів даних через контакти порту Px.Y. При DDRx.Y= 0 контакти порту Px.Y є входом, при DDRx.Y=1 – виходом. У режимі входу стан розрядів визначає стан контактів Px.Y. При PORTx.Y = 1 контакти порту через внутрішні опори підключаються до шини живлення VCC. При PORTx.Y=0 опори відключаться, виходи знаходиться в високоімпедансному стані (Z-стан).

У режимі виходу стан розрядів PORTx.Y визначає значення сигналу на контактах Px.Y. При PORTx.Y= 0 на контактах встановлюється напруга низького рівня (лог. 0), при PORTx.Y=1 – високого рівня (лог. 1).

При початковому пуску і перезапуску МК всі розряди регістрів DDRx і PORTx скидаються в нульовий стан, внаслідок чого контакти портів працюють в режимі входу і знаходяться в Z-стані.

На рисунку 3.7 наведена загальна структурна схема 8-розрядних портів Px і структурна схема одного розряду порту Px.Y (Y=0 ... 7) МК AT90S8515 [1].



а) б)

Рисунок 3.7 – Загальна структурна схема 8-розрядних портів Px (а) і структурна схема одного розряду порту Px.Y (б) МК AT90S8515

При спільному використанні всіх розрядів порту для введення байта даних використовуються команди IN Rd, PINx, для виведення – OUT PORTx, Rr (r=0 … 31). Значення вихідного сигналу на окремому контакті порту можна задати за допомогою команд установки в 0 (CBI PORTx.Y) і 1 (SBI PORTx.Y). Вхідний сигнал на сигналу на окремому контакті порту можна перевірити використовуючи команди умовного переходу SBIC PINx, Y або SBIS PINx, Y, які передбачають пропуск наступної команди за нульовим або одиничного значенням Px.Y.

У додатку А наведена діагностична програма взаємодії МК з кнопками і світлодіодами, яка при натисканні кнопки START виконує почергове перемикання світлодіодів, при натисканні кнопки STOP зупиняє перемикання і відновлює при повторному натисканні кнопки START.

У програмі для МК AT90S8515 використовується файл визначень 8515def.inc. У програмі лінії порту PB використані для індикації і, отже, ініціалізовані на виведення, а лінії 0 і 1 порту PD, що з'єднані з кнопками, – на введення (рис. 3.8).

Після натискання кнопки START починається послідовне перемикання світлодіодів з затримкою і перевірка стану кнопки STOP.

Використовуючи команди роботи з портами, розробимо програму роботи «кодового замка», яка після набору 4-х розрядного PIN-коду за допомогою кнопок К1-К4, що підключені до порту B (біти 0-3), та вводу при натисненні кнопки К5, що підключена до порту B (біт 4) здійснює порівняння з паролем та включає світлодіод, що підключено до 0-розряду порту D (додаток Б).

Якщо введено невірний код, то необхідно включити світлодіод, що підключено до 1-розряду порту D (додаток Б).

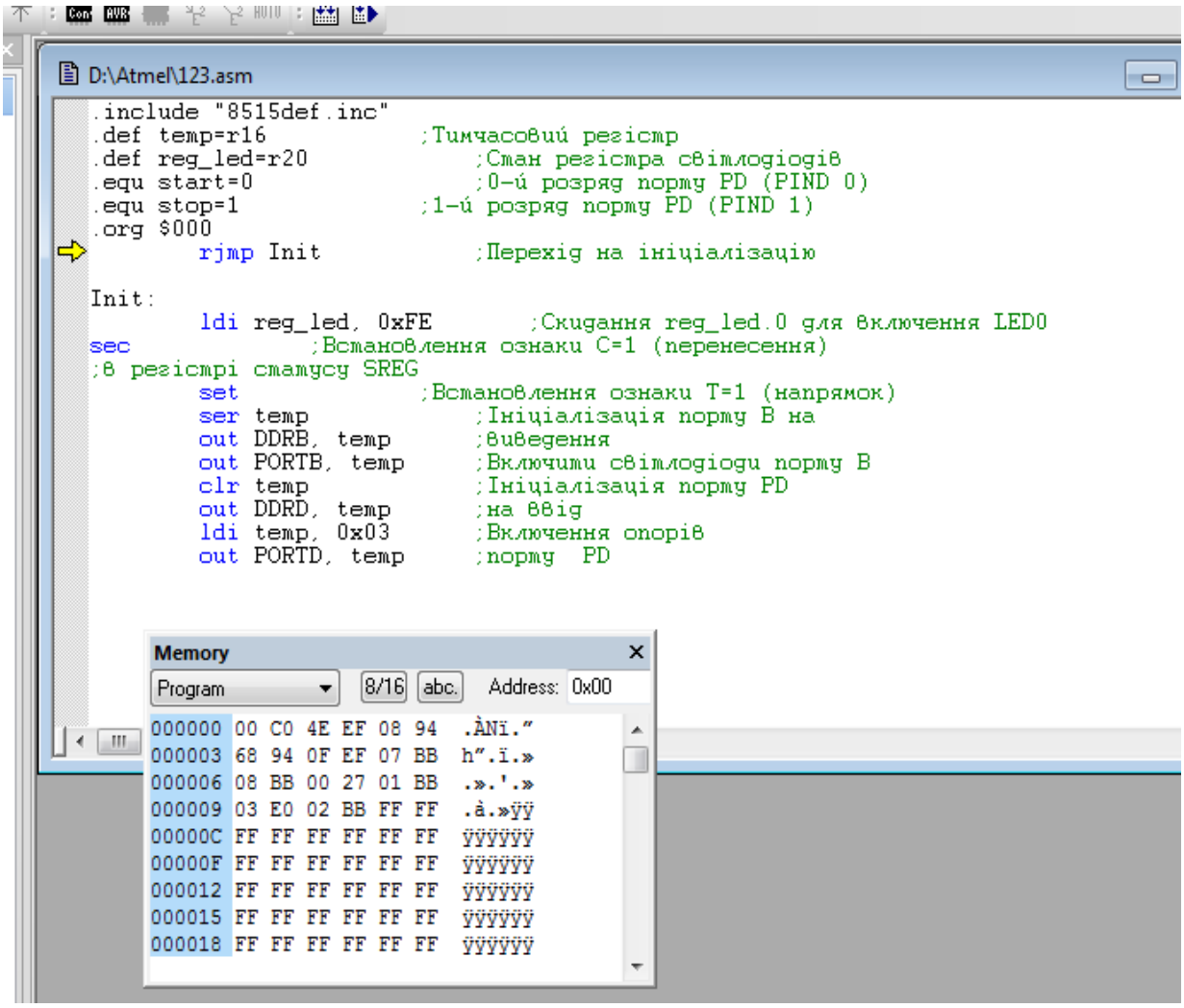


Рисунок 3.8 – Фрагмент виконання програми взаємодії МК з кнопками і світлодіодами

У таблиці 3.4 наведені PIN-код та кількість спроб вводу.

Таблиця 3.4 – PIN-код та кількість спроб вводу

PIN-код	Кількість спроб	PIN-код	Кількість спроб
0001	2	1000	3
0010	3	1001	4
0011	4	1010	2
0100	2	0110	4
0101	3	0111	2
1011	3	1100	4
1101	4	1110	2

На рисунку 3.9 наведено фрагмент програми вводу коду з порту B.

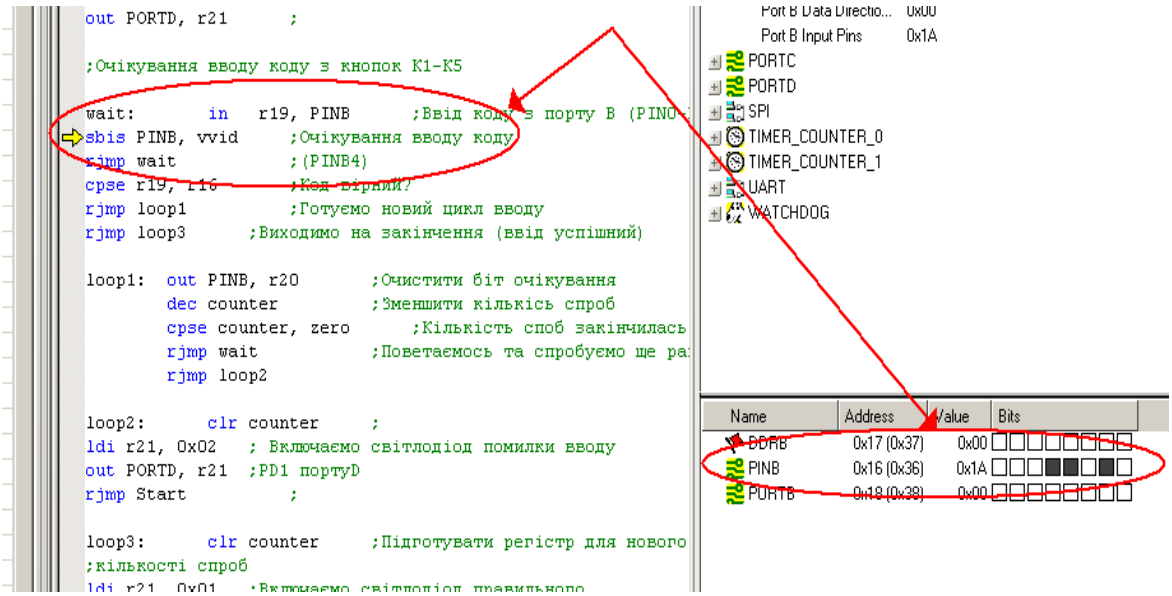


Рисунок 3.9 – Фрагмент програми вводу коду з порту B

3.4 Діагностика роботи таймера МК AVR

Режими роботи таймера безпосередньо пов’язані з системою переривань МК AVR фірми Atmel. Досить часто при роботі в реальних побутових пристроях таймери МК AVR використовуються в режимі лічильника подій [5].

МК AVR в залежності від класу (Tiny, Classic, Mega) і типу моделі мають у своєму складі від одного до трьох таймерів/лічильників загального призначення – T0, T1 і T2.

Розглянемо таймери МК AVR фірми Atmel на прикладі серії ATx8515.

Перший таймер (8-розрядний T0), що є у всіх моделях може використовуватися для відліку і вимірювання часових інтервалів або як лічильник зовнішніх подій, а в моделях ATmega8515 ще й для порівняння із заданим значенням. При переповненні лічильника регістра таймера генерується запит на переривання. Два інших таймера (16-розрядний T1 і 8-розрядний T2), крім вже названих, мають додаткові функції. Обидва таймера

можуть генерувати запит на переривання не тільки при переповненні лічильника регістра, а й при настанні ряду подій. Вони можуть використовуватися як широтно-імпульсні модулятори. Крім того, таймер Т2 може працювати в асинхронному режимі.

Кожен таймер/лічильник використовує один або більше контактів мікроконтролера. Ці контакти можуть бути або лініями портів вводу-виводу з альтернативною функцією, або виділеними контактами МК.

При використанні портів вводу-виводу необхідно провести конфігурацію контактів у відповідності з їх функціональним призначенням (вхід або вихід).

У всіх МК сімейства AVR є також сторожовий таймер, що є обов’язковим атрибутом сучасних МК.

Цей таймер використовується для запобігання циклічного повторення програми.

Усі контакти МК ATx8515, що пов'язані з таймерами/лічильниками, і їх функції наведено в таблиці 3.5 [4].

Таблиця 3.5 – Контакти МК, що використовуються таймерами/лічильниками загального призначення

Найменування	ATx8515	Призначення
T0	PB0	Вхід зовнішнього сигналу таймера T0
T1	PB1	Вхід зовнішнього сигналу таймера T1
ICP	ICP/P0*	Вхід захвату таймера T1
OC1A	PD5	Вихід схеми порівняння таймера T1
OC1B	OC1B//P2*	Вихід схеми порівняння таймера T1

*У МК AT90S8515 – виділений вихід, в ATmega8515 – вихід порту PE.

Таймер/лічильник T0 (8-розрядний) використовується для формування часових інтервалів або для підрахунку числа зовнішніх подій.

Структурна схема таймера/лічильника T0 МК AT90S8515 наведена на рисунку 3.10 [4].

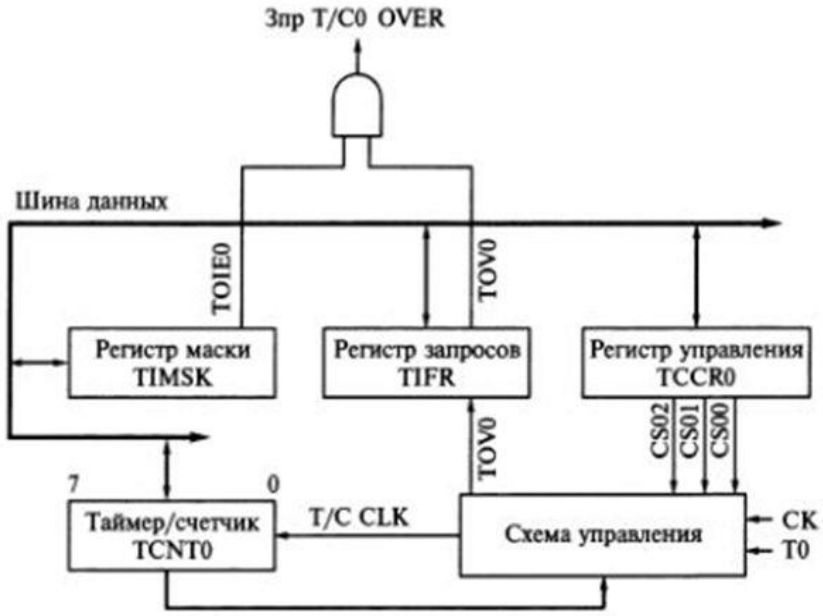


Рисунок 3.10 – Структурна схема таймера/лічильника T0 МК AT90S8515

Таймер/лічильник T0 може працювати у двох режимах:

– таймера. В цьому режимі на вхід поступають імпульси тактового сигналу мікроконтролера СК;

– лічильника подій. В цьому режимі збільшення значення лічильника (інкремент) здійснюється по активному фронту сигналу на вході T0 МК (лінія порту PB0).

Вибір режиму роботи (джерела тактового сигналу), а також запуск та зупинка таймера/лічильника здійснюється за допомогою розрядів CS00-CS02 регістра керування таймером TCCR0 (таблиця 3.6) [4].

Таблиця 3.6 – Формат регістра TCCR0

№ розряду	7	6	5	4	3	2	1	0
Ім’я	–	–	–	–	–	CS02	CS01	CS00

Решта розрядів регістра доступні тільки для читання та містять 0.

Відповідність між станом розрядів і режимом роботи таймера/лічильника Т0 наведено в таблиці 3.7 [4].

Таблиця 3.7 – Вибір джерела тактового сигналу для таймера/лічильника Т0

CS02	CS01	CS00	Джерело тактового сигналу
0	0	0	Таймер/лічильник зупинено
0	0	1	СК (тактовий сигнал мікроконтролера)
0	1	0	СК/8
0	1	1	СК/64
1	0	0	СК/256
1	0	1	СК/1024
1	1	0	Контакт Т0, інкремент лічильника відбувається по задньому фронту імпульсів (перехід з 1 в 0)
1	1	1	Контакт Т0, інкремент лічильника відбувається по передньому фронту імпульсів (перехід з 0 в 1)

Кількість вхідних імпульсів можна обрати довільно. В даній реалізації діагностичної програми обрано 4 вхідних імпульси.

На рисунку 3.11 наведено вікно програми, у якому зображений стан регістрів TCCR0 та TIMSK таймера.

Як було наголошено, режими роботи таймера безпосередньо пов’язані з системою переривань. Для цього у складі МК є спеціальний регістр – регістр статусу МК.

Як показано на рис. 3.12, за переривання відповідає 8-й біт цього регістра (I –Interrupt).

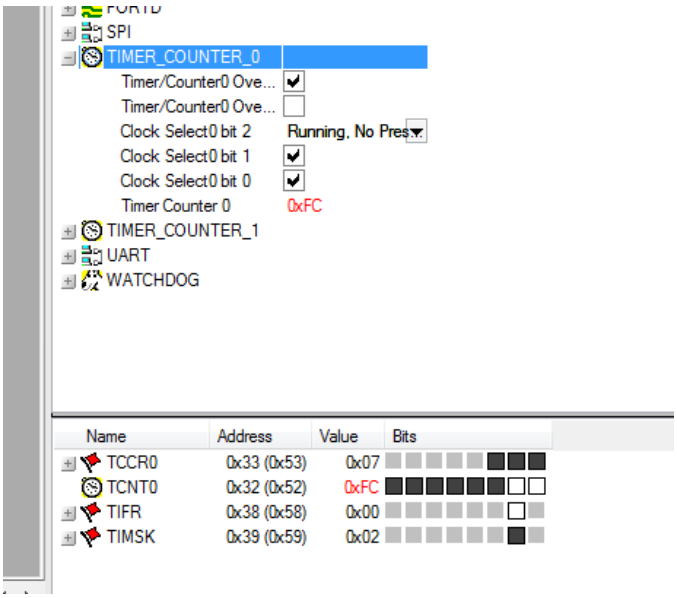


Рисунок 3.11 – Стан регістрів TCCR0 та TIMSK таймера

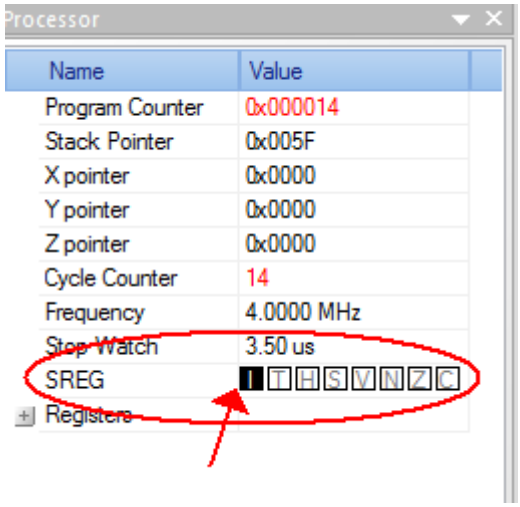


Рисунок 3.12 – Регістр статусу SREG

Процедура встановлення цього переривання в даному випадку наступна. Записуємо в регістр маски (TIMSK) код 02, що відповідає за переривання, якщо відбудеться переповнення таймера. Це в свою чергу дозволяє глобальне переривання в регістрі SREG (рис. 3.13).

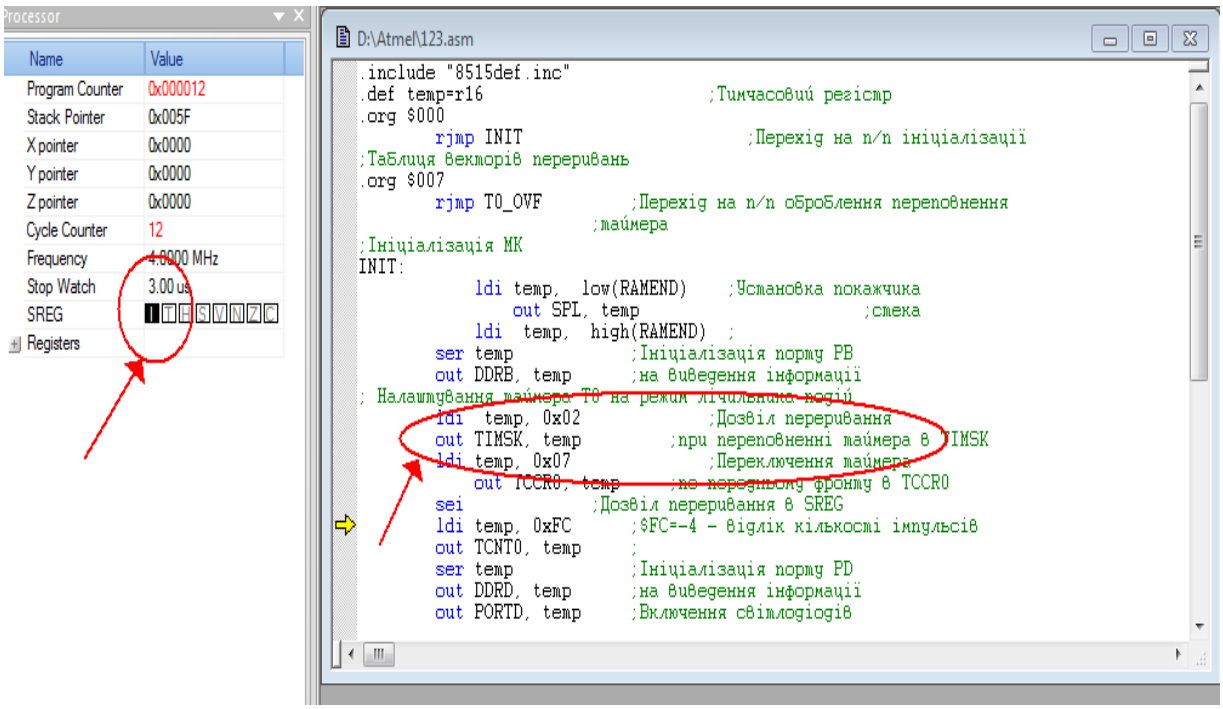


Рисунок 3.13 – Фрагмент програмування регістра маски таймера

Слід відмітити, що у програмі наведеній у додатку В, підрахунок вхідних імпульсів ведеться по передньому фронту вхідного імпульсу. Для розробника є можливість, на вибір, підраховувати вхідні імпульси і по задньому фронту.

3.5 Діагностика виконання команд зсуву

Для зміни напрямку зсуву інформації, що занесена в регістри МК AVR, використовують команди зсуву.

Зазвичай, команди зсуву у мікропроцесорах використовуються для обчислення степеневих функцій, тобто збільшення/зменшення числа.

З кожним зсувом вліво/вправо число збільшується/зменшується в 2 рази, тобто реалізується функція:

$$y = 2^n, \tag{2.1}$$

де n= 1,…8.

МК, звичайно, не володіють такими потужними обчислювальними можливостями, оскільки і апаратні ресурси і система команд їх є набагато «скромнішими».

Прикладом використання команд зсуву МК на практиці можуть бути різноманітні стенди з ефектами мерехтіння, новорічні гірлянди, інформаційні табло тощо.

В таблиці 3.8 наведено перелік команд зсуву.

Таблиця 3.8 – Перелік команд зсуву

Мнемоніка команди	Операція
asr	арифметичного зсуву вправо
lsr	логічного зсуву вправо
lsl	логічного зсуву вліво
ror	циклічного зсуву вправо
rol	циклічного зсуву вліво

Система числення – двійкова. Для введення значення використати порт D МК, для виводу – порт B.

Команди, що наведені в таблиці 2.8, належать до групи команд роботи з бітами та тестувань бітів. Мнемоніка запису команди зсуву виглядає наступним чином:

<команда> <регістр>

Наприклад, команда арифметичного зсуву вправо вмісту регістру r16 має вигляд:

asr r16

На рисунку 3.14 наведено графічне зображення виконання команд зсуву [1].

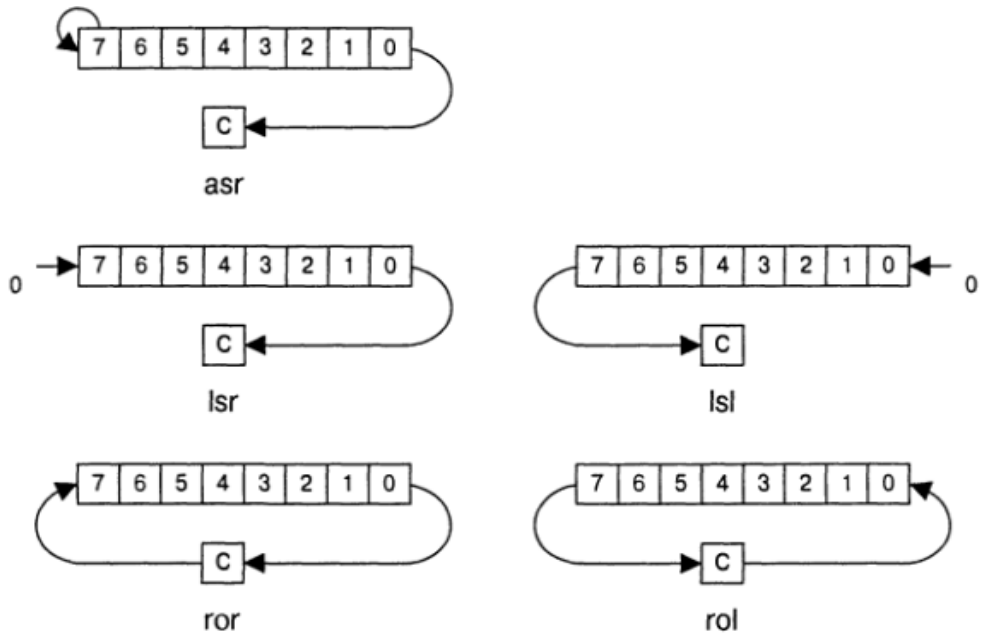


Рисунок 3.14 – Графічне зображення виконання команд зсуву

При арифметичному зсуві вправо всі біти регістра зсуваються вправо, при цьому біт 7 не змінюється, а біт 0 завантажується в прапорець переносу (біт C регістра SREG).

При логічному зсуві вправо всі біти регістра зсуваються вправо, а біт 7 скидається в 0.

Циклічний зсув вправо здійснюється через прапорець переносу, тобто значення прапорця заноситься у біт 7 регістра, а вміст 0-го біту – у прапорець переносу.

Наведемо приклад виконання програми команди логічного зсуву вліво:

```
.include "8515def.inc"
;Визначення регістрів
.def temp=r16              ;Регістр тимчасового зберігання
;Початок програми
Start:
ser temp
out DDRB, temp             ;Порт B – на вивід
in temp, pinD              ;Ввід коду
```

```
lsl temp                ;Логічний зсув вліво
out PortB, temp         ;Вивід результату
clr temp                ;Підготовка регістру та
out PortB, temp         ;порту до виконання нової команди
rjmp Start              ;Повертаємось до мітки Start
```

Виконання команди практично до двох дій – завантаження операнда в регістр (у даному випадку R16) та власне виконання команди зсуву.

Така проста реалізація робить виконання таких операцій дуже швидкими та вимагає всього один машинній такт роботи МК

Оскільки після скидання МК у всіх розрядах регістра DDRD порту D встановлюється 0, то нема необхідності програмувати його на введення інформації.

У якості порту виведення у програмі використовується порт B (рис. 3.15).

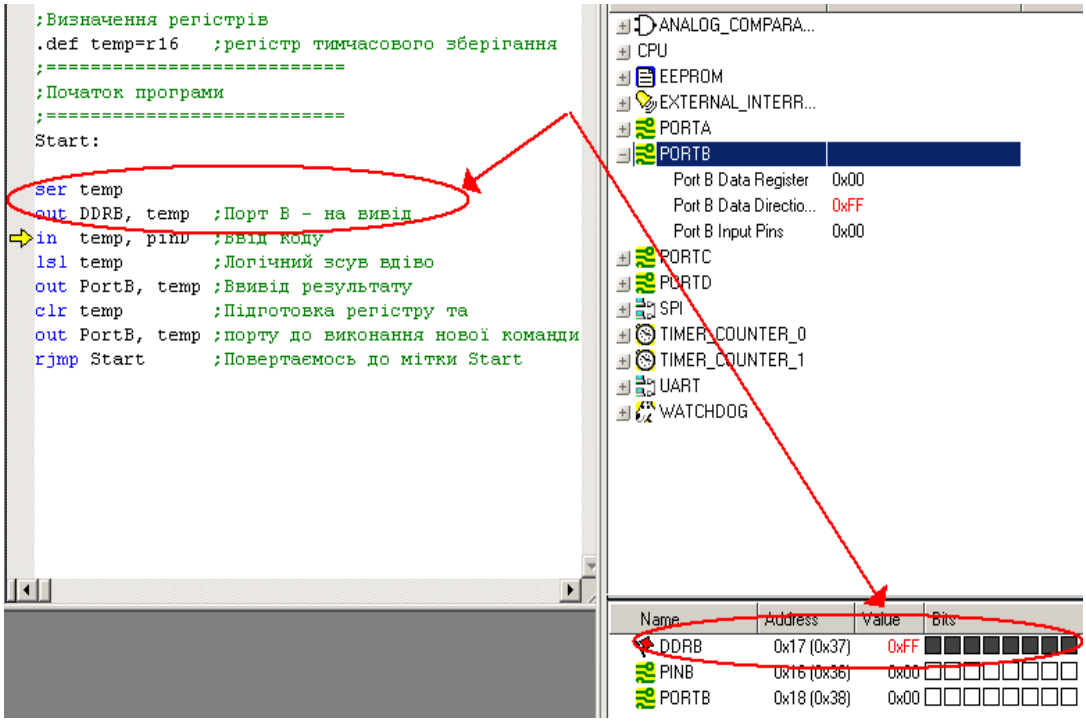


Рисунок 3.15 – Фрагмент ініціалізації порту B на виведення

На рисунку3.16 наведений результат завантаження операнда з порту D в регістр r16.

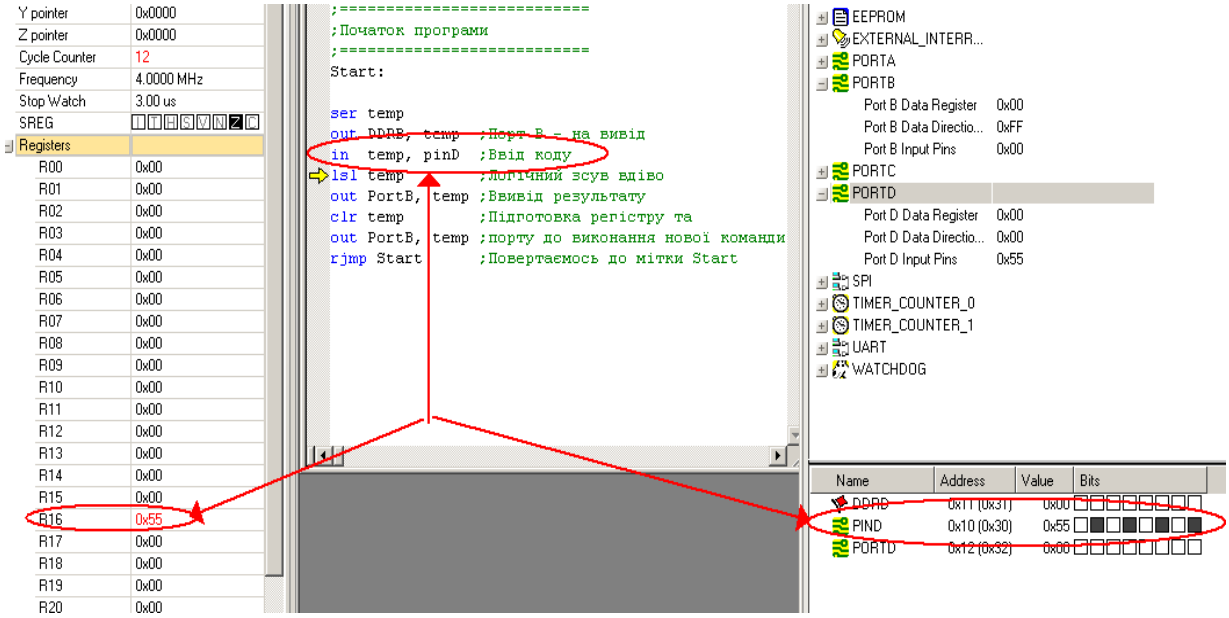


Рисунок 3.16 – Результат завантаження операнда в регістр r16

На рисунку 3.17 наведений фрагмент результату виконання команди логічного зсуву вліво.

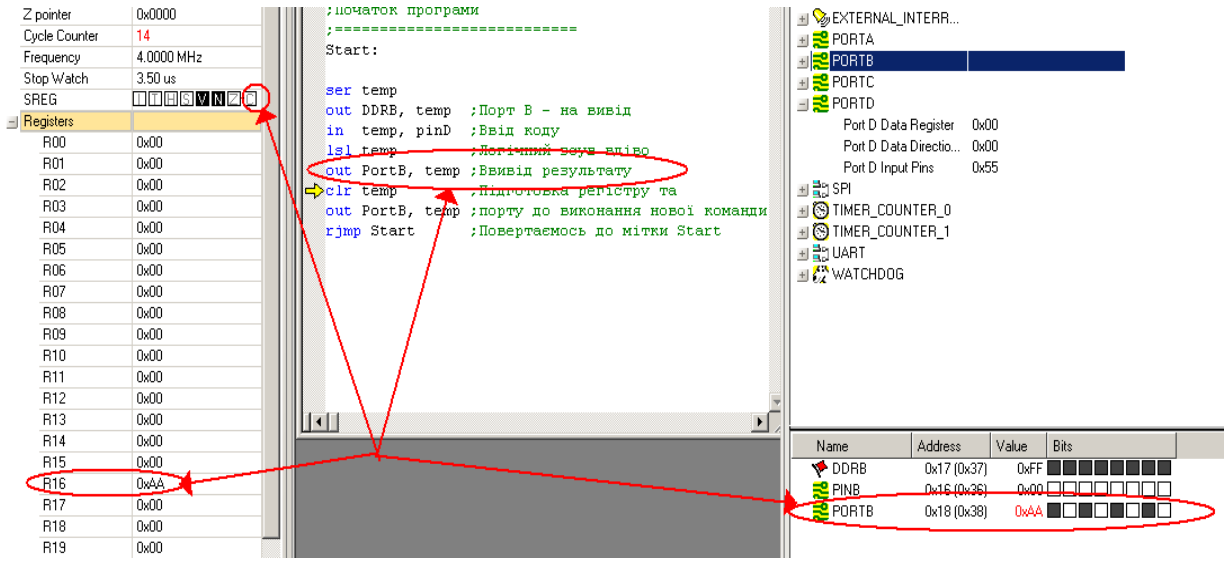


Рисунок 3.17 – Результат виконання команди логічного зсуву вліво

Розглянемо практичне використання команд зсуву на прикладі пристрою, що формує включення світлодіодів з зсувом свічення, наприклад, вліво, т. зв. «вогник, що біжить» (рис. 3.18).

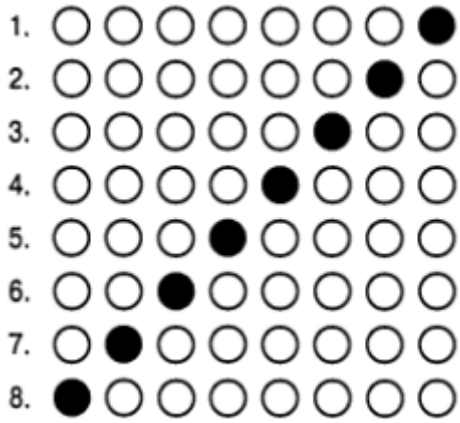


Рисунок 3.18 – Зсув свічення світлодіодів вліво

Для виконання такої команди необхідно встановити біт 0 відповідного порту в 1, а потім виконати команду логічного зсуву (lsl) вліво 8 разів.

Приклад програмно-апаратної реалізації застосування команд зсуву у світлодіодних пристроях покажемо на мікроконтролері AT90S1200 (рис. 3.19).

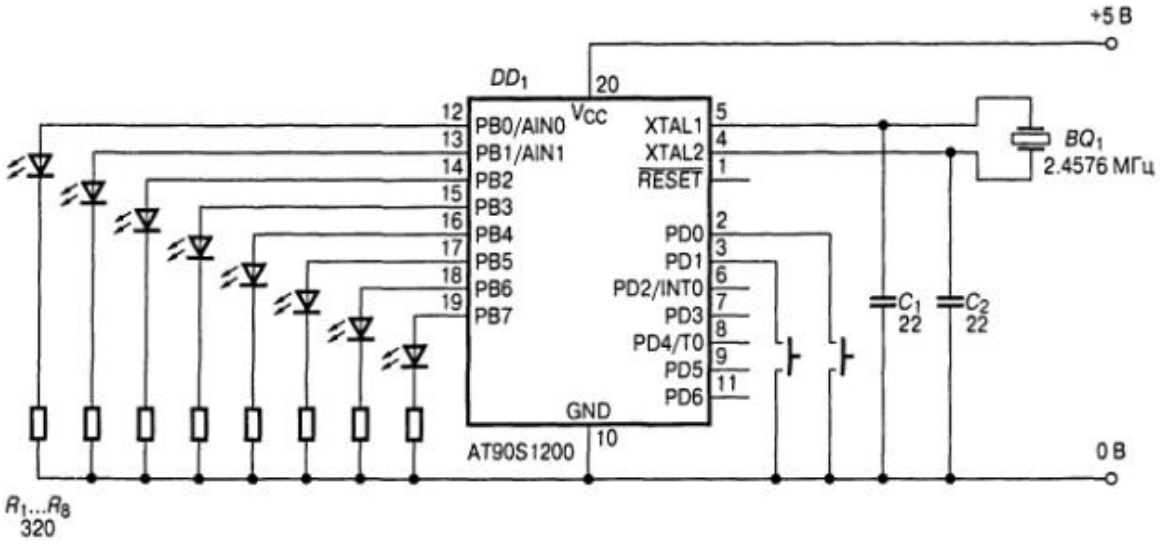


Рисунок 3.19 – Електрична схема пристрою на базі МК AT90S1200

Секція світлодіодів підключена до контактів порту PB0-PB7 МК.

Оскільки світлодіоди підключені до контактів PB0-PB7 (PinB0-PinB7), а звідти поступають регістр PortB, тому неможливо напряму застосувати

команду зсуву безпосередньо до вмісту регістра PortB. Для цього необхідно завантажити поточний стан світлодіодів в тимчасовий регістр (temp), який слід обрати з діапазону r16-r31, виконати зсув вмісту регістра temp і потім записати це значення в PortB.

У програмі використовується таймер TCCR0, що задає частоту мерехтіння світлодіодів.

3.6 Завантаження діагностичних програм в пам'ять МК AVR

Кінцевим етапом є завантаження діагностичних програм в пам'ять EEPROM МК AVR.

Відмінною особливістю МК сімейства AVR є те, що в них як пам'ять програм використовується одна і та ж пам'ять типа FLASH–EEPROM, яка може бути різного об'єму, вільно програмуватись користувачем і знову витиратися електричним способом [6].

МК сімейства AVR з вбудованою флеш–пам'яттю EPROM економлять не лише місце на платі електронного пристрою, але також представляють в розпорядження користувача всі контакти вводу/виводу МК. Також може відпасти необхідність і в колодці для зовнішньої пам'яті EPROM.

Організація пам’яті та регістрів пам’яті МК AVR наведена в [2].

Програмна реалізація запису та верифікації пам'яті EEPROM МК AVR наведена у додатку Е.

В такому вигляді програма буде читати з виводів порту С код $FF, оскільки вони находяться в третьому стані. Тому попередньо на виводах порту С треба встановити довільний код в межах $00…$FF (рисунок 3.20).

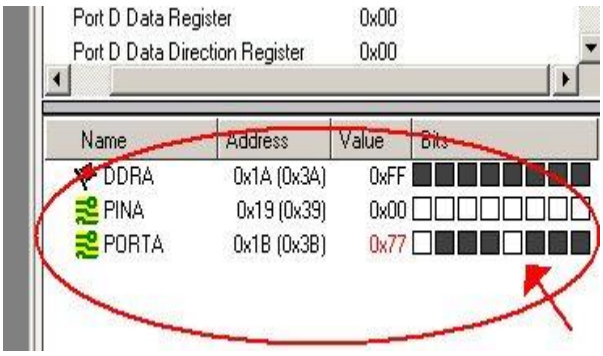


Рисунок 3.20 – Завершення програми – видача коду в порт А

Підпрограма EEWrite (див. додаток Е) за допомогою циклічного опитування розряду EEWE чекає готовності пам'яті EEPROM до нового програмування. Як тільки розряд EEWE переходить в стан лог. 0, в регістри EEARH, EERAL і EEDR будуть записані відповідно два байти адреси EEPROM і байт даних, що підлягає програмуванню. За допомогою установки розряду EEMWE буде дозволене програмування.

Тепер у програмі користувача є час тривалістю 4 періоди такту системної синхронізації для запуску процесу програмування за допомогою установки розряду EEWE.

У розглянутому вище прикладі це відбувається вже при виконанні наступної команди. Якщо вихід з підпрограми здійснюється по команді ret, то процес програмування продовжується далі.

Підпрограма EERead за допомогою циклічного опитування розряду EEWE чекає закінчення процесу програмування (якщо він в даний момент активний). Як тільки розряд EEWE перейде в стан лог. 0, в регістри EEARH і EEARL будуть записані 2 байти адреси пам'яті EEPROM, що підлягають читанню. Після цього для початку процесу читання буде встановлений розряд EERE.

У зв'язку з тим, що такт системної синхронізації в мікроконтролері AT90S8515 повинен складати максимум 8 МГц, тут немає необхідності встановлювати розряд EERE 2 рази, як це відбувається в мікроконтролері

AT90S1200, для того, щоб надати в розпорядження апаратної частини досить часу для читання комірки пам'яті EEPROM. Необхідний байт знаходиться в області вводу/виводу, що відповідає регістру EEDR, і переписується для подальшої обробки в робочий регістр EEDRD.

Основна програма на початку проводить ініціалізацію покажчика стеку та портів A і C, що використовуються для вводу/виводу. Після цього читається байт, що присутній на виводах порту C, та за допомогою підпрограми EEWrite записується за адресою $100 пам'яті EEPROM. Потім підпрограма EERead читає дані в робочий регістр EEdrd пам'яті EEPROM за адресою $100. Цей байт надалі передається в порт A.

Подальше виконання програми показане у вигляді циклу.

На рисунку 3.21 – 3.24 наведені фрагменти виконання програми програмування пам’яті EEPROM.

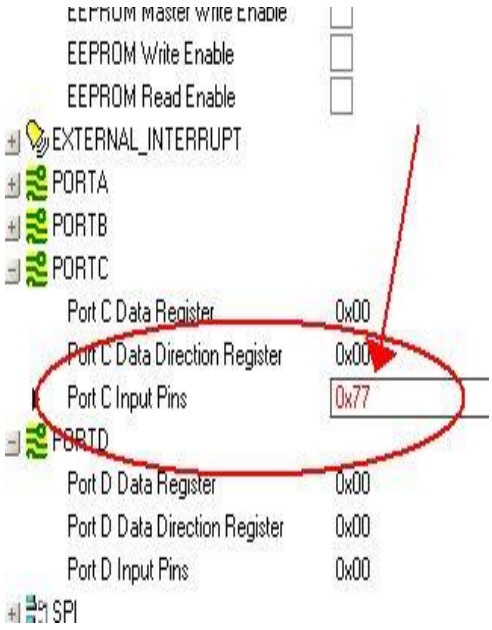


Рисунок 3.21 – Попередня установка коду для запису

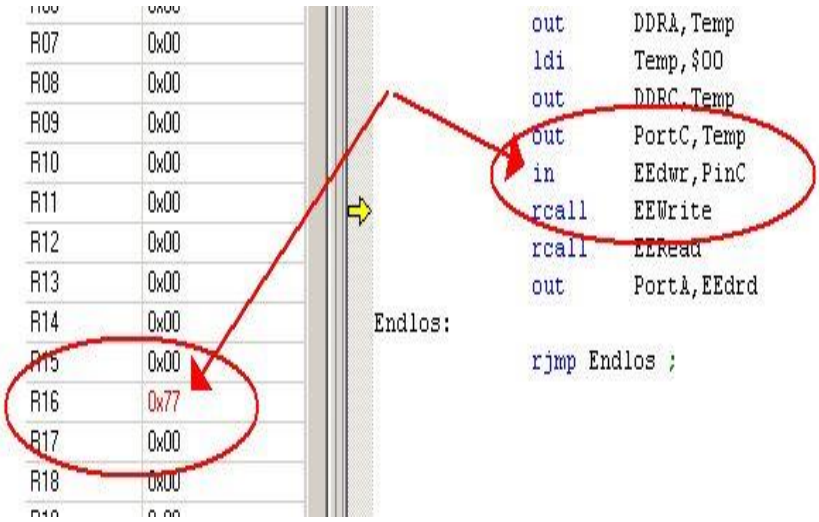


Рисунок 3.22 – Завантаження коду з виводів порту С та зберігання в регістрі R16

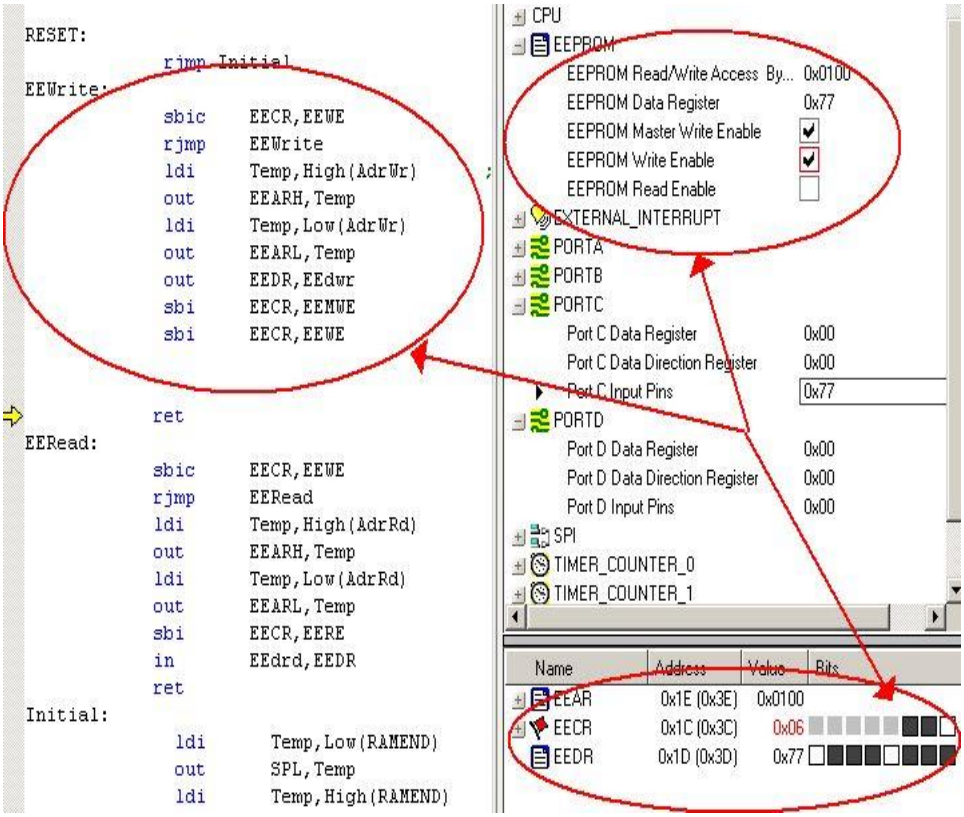


Рисунок 3.23 – Програмування комірки $100 пам’яті EEPROM

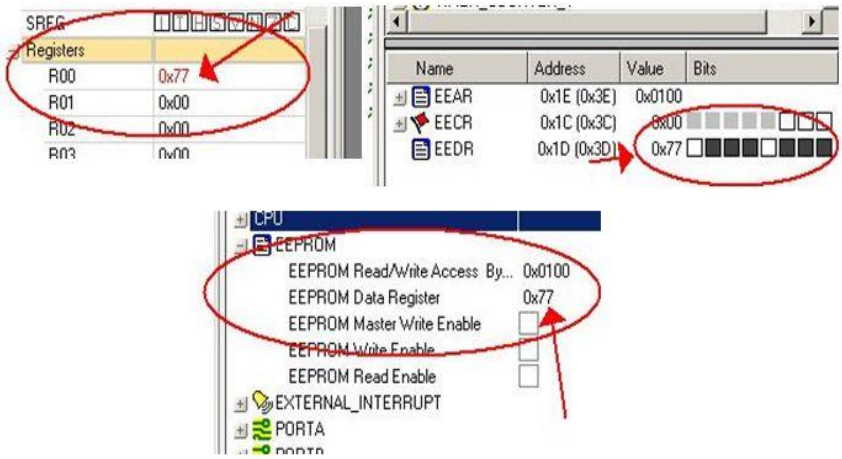


Рисунок 3.24 – Перевірка процедури запису комірки $100 пам’яті EEPROM

3.7 Висновки до розділу 3

У третьому розділі дипломного проекту здійснено розроблення діагностичних програм для мікроконтролерів AVR фірми Atmel.

З метою розроблення проаналізовано виконання команд з застосуванням асемблера в інструментальному середовищі AVR Studio, визначено перелік команд логічних операцій, команд роботи з портами введення/виведення.

Для виконання програмно-діагностичної реалізації використано мікроконтролери AT90S8515 та AT90S1200 фірми Atmel.

При реалізації діагностики таймера проведено огляд внутрішніх регістрів таймера, визначені режими його роботи, проаналізована внутрішня структура регістрів таймера.

На прикладі підключення світлодіодів до МК AT90S1200 показано застосування команд зсуву.

На кінцевому етапі реалізації проекту проведення завантаження діагностичних програм в пам'ять МК AVR з детальним аналізом регістрів пам’яті.

3.8 Перелік джерел посилань до розділу 3

1. Мікроконтролери AVR. [Електронний ресурс]. Режим доступу https://msn.khnu.km.ua/pluginfile.php/304356/mod_resource/content/1/SAU_LEK2.pdf (дата звернення: 09. 05. 2020).

2. Пристрій і структура мікроконтролерів AVR. [Електронний ресурс]. Режим доступу http://ua.nauchebe.net/2011/11/pristrij-i-struktura-mikrokontroleriv-avr/ (дата звернення: 19. 05. 2020).

3. . RISC-мікроконтролери сімейства AVR. . [Електронний ресурс]. Режим доступу https://microchipinf.com/ua_articles/55 (дата звернення: 16. 05. 2020).

4. Використання таймерів мікроконтролерів для реалізації періодичних обчислень. [Електронний ресурс]. Режим доступу https://microchipinf.com/ua_articles/51/676 (дата звернення: 17.05.2020).

5. Среда разработки AVR Studio для микроконтроллеров AVR и Arduino. [Електронний ресурс]. Режим доступу https://arduinoplus.ru/avr-studio-sreda-razrabotki/ (дата звернення: 21.05.2020).

6. Atmel Start. [Електронний ресурс]. Режим доступу https://start.atmel.com (дата звернення: 22.05.2020).

4 ОХОРОНА ПРАЦІ

4.1 Загальні питання з охорони праці

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

Завданням даної дипломної роботи є розроблення діагностичних програм для мікроконтролерів фірми Atmel, і як результат було створено діагностичні програми тестування логічних команд, портів введення-виведення, таймерів, команд зсуву та ряд інших За цим завданням в подальшому розроблятиметься реальна система, яка значно полегшить процес діагностування мікроконтролерів. Так як в процесі проектування використовувалося інструментальне середовище налаштування МК AVR Studio, то аналіз потенційно небезпечних і шкідливих виробничих чинників виконується для персонального комп'ютера на якому буде розроблятися/використовуватися розроблена програма.

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності [1].

При роботі з обчислювальною технікою змінюються фізичні і хімічні фактори навколишнього середовища: виникає статична електрика, електромагнітне випромінювання, змінюється температура і вологість, рівень вміст кисню і озону в повітрі.

4.2 Правові та організаційні основи охорони праці

Державна політика в галузі охорони праці визначається відповідно до Конституції України Верховною Радою України і спрямована на створення належних, безпечних і здорових умов праці, запобігання нещасним випадкам та професійним захворюванням. Відповідно до статті 3 [1] законодавство про охорону праці складається з Закону, Кодексу законів про працю України, Закону України "Про загальнообов'язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захворювання, які спричинили втрату працездатності" та інших правових актів.

Обов’язки працівників щодо додержання вимог нормативно-правових актів з охорони праці, відповідальність робітників всіх категорій за порушення вимог (ст. 44) [1] щодо охорони праці та структура організації/виробництв системи управління охорони праці визначені безпосередньо «Інструкцією на робоче місце № 1», та іншими затвердженими власними нормативними актами з питань охорони праці (правилами, нормами, регламентами, положеннями, стандартами, інструкціями та іншими документами, обов’язковими до виконання), тобто тих, що діють на підприємстві/організації, і визначені в [2].

4.3 Організаційно-технічні заходи з безпеки праці

В організації/підприємстві проводиться навчання і перевірка знань з питань охорони праці відповідно до [3].

Також впроваджені організаційні заходи з пожежної безпеки - навчання і перевірку знань відповідно до [3].

4.4 Аналіз стану умов праці та вимоги до приміщення

Робота над створенням дипломного проекту проходитиме в приміщенні відповідної установи (компанії, підприємстві тощо). Для даної роботи достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп'ютером. Геометричні розміри приміщення зазначені в таблиці 4.1.

Таблиця 4.1 – Розміри приміщення

Найменування	**Значення**
Довжина, м	5
Ширина, м	5
Висота, м	3
Площа, $м^2$	25
Об'єм, $м^3$	75

Згідно з ДСН 3.3.6.042-99 [4] розмір площі для одного робочого місця оператора персонального комп'ютера має бути не менше 6 кв. м, а об'єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам. Для забезпечення потрібного рівного освітленості кімната має вікно та систему загального рівномірного освітлення, що встановлена на стелі. Для дотримання вимог пожежної безпеки встановлено порошковий вогнегасник та систему автоматичної пожежної сигналізації відповідно [5, 13].

4.5 Вимоги до організації робочого місця

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця згідно за ДСанПІН 3.3.2.007-98 [6] і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність в таблиці 4.2.

Таблиця 4.2 - Характеристики робочого місця

Найменування параметра	**Фактичне значення**	**Нормативне значення**
Висота робочої поверхні, мм	750	680 ÷800
Висота простору для ніг, мм	730	не менше 600
Ширина простору для ніг, мм	660	не менше 500
Глибина простору для ніг, мм	700	не менше 650
Висота поверхні сидіння, мм	470	400 ÷500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	600	не менше 300
Ширина опорної поверхні спинки, мм	500	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	800	700 ÷ 800

Екран монітору знаходиться на відстані 0.8 м, клавіатура має можливість регулювання кута нахилу 5-15°. Отже, за всіма параметрами робоче місце відповідає [7].

Приміщення кабінету знаходиться на другому поверсі трьох поверхової будівлі і має об’єм 78 м^3, площу — 18 м^2. У цьому кабінеті обладнано три місця праці, з яких два укомплектовані ПК.

Температура в приміщенні протягом року коливається у межах 18–24°С, відносна вологість — близько 50%. Швидкість руху повітря не перевищує 0,2 м/с. Шум на робочому місці знаходиться на рівні 50 дБА. Система вентилювання приміщення — природна неорганізована, а опалення — централізоване.

Розміщення вікон забезпечує природне освітлення з коефіцієнтом природного освітлення не менше 1,5%, а загальне штучне освітлення, яке здійснюється за допомогою восьми люмінесцентних ламп, забезпечує рівень освітленості не менше 200 Лк.

У кабінеті є електрична мережа з напругою 220 В, яка створює небезпеку ураження електричним струмом. ПК та периферійні пристрої можуть бути джерелами електромагнітних випромінювань, аерозолів та шкідливих речовин (часток тонеру, оксидів нітрогену та озону).

За ступенем пожежної безпеки приміщення належить до категорії В. Кабінет має бути оснащений переносним вуглекислотним вогнегасником ВВК-5.

4.6 Навантаження та напруженість процесу праці

Під час виконання робіт використовують ПК та периферійні пристрої, що призводить до навантаження на окремі системи організму.

Найбільшому ризику виникнення різноманітних порушень піддаються: органи зору, м’язово скелетна система, нервово-психічна діяльність, репродуктивна функція у жінок.

При роботі наявні психофізіологічні небезпечні та шкідливі фактори: фізичне перевантаження, розумове перенапруження та ін.

Роботу за дипломним проектом визнано, таку, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви для розробників програм тривалістю 15 хв через кожну годину роботи.

4.7 Аналіз небезпечних та шкідливих факторів при роботі на персональному комп’ютері

Роботу, пов'язану з електронно-обчислювальними машинами (далі - ПК) з відео дисплейними терміналами (далі - ВДТ), у тому числі на тих, які мають робочі місця, обладнані ПК з ВДТ і периферійними пристроями (далі - ПП), виконують із забезпеченням виконання згідно [7], які встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ПК з ВДТ і ПП.

Переважно роботи за проектами виконують у кабінетах чи інших приміщеннях, де використовують різноманітне електрообладнання, зокрема ПК та периферійні пристрої.

Робочі місця мають відповідати вимогам [8,9].

Це передбачає, що визначена виробнича діяльність пов’язана з наявністю певної кількості небезпечних та/або шкідливих виробничих факторів згідно [10].

Робота ПК та периферійних пристроїв супроводжує виділення багатьох хімічних речовин, зокрема озону, оксидів нітрогену та аерозолів (високодисперсних частинок тонера).

4.8 Пожежна безпека

Пожежна безпека при застосуванні ПК забезпечується системою запобігання пожежі, протипожежного захисту та організаційно-технічними заходами

Згідно [10], таке приміщення, площею 25 м^2, відноситься до категорії "В" (пожежонебезпечної) та для протипожежного захисту в ньому проектом передбачено устаткування автоматичною пожежною сигналізацією із застосуванням датчиків-сповіщувачів РІД-1 (сповіщувач димовий ізоляційний) в кількості 1 шт., і застосуванням первинних засобів пожежогасіння. Відповідно до норм первинних засобів пожежогасінні пропонується використовувати вогнегасник порошковий ВП-2 в кількості 1 шт.

Простори усередині приміщень в межах, яких можуть утворюватися або знаходиться пожежонебезпечні речовини і матеріали відповідно [10] відносяться до пожежонебезпечної зони класу П-IIа.

Це обумовлено тим, що в приміщенні знаходяться тверді горючі та важкозаймисті речовини та матеріали. Приміщенню, у якому розташоване робоче місце, присвоюється II ступень вогнестійкості.

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор тощо.

При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідридні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол згідно [10].

Для захисту персоналу від дії небезпечних і шкідливих чинників пожежі проектом передбачається застосування промислового протигаза, що фільтрує, з коробкою марки «В» із сірою відміткою забарвлення – захист від неорганічних газів (хлор, фтор, бром, сірководень, сірковуглець, хлорциан, галогени), а цей фільтр не захистить від СО (тобто від чадного газу).

4.9 Електробезпека

На робочому місці периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту повинні відповідати [12].

Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування виконана як окрема групова трипровідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Електромережа штепсельних розеток для живлення персональних ПК, укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне

заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.10 Мікроклімат

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з'єднанням температури, вологості, швидкості переміщення повітря. В даному приміщенні проводяться роботи, що виконуються сидячи і не потребують динамічного фізичного напруження, то для нього відповідає категорія робіт Іа. Отже оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають [4] і наведені в таблиці 4.3.

Таблиця 4.3 – Норми мікроклімату робочої зони об’єкту

Період року	**Категорія робіт**	**Температура C^0**	**Відносна вологість %**	**Швидкість руху повітря, м/с**
Холодна	легка-1 а	22 - 24	40 – 60	0,1
Тепла	легка-1 а	23 - 25	40 – 60	0,1

Дане приміщення обладнане системами опалення, кондиціонування повітря або припливно-витяжною вентиляцією. У приміщенні на робочому місці забезпечуються оптимальні значення параметрів мікроклімату: температури, відносної вологості й рухливості повітря у відповідності до [4].

Рівні позитивних і негативних іонів у повітрі також мають їм відповідати.

Для забезпечення оптимальних параметрів мікроклімату в приміщенні проводяться перерви в роботі користувача, з метою його провітрювання.

Контроль параметрів мікроклімату в холодний і теплий період року здійснюється не менше 3-х разів на зміну (на початку, середині, в кінці).

4.11 Освітлення робочого місця

У проекті, що розробляється, передбачається використовувати суміщене освітлення. У світлий час доби використовуватиметься природне освітлення приміщення через віконні отвори, в решту часу використовуватиметься штучне освітлення. Штучне освітлення створюється газорозрядними лампами.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла в світильниках загального освітлення, оскільки люмінесцентні лампи мають високу потужність (80 Вт), тривалий термін служби (до 10000 годин), спектральний складом випромінюваного світла, близький до сонячного. При експлуатації ПК виконується зорова робота IV в розряду точності (середня точність). При цьому нормована освітленість на робочому місці (Ен) рівна 200 лк. Джерелом природного освітлення є сонячне світло.

У приміщенні, де розташовані ПК передбачається природне бічне освітлення, рівень якого відповідає [8].

Джерелом природного освітлення є сонячне світло. Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє [8] і для даного приміщення в світлий час доби достатньо природного освітлення.

Розрахунок освітлення.

Для будівель виробництв світловий коефіцієнт приймається в межах 1/6 - 1/10:

$$\sqrt{a^2+b^2}\cdot S_b=\left(1/8\div 1/10\right)\cdot S_n \qquad (4.1)$$

де S_b – площа віконних прорізів, м^2;

S_n – площа підлоги, м^2.

$$S_n = a \cdot b = 5 \cdot 5 = 25 \text{ м}^2$$

$$S_{вік} = 1/8 \cdot 25 = 3{,}125 \text{ м}^2$$

Приймаємо 2 вікна площею $S = 1{,}6 \text{ м}^2$ кожне.

Світильники загального освітлення розташовуються над робочими поверхнями в рівномірно-прямокутному порядку. Для організації освітлення в темний час доби передбачається обладнати приміщення, довжина якого складає 5 м, ширина 5 м, світильниками ЛПО2П, оснащеними лампами типа ЛБ (дві по 80 Вт) з світловим потоком 5400 лм кожна.

Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні. Розрахунок кількості світильників *N* виробляється по формулі (4.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M} \tag{4.2}$$

де *E* – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2; S = 25 м^2;

Z – поправочний коефіцієнт світильника (для стандартних світильників Z = 1.1 - 1.3) приймаємо рівним 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0,575

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм.

Підставивши числові значення у формулу (4.2), отримуємо:

$$n = \frac{300 \cdot 25 \cdot 1{,}1 \cdot 1{,}5}{5400 \cdot 0{,}575 \cdot 2} = 1{,}99$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160 Вт, напругою – 220 В.

Потужність електроосвітлювальної установки з урахуванням місцевого освітлення визначається за формулою:

$$N = \frac{n \cdot W + (0{,}1 \div 0{,}2) \cdot n \cdot W}{1000}, \text{ кВт} \quad (4.3)$$

де n– розрахункова кількість ламп для освітлення даного приміщення;

W – потужність однієї лампи, Вт;

(0,1÷0,2) – додаткова потужність для ламп місцевого освітлення, Вт

$$N = \frac{3 \cdot 160 + 0{,}2 \cdot 3 \cdot 160}{1000} = 0{,}576 \text{ кВт}$$

4.12 Шум, вібрація та електромагнітне випромінювання

Рівень шуму, що супроводжує роботу користувачів персональних комп’ютерів, а також зовнішніми чинниками, коливається у межах 50–65 дБА, відповідно [11]. Шум такої інтенсивності на тлі високого ступеня напруженості праці негативно впливає на функціональний стан користувачів.

Для зниження шуму на шляху його поширення передбачається розміщення в приміщенні штучних поглиначів. Для зниження рівня шуму стелю або стіни вище 1.5 - 1.7 метра від підлоги повинні облицьовуватися звукопоглинальним матеріалом з максимальним коефіцієнтом звукопоглинання в області частот 63-8000 Гц. Додатковим звукопоглинанням в КВТ можуть бути фіранки, підвішені в складку на відстані 15-20 см. Від огорожі, виконані з щільної, важкої тканини. У приміщенні з ПК коректований рівень звукової потужності не перевищує 45 дБ.

Вібрація на робочому місці в приміщенні, що розглядається, відповідає нормам відображеним в [11].

Допустимий рівень вібрацій на робочому місці: - для 1 ступеня шкідливості до 3 дБ; - для 2-3 - 1-6 дБ; - для 3 - більше 6 дБ.

4.13 Розрахунок захисного заземлення

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом [10], приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Під час роботи використовуються електроустановки з напругою живлення 36 В, 220 В, та 360 В. Опір контуру заземлення повинен мати не більше 4 Ом.

Розрахунок проводять за допомогою методу коефіцієнта використання (екранування) електродів. Коефіцієнт використання групового заземлювача η – це відношення діючої провідності цього заземлювача до найбільш можливої його провідності за нескінченно великих відстаней між його електродами. Коефіцієнт використання вертикальних заземлювачів ηв в залежності від розміщення заземлювачів та їх кількості знаходиться в межах 0,4…0,99. Взаємну екрануючу дію горизонтального заземлювача (з'єднувальної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача ηс.

Визначимо необхідний опір штучних заземлювачів Rшт.з.:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \quad (4.4)$$

де Rпр.з. – опір природних заземлювачів; Rд – допустимий опір заземлення. Якщо природні заземлювачі відсутні, то Rшт.з.=Rд.

Підставивши числові значення у формулу (4.4), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

Опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом·м. Приблизне значення питомого опору глини приймаємо ρ=40 Ом•м (табличне значення).

Розрахунковий питомий опір ґрунту, $\rho_{розр}$, Ом·м, визначається відповідно для вертикальних заземлювачів $\rho_{розр.в}$, і горизонтальних $\rho_{розр.г}$, Ом·м за формулою:

$$\rho_{розр..} = \psi \cdot \rho, \quad (4.5)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів I кліматичної зони з нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в}$=1,7 і горизонтальних $\rho_{розр.г}$=5,5 Ом·м.

$$\rho_{розр.в} = 1,7 \cdot 40 = 68 \text{ Ом•м}$$

$$\rho_{розр.г} = 5,5 \cdot 40 = 220 \text{ Ом•м}$$

Розрахуємо опір розтікання струму вертикального заземлювача Rв, Ом, за (4.6).

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \quad (4.6)$$

де lв – довжина вертикального заземлювача (для труб - 2−3 м; lв=3 м);

dст – діаметр стержня (для труб - 0,03−0,05 м; dст=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за формулою (4.7):

$$t = h_в + \frac{l_в}{2}, \quad (4.7)$$

де hв – глибина закладання вертикальних заземлювачів (0,8 м);

тоді $t = 0,8 + \frac{3}{2} = 2,3$ м

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0,05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2,3 + 3}{4 \cdot 2,3 - 3} \right) = 18,5 \text{ Ом}$$

Визначаємо теоретичну кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання ηв:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18,5}{4} = 9,25 \quad (4.8)$$

Визначаємо коефіцієнт використання вертикальних електродів групового заземлювача без врахування впливу з'єднувальної стрічки ηв =0,57 (табличне значення).

Визначаємо необхідну кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_в = \frac{2 \cdot R_в}{R_д \cdot \eta_в} = \frac{2 \cdot 18,5}{4 \cdot 0,57} = 16,2 \approx 16 \quad (4.9)$$

Визначаємо довжину з'єднувальної стрічки горизонтального заземлювача lc, м:

$$l_с = 1,05 \cdot L_в \cdot (n_в - 1), \quad (4.10)$$

де Lв – відстань між вертикальними заземлювачами, (прийняти за Lв = 3м);

$n_в$ – необхідна кількість вертикальних заземлювачів.

$$l_с = 1,05 \cdot 3 \cdot (16 - 1) \approx 48 \text{ м}$$

Визначаємо опір розтіканню струму горизонтального заземлювача (з'єднувальної стрічки) Rг, Ом:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_с} \cdot \ln \frac{2 \cdot l_с^2}{d_{см} \cdot h_г}, \quad (4.11)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_г$ – глибина закладання горизонтальних заземлювачів (0,5 м);

$l_с$ - довжина з'єднувальної стрічки горизонтального заземлювача lc, м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0,95 \cdot 0,15 \cdot 0,5} = 8,1 \text{ Ом}$$

Визначаємо коефіцієнт використання горизонтального заземлювача ηс. відповідно до необхідної кількості вертикальних заземлювачів nв.

Коефіцієнт використання з'єднувальної смуги ηс=0,3 (табличне значення).

Розраховуємо результуючий опір заземлювального електроду з урахуванням з'єднувальної смуги:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_с + R_г \cdot n_в \cdot \eta_в} \le R_д. \quad (4.12)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: Rзаг <4 Ом, а саме:

$$R_{заг} = \frac{18,5 \cdot 8,1}{18,5 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = 1,9 \le R_д$$

4.14 Висновки до розділу 4

У третьому розділі розроблені рекомендації з охорони праці, техніки безпеки при роботі на комп'ютері. Проведений аналіз умов праці, вплив шкідливих та небезпечних чинників на здоров'я людини. Визначено параметри і характеристики приміщення. Приведені рекомендації щодо організації робочого місця, електробезпеки та пожежної безпеки.

Наведені розміри приміщення та значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці.

4.15 Перелік джерел посилань до розділу 4

1. Закон України «Про охорону праці». Вводиться в дію Постановою ВР № 2695-XII від 14.10.92, ВВР, 1992, № 49, ст.669. - Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12.

2. НПАОП 0.00-6.03-93 «Порядок опрацювання та затвердження власником нормативних актів про охорону праці, що діють на підприємстві». Режим доступу: https://dnaop.com/html/43271/doc-ДНАОП_0.00-6.03-93.

3. НПАОП 0.00-4.12-05. «Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці». Режим доступу: https://zakon3.rada.gov.ua/laws/show/z0231-05.

4. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень». – Режим доступу: https://zakon.rada.gov.ua/rada/show/va042282-99.

5. НПАОП 0.00-8.24-05 «Перелік робіт з підвищеною небезпекою». Режим доступу: - Режим доступу: https://zakon.rada.gov.ua/laws/show/z0232-05.

6. ДСанПіН 3.3.2.007-98 «Правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин». - Режим доступу: https://zakon.rada.gov.ua/rada/show/v0007282-98.

7. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями». - Режим доступу: https://zakon.rada.gov.ua/laws/show/z0508-18.

8. ДБН В.2.5-28:2018 «Природнє і штучне освітлення». - Режим доступу: http://online.budstandart.com/ua/catalog/klassifikator-minregionstroya/09._dbn__(derzhavnii_23018/V.2.5-28-2018+79885-detail.html.

9. Закон України «Про забезпечення санітарного та епідемічного благополуччя населення». - Режим доступу: https://zakon.rada.gov.ua/laws/show/4004-12.

10. ДСТУ Б В.1.1-36:2016 «Визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою». - Режим доступу: http://online.budstandart.com/ua/catalog/klassifikator-minregionstroya/00._klasyfiikatsiiya_23686/v._tekhniichnii_norm_224/v.1_zahal notekhniich_234/v.1.1_zakhyst_viid_n_235/V.1.1-36-2016+70714-detail.html.

11. ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку». - Режим доступу: https://zakon2.rada.gov.ua/rada/show/va037282-99.

12. НПАОП 40.1-1.01-97. Правила безпечної експлуатації електроустановок. - Режим доступу: https://zakon.rada.gov.ua/laws/show/z0011-98.

13. НАПБ.А.01.001-2014 «Правила пожежної безпеки в Україні». - Режим доступу: https://zakon.rada.gov.ua/laws/show/z0252-15.

ВИСНОВКИ

З метою реалізації поставленої задачі у першому розділі дипломного проекту проведено аналіз інструментальних та програмних засобів налаштування мікроконтролерів.

За результатами дослідження сформульовані мета та завдання дипломного проекту. Здійснена постановка задачі на розроблення діагностичних програм для мікроконтролерів Atmel.

У другому розділі дипломного проекту проведений огляд інструментального середовища налаштування AVR Studio.

У третьому розділі дипломного проекту здійснено розроблення діагностичних програм для мікроконтролерів AVR фірми Atmel.

З метою розроблення проаналізовано виконання команд з застосуванням асемблера в інструментальному середовищі AVR Studio, визначено перелік команд логічних операцій, команд роботи з портами введення/виведення. При реалізації діагностики таймера проведено огляд внутрішніх регістрів таймера, визначені режими його роботи, проаналізована внутрішня структура регістрів таймера. Для виконання програмно-діагностичної реалізації використано мікроконтролери AT90S8515 та AT90S1200 фірми Atmel. На прикладі підключення світлодіодів до МК AT90S1200 показано застосування команд зсуву.

На кінцевому етапі реалізації проекту проведення завантаження діагностичних програм в пам'ять МК AVR з детальним аналізом регістрів пам’яті.

У четвертому розділі визначені заходи щодо охорони праці в умовах виробництва.

Результати роботи та запропоновані рішення можуть бути використані у навчальному процесі кафедри комп’ютерних наук та інженерії при вивченні дисципліни «Цифрова схемотехніка».

Додаток А

Програма діагностики взаємодії МК з кнопками і світлодіодами

```
.include "8515def.inc"
.def temp=r16                ;Тимчасовий регістр
.def reg_led=r20             ;Стан регістра світлодіодів
.equ start=0                 ;0-й розряд порту PD (PIND 0)
.equ stop=1                  ;1-й розряд порту PD (PIND 1)
.org $000
        rjmp Init            ;Перехід на ініціалізацію
;Ініціалізація
Init:
        ldi reg_led, 0xFE    ;Скидання reg_led.0 для включення
LED0
        sec                  ;Встановлення ознаки C=1 (перенесення)
                             ;в регістрі статусу SREG
        set                  ;Встановлення ознаки T=1 (напрямок)
        ser temp             ;Ініціалізація порту B на
        out DDRB, temp       ;виведення
        out PORTB, temp      ;Включити світлодіоди порту B
        clr temp             ;Ініціалізація порту PD
        out DDRD, temp       ;на ввід
        ldi temp, 0x03       ;Включення опорів
        out PORTD, temp      ;порту  PD
waitstart:
        sbic PIND, START     ;Очікування натиснення кнопки START
        rjmp waitstart
Loop:
        out PORTB, reg_led   ;Увімкнути світлодіоди
```

```
;Затримка 2 цикла
        ldi r17, 2
d1:     ldi r18, 2
d2:     dec r18
        brne d2
        dec r17
        brne d1

sbic PIND, stop          ;Якщо кнопка STOP
        rjmp mm          ;замкнута, то перехід
        rjmp waitstart   ;для перевірки кнопки START
mm:     ser temp         ;Інакше – вимкнути світлодіоди
        out PORTB, temp  ;
        brts left        ;Перехід, якщо прапорець T встановлено
        sbrs reg_led,0   ;Пропуск наступної команди, якщо
                         ;0-й розряд reg_led встановлено
        set              ;T=1 – переключення прапорця
        ror reg_led      ;зсуву reg_led вправо
        rjmp Loop        ;
left:   sbrs reg_led, 7  ;Пропуск наступної команди, якщо
                         ;7-й розряд reg_led встановлено
        clt              ;T=0 – переключення прапорця
                         ;напрямку
        rol reg_led      ;Зсув reg_led вліво
        rjmp   loop
```

Додаток Б

Програма діагностики портів

```
.include "8515def.inc"
.def port_B=r17
.def port_D=r18
.def counter=r22
.def zero=r20
.equ kod=0x1A
.equ kilkist_sprob=0x03

;Ініціалізація портів та початок програми

        ser port_D              ;Ініціалізація порту D
        out DDRD, port_D        ;на вихід
        out DDRB, port_B        ;Ініціалізація порту B на вхід
        ldi r16, kod            ;Завантаження коду
        ldi zero, 0x00          ;Регістр, що містить 0

Start:  ldi counter, kilkist_sprob; Введення кількості спроб

        ldi r21,0x00            ;Виключаємо світлодіоди
        out PORTD, r21          ;

;Очікування вводу коду з кнопок K1-K5

wait:   in r19, PINB        ;Ввід коду з порту B (PIN0-PIN3)
        cpse r19, r16       ;Код вірний?
        rjmp loop1              ;Готуємо новий цикл вводу
```

```
        rjmp loop3              ;Виходимо на закінчення (ввід
успішний)
loop1:  out PINB, r20           ;Очистити біт очікування
        dec counter             ;Зменшити кількість спроб
        cpse counter, zero      ;Кількість спроб закінчилась?
        rjmp wait               ;Повертаємось та спробуємо ще раз
        rjmp loop2

loop2:  clr counter         ;
        ldi r21, 0x02       ;Включаємо світлодіод помилки вводу
        out PORTD, r21      ;PD1 портуD
        rjmp Start          ;

loop3:  clr counter         ;Підготувати регістр для нового вводу
                            ;кількості спроб
        ldi r21, 0x01       ;Включаємо світлодіод правильного
                            ;вводу PD0 портуD
        out PORTD,  r21     ;
        rjmp start          ;Перехід на початок
```

Додаток В

Програма роботи таймера Т0 МК AVR у режимі лічильника подій

```
.include "8515def.inc"
.def temp=r16                    ;Тимчасовий регістр
.org $000
        rjmp INIT                ;Перехід на п/п ініціалізації
;Таблиця векторів переривань
.org $007
      rjmp T0_OVF                ;Перехід на п/п оброблення
переповнення
                                 ;таймера
;Ініціалізація МК
INIT:
            ldi temp,  low(RAMEND)   ;Установка покажчика
            out SPL, temp            ;стека
            ldi  temp,  high(RAMEND) ;
            ser temp                 ;Ініціалізація порту PB
            out DDRB, temp           ;на виведення інформації
; Налаштування таймера T0 на режим лічильника подій
            ldi  temp, 0x02          ;Дозвіл переривання
            out TIMSK, temp          ;при переповненні таймера в TIMSK
            ldi temp, 0x07           ;Переключення таймера
            out TCCR0, temp          ;по передньому фронту в TCCR0
            sei                      ;Дозвіл переривання в SREG
            ldi temp, 0xFC           ;$FC=–4 – відлік кількості імпульсів
            out TCNT0, temp          ;
            ser temp                 ;Ініціалізація порту PD
            out DDRD, temp           ;на виведення інформації
```

```
                out PORTD, temp         ;Включення світлодіодів

;Імітація переднього фронту імпульсу
;Кількість імпульсів – 4.
CYCL:       cbi PORTB, 0                ;Формування переднього
            sbi PORTB, 0                ;фронту імпульсу
            rjmp CYCL
;Оброблення переривання при переповненні таймера T0
T0_OVF:
            clr temp                    ;Скидання тимчасового регістра
            out PORTD, temp             ;Вимкнення світлодіодів
            rcall DELAY                 ;Затримка
            reti                        ;Повернення із п/п переривання
; Затримка виключення світлодіодів
DELAY:
            ldi r19, 5
dd:
            dec r19
            brne dd
            ret
```

Додаток Д

Програмна реалізація діагностики виконання команд зсуву

```
.device at90s1200
.include "1200def.inc"
;=======================
;Визначення регістрів

.def temp=r16        ;регістр тимчасового зберігання
.def Marc240=r17  ;регістр числового значення
.def Counter=r18   ;регістр-лічильник
.def Speed=r19      ;регістр значення швидкості
;==========================
;Початок програми
rjmp Init
;==========================
Init:
ser temp                ;Установка розрядів 0-7 регістра Port B – виходи МК
out DDRB, temp
ldi temp, 0b11111100      ;Установка розрядів 0,1 Port D – входи МК
out DDRD, temp
ldi temp, 0b00000001      ;При старті включений тільки розряд PB0
out PortB, temp
ldi temp, 0b00000011
out PortD, temp
ldi temp, 0b00000001       ;Установка частоти таймера: частота МК
out TCCR0, temp            ;Загрузка регістра керування таймером TCCR0
ldi Marc240, 240                Загрузка регістрів r17 - r19
ldi Counter, 5                  ;значеннями констант
```

```
ldi Speed, 5                ;
;============================
Start:
sbic PinD, 0        ;Перевіряємо кнопку зменшення швидкості
rjmp UpTest         ;Якщо не нажата – переходимо
inc Speed           ;Зменшуємо швидкість
cpi Speed, 11       ;Швидкість 11?
brne ReleaseDown ;    Переходимо до ReleaseDown, якщо ні, то
dec Speed           ;зменшуємо Speed на 1

ReleaseDown:
sbis PinD, 0        ;Кнопка зменшення швидкості натиснута?
                    ;Чекаємо відпускання кнопки зменшення швидкості
                    ;Перевірка біту 0 Pin D
                     ;Тут необхідно примусово встановити
                    ;біт 0 PinD в 1, інакше виникне зациклення
rjmp ReleaseDown;

UpTest:
sbic PinD,1             ;Перевіряємо кнопку збільшення швидкості
rjmp Timer              ;Не нажата - переходимо
dec Speed               ;Збільшуємо швидкість
brne ReleaseUp          ;Переходимо до Timer, якщо не 0
inc Speed               ;Збільшуємо швидкість на 1

ReleaseUp:
sbis PinD,0             ;Чекаємо відпускання кнопки
                        ;збільшення швидкості
rjmp ReleaseUp
```

```
Timer:
out TCNT0,Marc240       ;Читаємо стан T/C0 в temp
in temp,TCNT0
cp temp,Marc240         ;Порівнюємо з Marc240
brne Timer              ;Якщо не рівно, то повертаємось до Timer
subi Marc240,-240 ;Додаємо до 240 до Marc240
dec Counter             ;Зменшуємо Counter на 1
brne Start              ;Якщо не 0 повертаємось до Start

;Заданий час закінчився, змінюємо індикатор
mov Counter,Speed       ;Скидаємо Counter
in temp,PortB           ;Читаємо поточний стан PortB
lsl temp                ;Зсуваємо temp на 1 розряд вліво
brcc PC+2               ;Якщо преносу не було – пропускаємо команду
ldi temp,0b00000001     ;PB0 – включаємо, решта – виключаємо
out PortB,temp          ;Виводимо в PortB
rjmp Start              ;Повертаємось до мітки Start
```

Додаток Е

Програма запису та верифікації пам'яті EEPROM МК AT90S8515

```
        .include
"8515def.inc"
        .equ            ; Адреса програмування
AdrWr=$100
        .equ            ; Адреса читання
AdrRd=$100
        .def EEdrd=r0   ; Байт, що прочитали з пам’яті EEPROM
        .def            ; Байт, що належить записати в пам’ять EEPROM
EEdwr=r16
        .def Temp=r17   ; Допоміжний тимчасовий регістр зберігання

        RESET:
        rjmp Initial    ; Перехід до ініціалізації

        EEWrite:        ; Підпрограма "Запис в EEPROM"
        sbic            EECR,EEWE           ;Якщо EEWE не лог.0,
        rjmp            EEWrite             ;то чекати далі
        ldi             Temp,High(AdrWr)    ;Старший байт адреси
                                            ;запису в EEPROM
        out             EEARH,Temp          ; В регістр адреси
                                            ; (старша частина)
        ldi             Temp,Low(AdrWr)     ; Молодший байт адреси
                                            ; запису в EEPROM
        out             EEARL,Temp          ; В регістр адреси
                                            ; (молодша частина)
        out             EEDR,EEdwr          ; Байт даних – в регістр
```

```
                                    ; даних
sbi         EECR,EEMWE              ; Розряд EEMWE
                                    ; дозволяє програмування
sbi         EECR,EEWE               ; Розряд EEME: встановлено
                                    ; початок програмування.
                                    ; Команда виконується
                                    ; протягом 4–х тактів,
                                    ; оскільки затримка ЦП
                                    ; складає 2 такти
ret

EERead:     ; Підпрограма "Читання EEPROM"
sbic        EECR,EEWE
rjmp        EERead
ldi         Temp,High(AdrRd)
out         EEARH,Temp
ldi         Temp,Low(AdrRd)
out         EEARL,Temp
sbi         EECR,EERE
in          EEdrd,EEDR
ret

Initial:    ; Підпрограмам ініціалізації
ldi         Temp,Low(RAMEND)
out         SPL,Temp
ldi         Temp,High(RAMEND)
out         SPH,Temp                ; Встановити початок стеку
ldi         Temp,$ff                ; Напрям передачі – вивід
out         DDRA,Temp               ; В регістр передачі даних
ldi         Temp,$00                ; Напрям передачі – ввід
```

```
out     DDRC,Temp     ; В регістр передачі даних
out     PORTC,Temp    ; Обрати порт С
in      EEdwr,PinC    ; Загрузити байт з порту С
rcall   EEWrite       ; Записуємо байт за адресою
                      $100
rcall   EERead        ; Читаємо байт даних за
                      ; адресою $101
out     PortA,EEdrd

Endlos:
rjmp Endlos
```

Додаток Ж

Презентація

Міністерство освіти і науки України
Східноукраїнський національний університет ім. Володимира Даля

«Програмна реалізація тестування мікроконтролерів AVR»

Студент гр. КІ – 16 бд Бережний П. А.
Керівник Кардашук В. С.

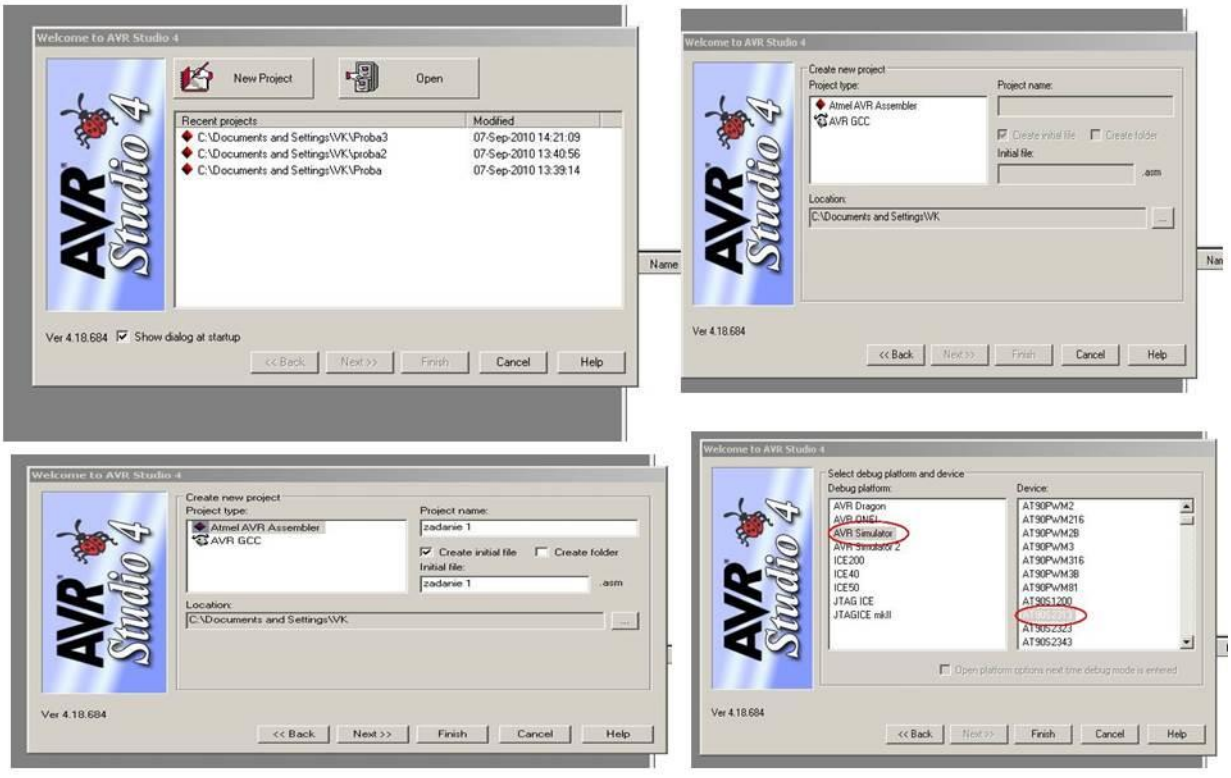
Актуальність теми

- Впровадження МК в усі сфери життєдіяльності ставить перед розробниками електронної техніки завдання із забезпечення діагностування МК.
- Номенклатура та область застосування таких систем постійно розширюється. На сучасному етапі науково-технічного розвитку їх впровадження охоплює практично всі види виробничої та наукової діяльності.
- МК в повсякденному житті застосовуються як в складній побутовій техніці, так і у супутникових навігаційних системах. До сфери застосування МК входить управління пристроями різного призначення за допомогою дискретних сигналів і багато іншого.
- Мета дипломного проекту – розроблення діагностичних програм для МК AVR фірми ATMEL з застосуванням програмного середовища налаштування AVR Studio 4.

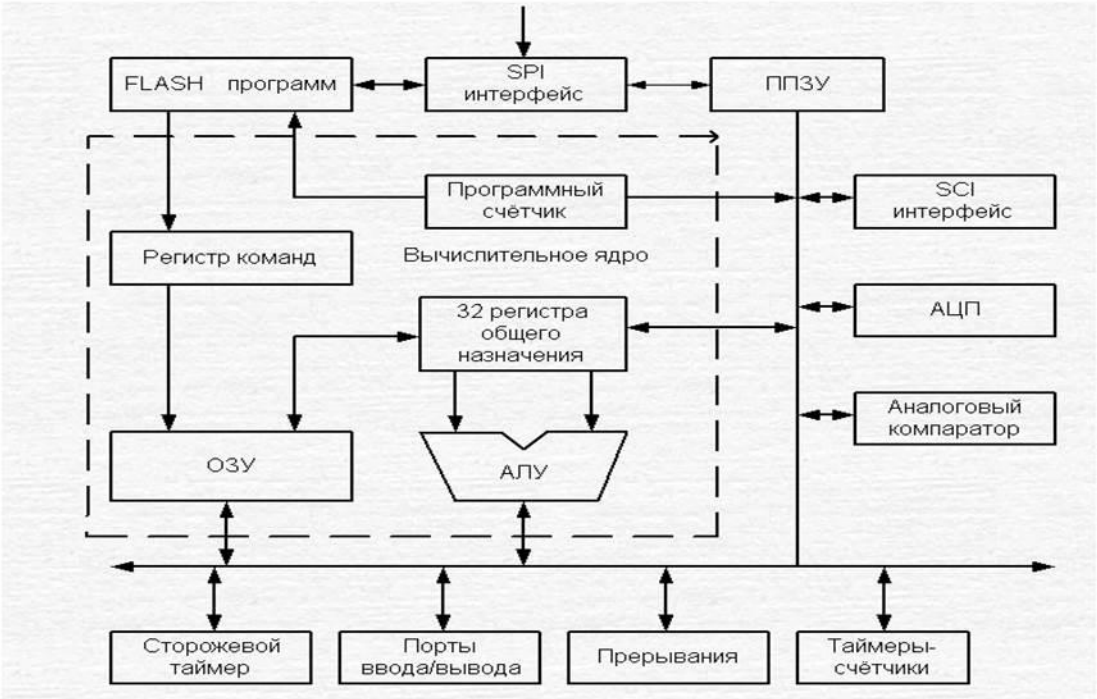
Для виконання поставленої задачі в дипломній роботі необхідно:

- провести аналіз інтерфейсу програми AVR Studio 4;
- розглянути можливості програми AVR Studio 4;
- розробити діагностичну програму звернення до портів вводу-виводу МК AVR для керування зовнішніми пристроями;
- розробити діагностичні програми для перевірки працездатності мікроконтролера;
- здійснити компіляцію, налаштування та завантаження діагностичних програм в пам'ять МК AVR.

Діалогові вікна AVR Studio 4



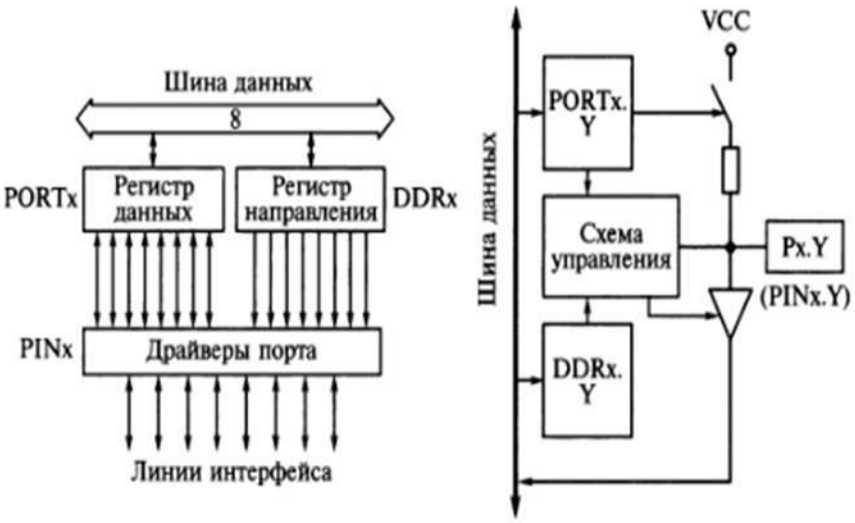
Загальна структурна схема МК AVR



Характеристики мікроконтролерів ATtiny фірми Atmel

Тип	Напр. питания В	Такт. Частота МГц	I/O	Flash	EEPROM	SRAM	Интер-фейсы	АЦП	Таймеры	ISP	Корпус
ATtiny11	2.7-5.5	6	6	1K	-	-	-	-	1x8bit	-	PDIP8 SOIC8
ATtiny12	1.8-5.5	6	6	1K	64	-	-	-	1x8bit	I	PDIP8 SOIC8
ATtiny13	1.8-5.5	20	6	1K	64	64	-	4x10bit	1x8bit 2xPWM	I	PDIP8 SOIC8
ATtiny15	2.7-5.5	6	6	1K	64	-	-	4x10bit	2x8bit	I	PDIP8 SOIC8
ATtiny2313	1.8-5.5	20	15	2K	128	128	SPI UART	-	1x8bit 1x16bit	I	PDIP20 SOIC20 MLF32
ATtiny24	1,8...5,5	20	12	2K	128	128	USI 4xPWM RTC	8x10bit	1x8bit 1x16bit	S	PDIP14 MLF20 SOIC14
ATtiny25	2,7...5,5	20	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed	I	PDIP8 SOIC8
ATtiny25V	1.8 - 5.5	10	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed	I	PDIP8 SOIC8
ATtiny26	2.7-5.5	16	16	1K	128	128	SPI UART	11x10bit	2x8bit	I	PDIP20 SOIC20 MLF32
ATtiny28	1.8-5.5	4	20	2K	-	-	-	-	1x8bit	-	PDIP20 SOIC20 MLF32

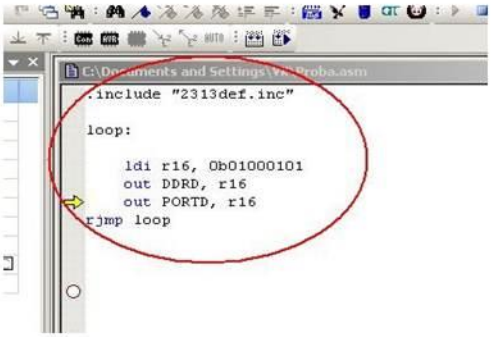
Структура портів вводу-виводу



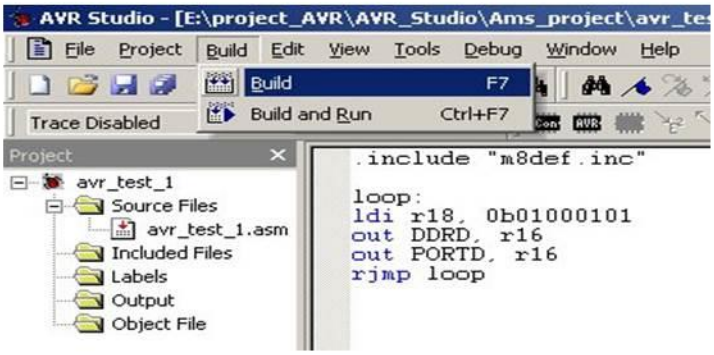
Команди логічних операцій

Мнемоніка команди	Операція
and	логічне множення
andi	логічне множення з безпосереднім значенням
or	логічне додавання
ori	логічне додавання з безпосереднім значенням
eor	мажоритарна операції (додавання по модулю 2)

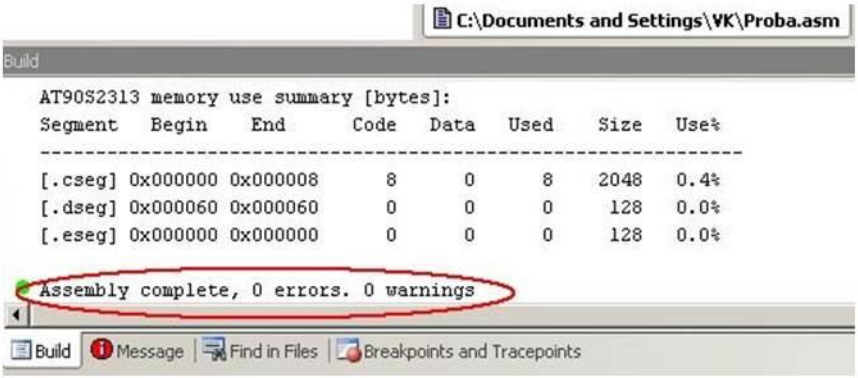
Робоче вікно проекту



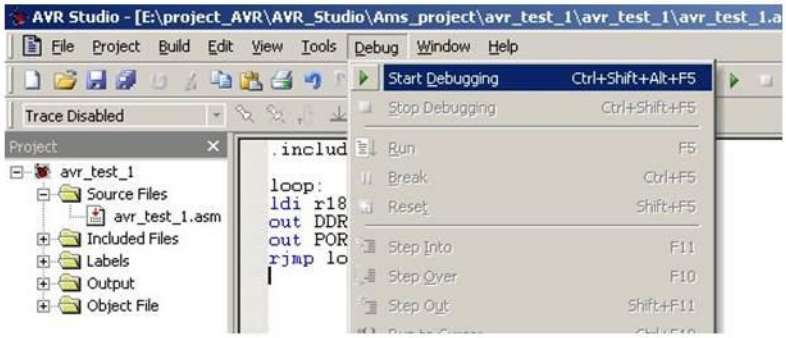
Початок компіляції програми



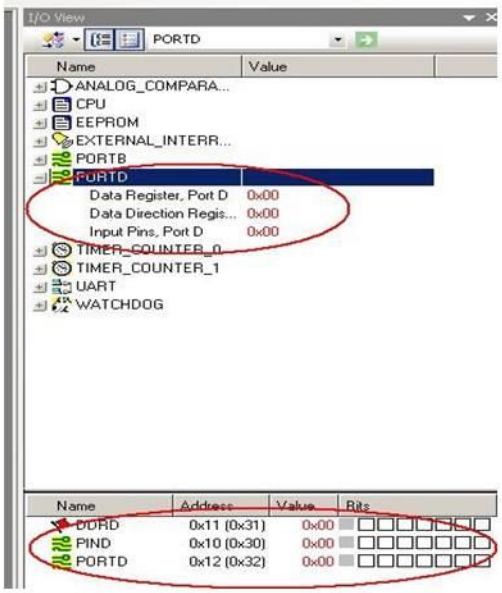
Звіт про роботу компілятора



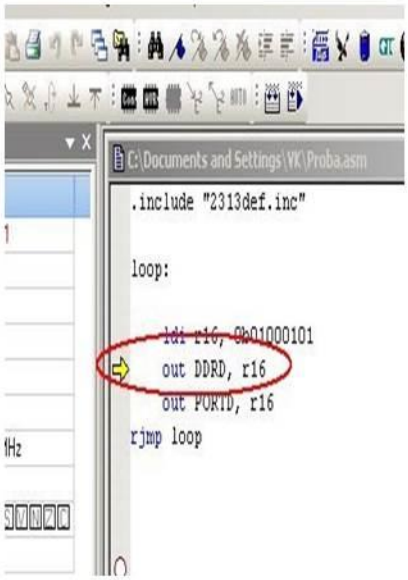
Запуск програми налаштування МК



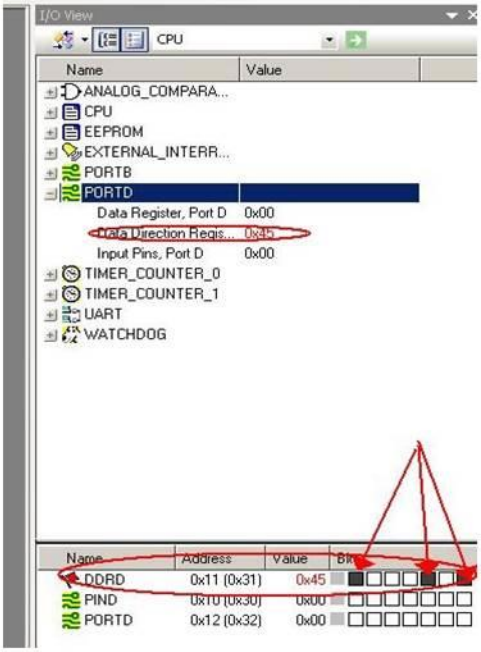
Вікно панелі I/O View та стан порту D

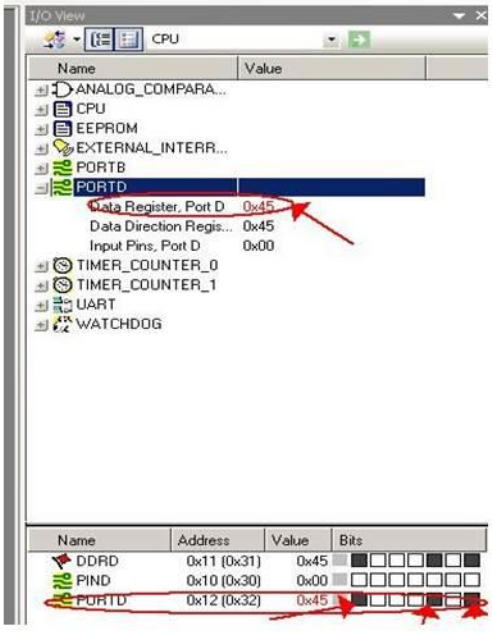


Переміщення покажчика виконання програми

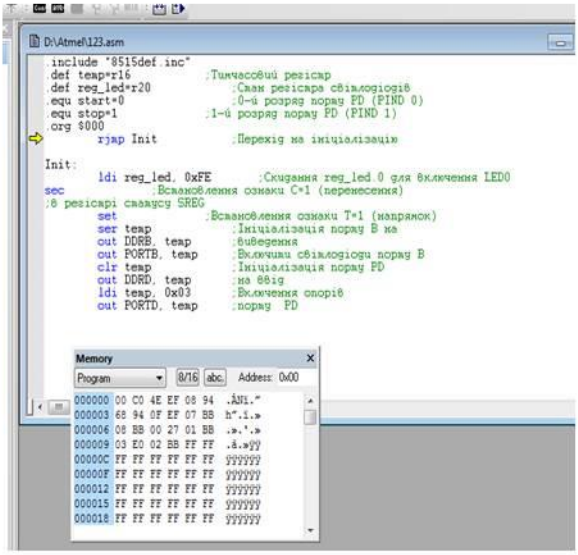


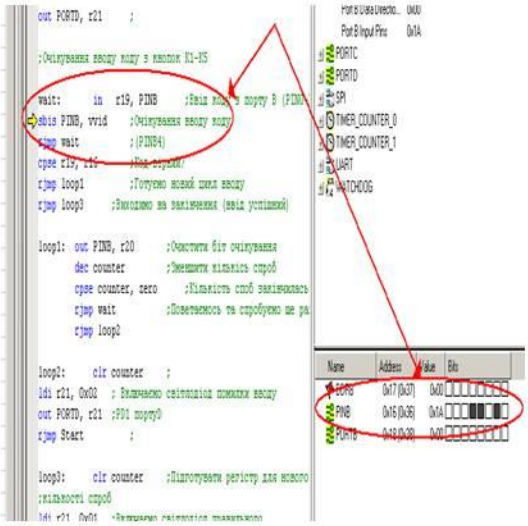
Результат виведення результату в порт



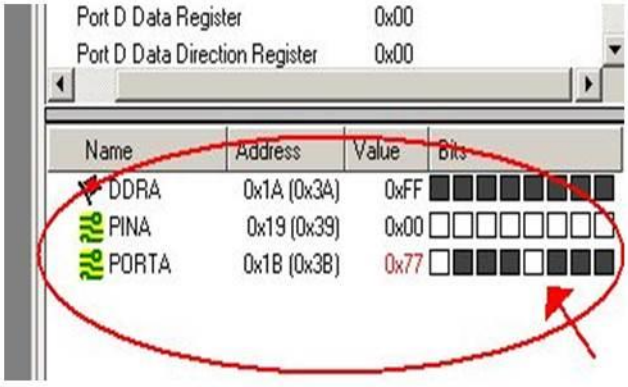


Діагностика портів

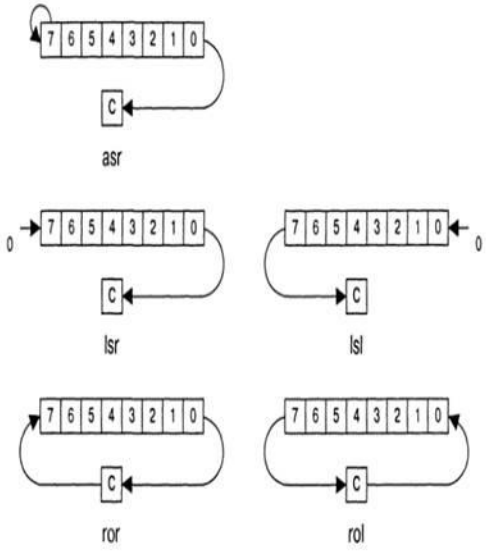


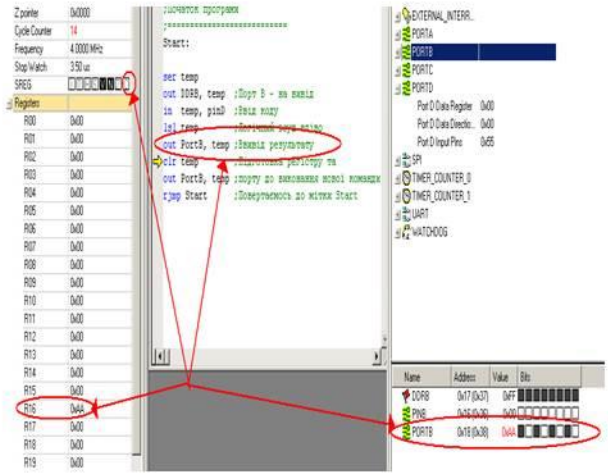


Завершення програми – видача коду в порт A

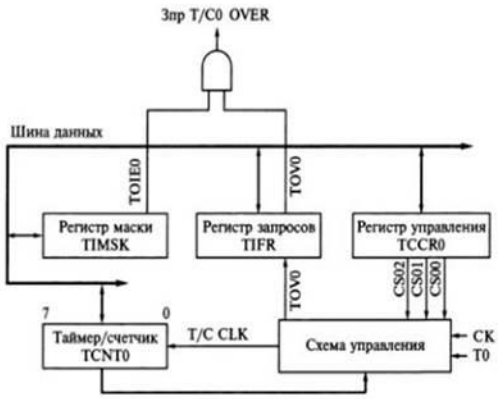


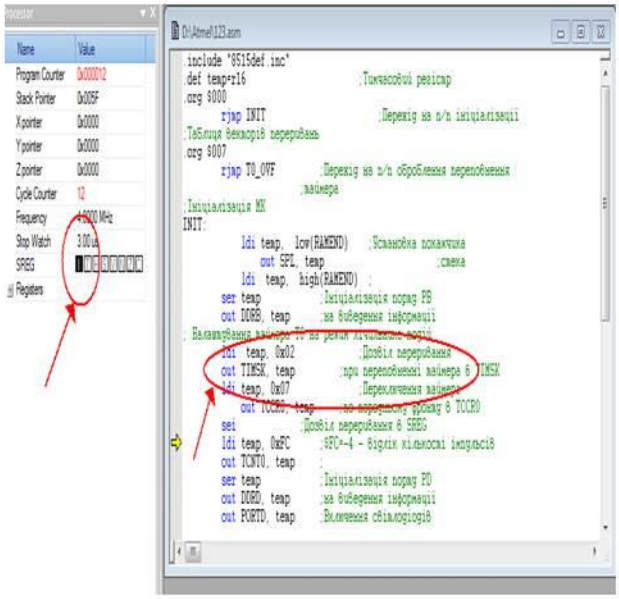
Структура та діагностика команд зсуву



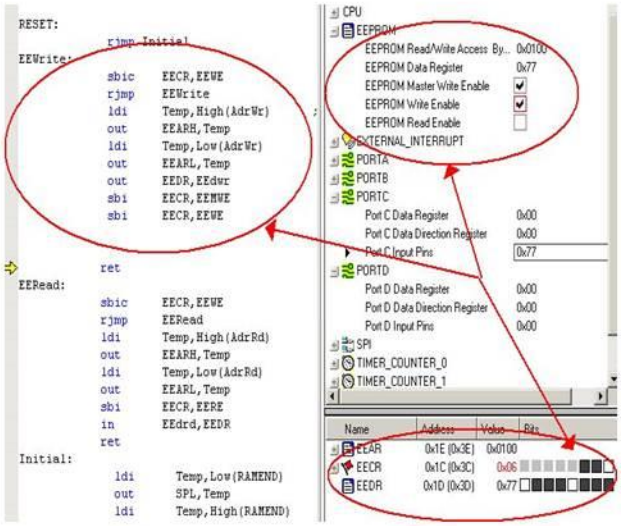


Структура та діагностика таймеру

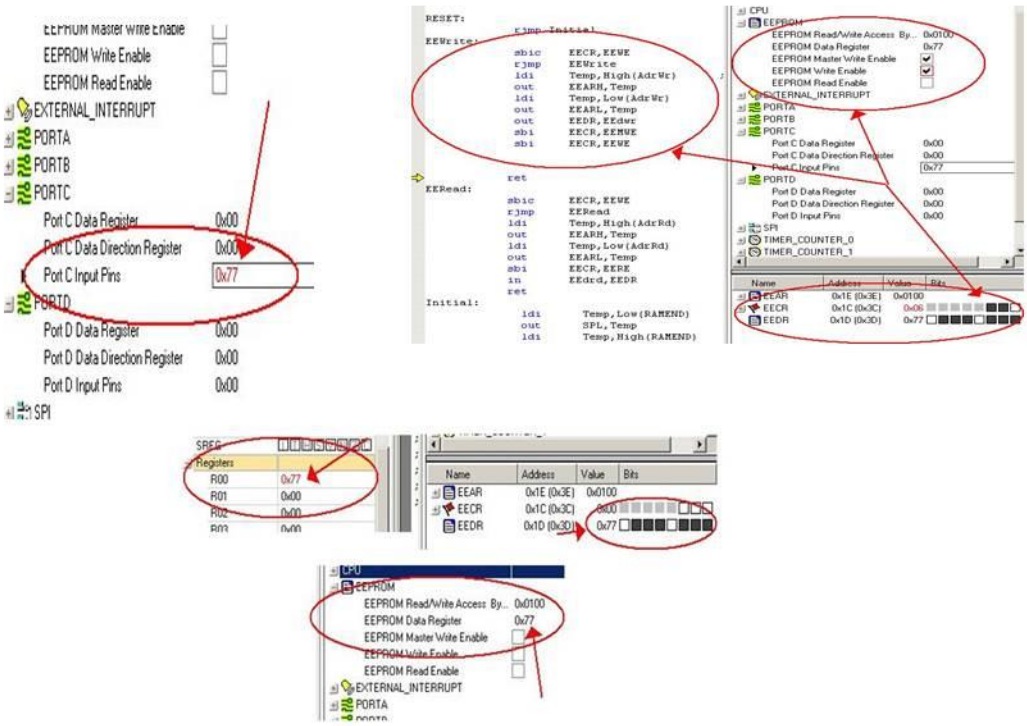




ЗАВАНТАЖЕННЯ ДІАГНОСТИЧНИХ ПРОГРАМ В ПАМ'ЯТЬ МК AVR

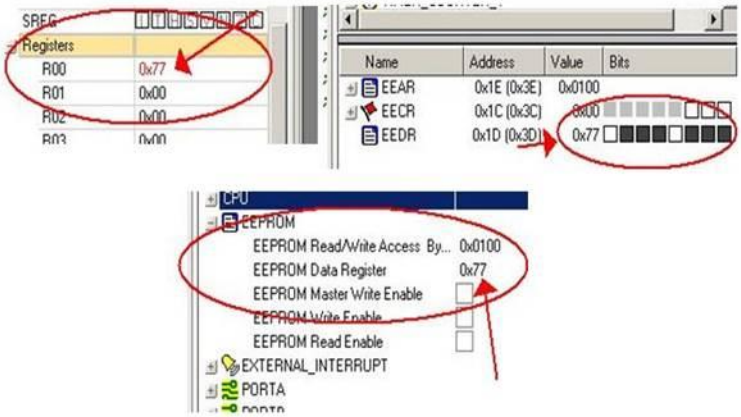


Завантаження діагностичних програм в пам'ять EEPROM МК AVR.



Верифікація пам'яті EEPROM

Перевірка процедури запису на прикладі комірки $100 пам'яті EEPROM



Висновки

У дипломному проекті:

- Проведено аналіз інтерфейсу програми AVR Studio 4;
- Розглянуті можливості програми AVR Studio 4;
- Розроблено діагностичну програму звернення до портів вводу-виводу МК AVR для керування зовнішніми пристроями;
- Розроблено діагностичні програми тестування з метою перевірки працездатності мікроконтролера;
- Здійснено компіляцію, налаштування та завантаження діагностичних програм в пам'ять МК AVR.
- Розроблені заходи щодо охорони праці та безпеки в надзвичайних ситуаціях
- Результати роботи та запропоновані рішення можуть бути використані у навчальному процесі кафедри комп'ютерних наук та інженерії при вивченні дисципліни «Цифрова схемотехніка».