МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ БОГДАНА ХМЕЛЬНИЦЬКОГО

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

КАФЕДРА АВТОМАТИЗАЦІЇ ТА КОМП'ЮТЕРНО-ІНТЕГРОВАНИХ ТЕХНОЛОГІЙ

МЕТОДИЧНІ ВКАЗІВКИ

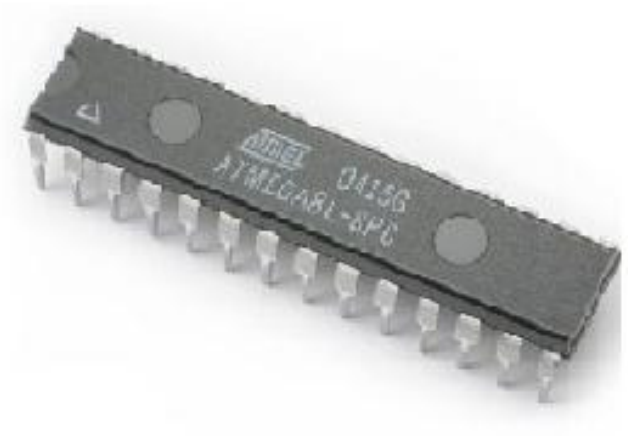
до лабораторних робіт з дисципліни

"МІКРОПРОЦЕСОРНА ТЕХНІКА"

для студентів напрямку підготовки 6.050202

" Автоматизація та комп'ютерно-інтегровані технології "

всіх форм навчання



Черкаси – 2013

ББК 32.973-04
УДК 004.312

Методичні вказівки до виконання лабораторних робіт з курсу «Мікропроцесорна техніка» для студентів напрямку підготовки 6.050202 “Автоматизація та комп’ютерно-інтегровані технології ”
Укл. В.А.Дідук – Черкаси, ЧНУ ім. Б.Хмельницького, 2013 –56с.

Навчальне видання

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

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

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

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

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

ЗМІСТ

ВСТУП

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

В ході лабораторних робіт досліджуються принципи мікропрограмного керування дискретними пристроями, сесмсегментними свтлодіодними індикаторами, рідкокристалічними індикаторами та аналого-цифровими перетворювачами.

Виконання лабораторних робіт базується на знаннях, отриманих студентами з основ теорії електроніки та мікросхемотехніки, цифрової схемотехніки, інформаційних систем, програмного забезпечення інформаційних систем, інформатики.

Загальний порядок виконання лабораторних робіт.

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

РОЗДІЛ 1

ОСНОВИ РОБОТИ З НАВЧАЛЬНО-ВІДЛАГОДЖУВАЛЬНОЮ ПЛАТОЮ AVR-EASY

ЗАГАЛЬНИЙ ОГЛЯД ТА ПРИЗНАЧЕННЯ НАВЧАЛЬНО-ВІДЛАГОДЖУВАЛЬНОЇ ПЛАТИ AVR-EASY

Навчально-відлагоджувальна плата (НВП) – AVR-Easy призначений для макетування і налагодження різних пристроїв на базі AVR-контролерів. Він може бути використаний як для навчання основам мікропроцесорної техніки так і для реалізації широкого спектру задач цифрової обробки даних і управління різного роду периферійними пристроями.

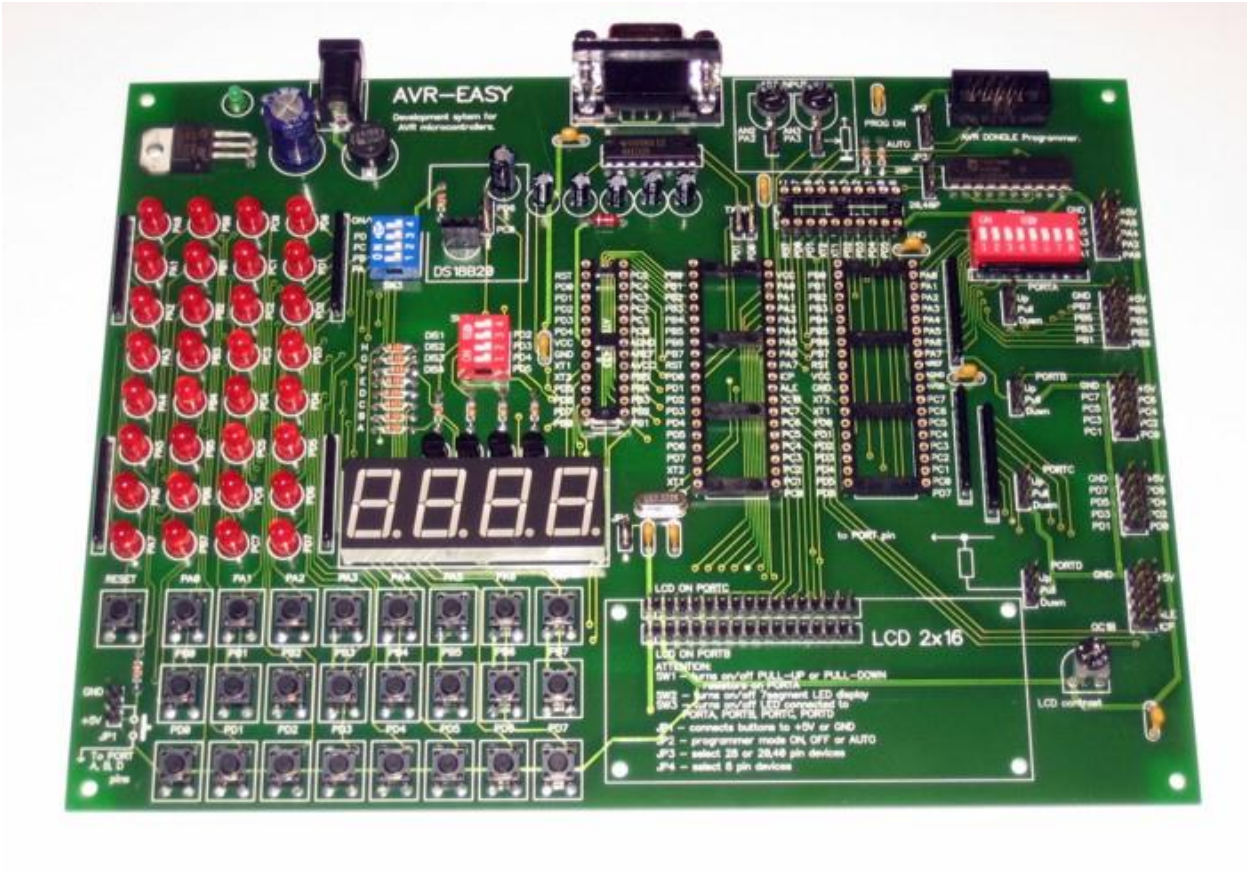


Рис. 1.1. Зовнішній вигляд навчально-відлагоджувальної плати AVR-Easy

В основу НВП покладено принцип максимальної універсальності і зручності програмування і налагодження програм на базі AVR-контролерів. НВП є ідеальним засобом навчання, що дозволяє в короткий термін навчитися вирішувати типові задачі спряження AVR-контролерів з найбільш часто використовуваними на практиці периферійними модулями. Зокрема, НВП дозволяє вивчити спряження AVR-контролерів з рідко-кристалічним алфавітно-цифровим індикатором (LCD) і семисегментним світлодіодним індикатором, організувати зв'язок з послідовним портом персонального комп'ютера (COM-порт), організувати роботу з інтелектуальними датчиками температури (на прикладі мікросхеми DS18b20 фірми DALLAS).

НВП має контакти або роз'єми для перерахованих вище елементів і не вимагає напайки зовнішніх навісних деталей. Разом з тим, кожен порт AVR-контролера має вихід на окремий роз'єм, куди можна підключити будь-який зовнішній пристрій. Передбачена світлодіодна індикація сигналів на входах-

виходах портів A, B, C, D AVR-контролера. На входах-виходах портів A, B, D підключені кнопки, при натисканні на які можна подавати логічний рівень "0" або "1" на відповідні порти. На входи АЦП AVR-контролера можна подавати аналогову напругу за допомогою змінних резисторів в діапазоні 0 – 5 В. Таким чином, НВП можна використовувати не тільки як засіб навчання, але і як пристрій для макетування і налагодження реальних практичних розробок на базі AVR-контролерів. НВП має кілька панельок для установки AVR-контролерів в корпусах DIP-20, DIP-28 і DIP-40, куди можна встановлювати практично будь-які контролери, включаючи самі останні моделі широко використовуваних контролерів сімейств ATMEGA8515 і ATMEGA8535.

До НВП може бути підключений будь-який контролер, що співпадає за виводами живлення та програмування з найбільш поширеними контролерами. НВП має роз'єм для програмування і налагодження, який дозволяє замінити програму AVR-контролера, не виймаючи його з посадкової панелі НВП.

Плата дозволяє вивчити роботу мікроконтролерів сімейства AVR фірми ATMEL, а також налагоджувати додаткові периферійних модулі. Підтримуються мікроконтролери у 8, 20, 28 і 40 вивідних корпусах.

ОПИС КОМПОНЕНТІВ НАВЧАЛЬНО-ВІДЛАГОДЖУВАЛЬНОЇ ПЛАТИ

Для живлення плата містить стабілізоване джерело на 5В, зібране по класичній схемі. Вхідна напруга – постійна або змінна 9 – 12 вольт.

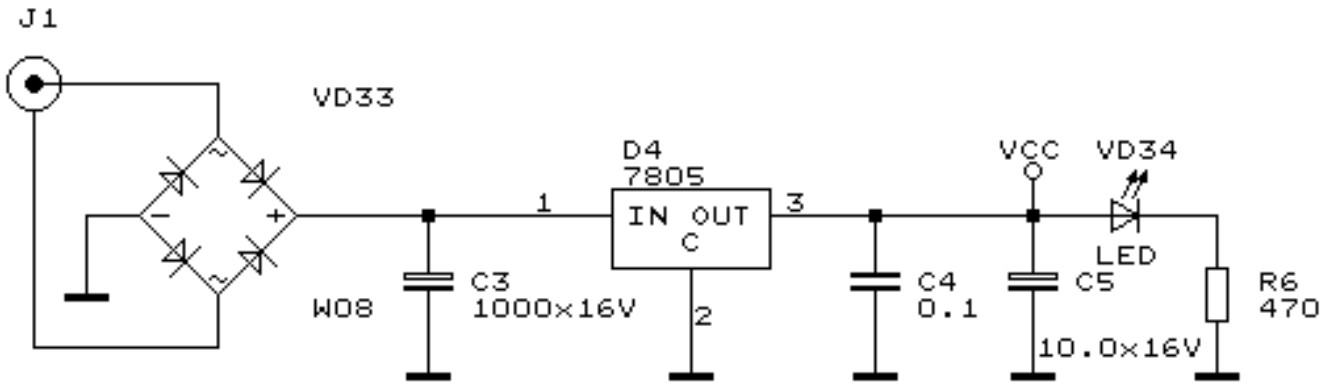


Рис. 1.2. Схема електрична принципова стабілізатора джерела живлення

Плата містить 32 світлодіода, підключених до портів A, B, C, D мікроконтролера. Для того щоб засвітити світлодіод необхідно ввімкнути потрібний порт перемикачем SW3 і записати '0 'в потрібний розряд. Тобто, низький віхідний рівень на портах мікроконтролера вмикає світлодіод, високий – вимикає.

Система містить кнопку початкової установки контролера RESET і 24 кнопки, що дозволяють симулювати вхідні дії на порти A, B, D. Загальний контакт кнопок може бути підключений до ланцюга живлення або "землі" за допомогою перемички JP1.

Плата дозволяє підключити стандартний рідкокристалічний індикатор двома способами, в першому випадку індикатор підключається на PORTB, у другому на PORTC.

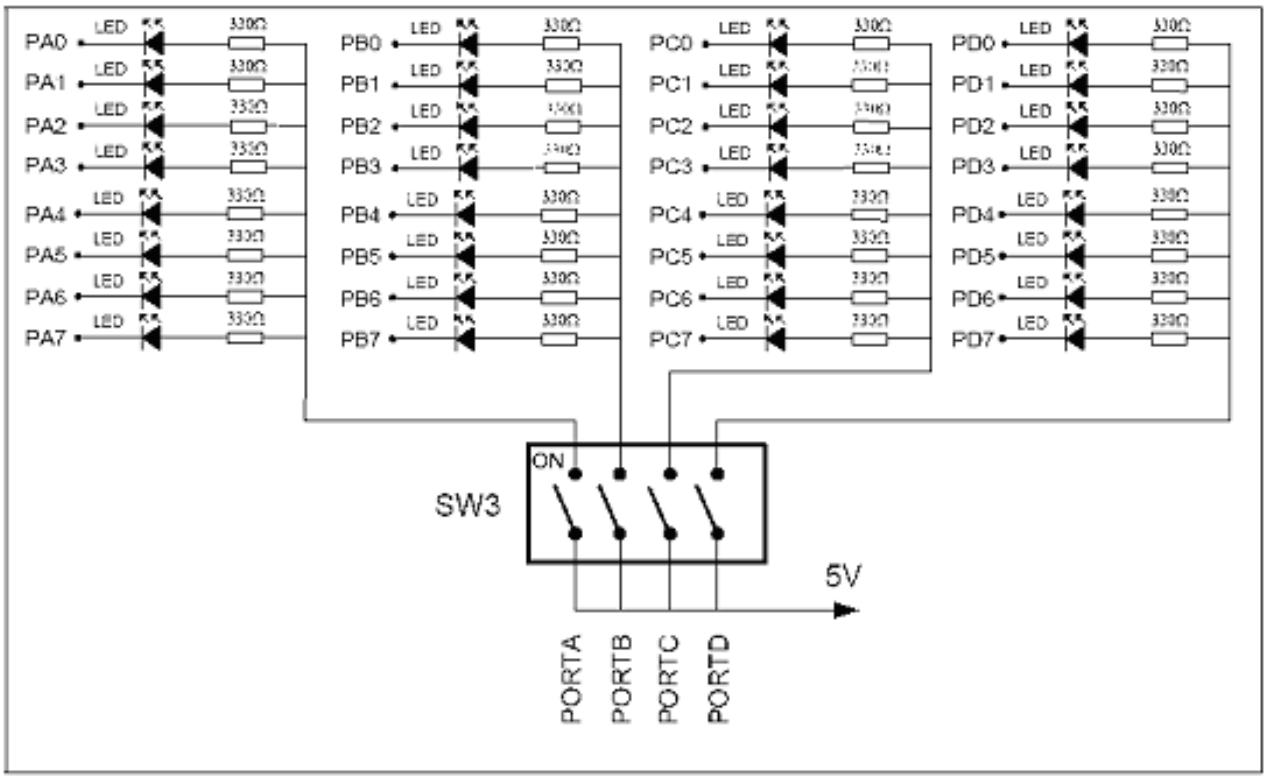


Рис. 1.3. Схема ввімкнення світло діодів

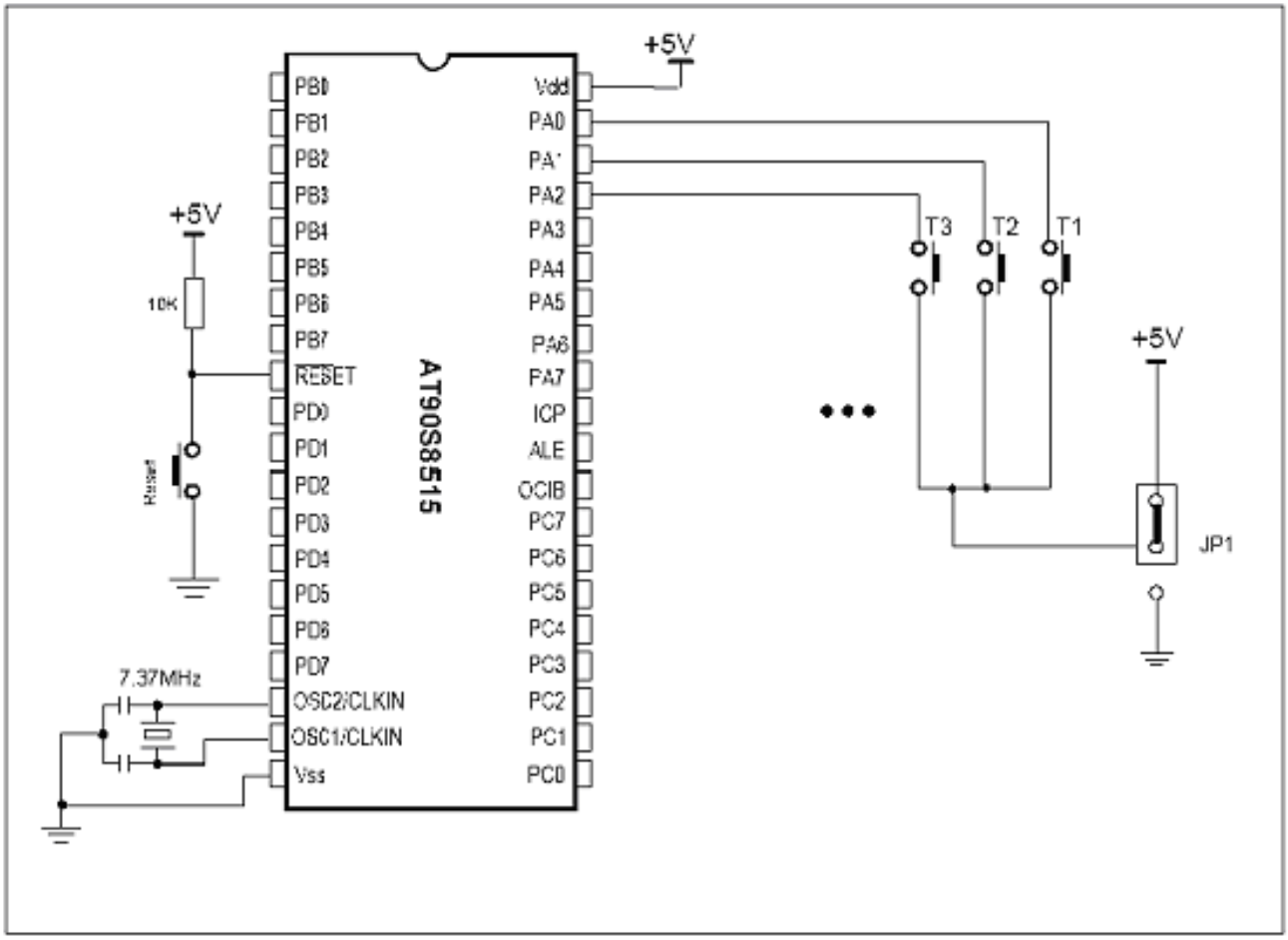


Рис. 1.4. Схема ввімкнення кнопок

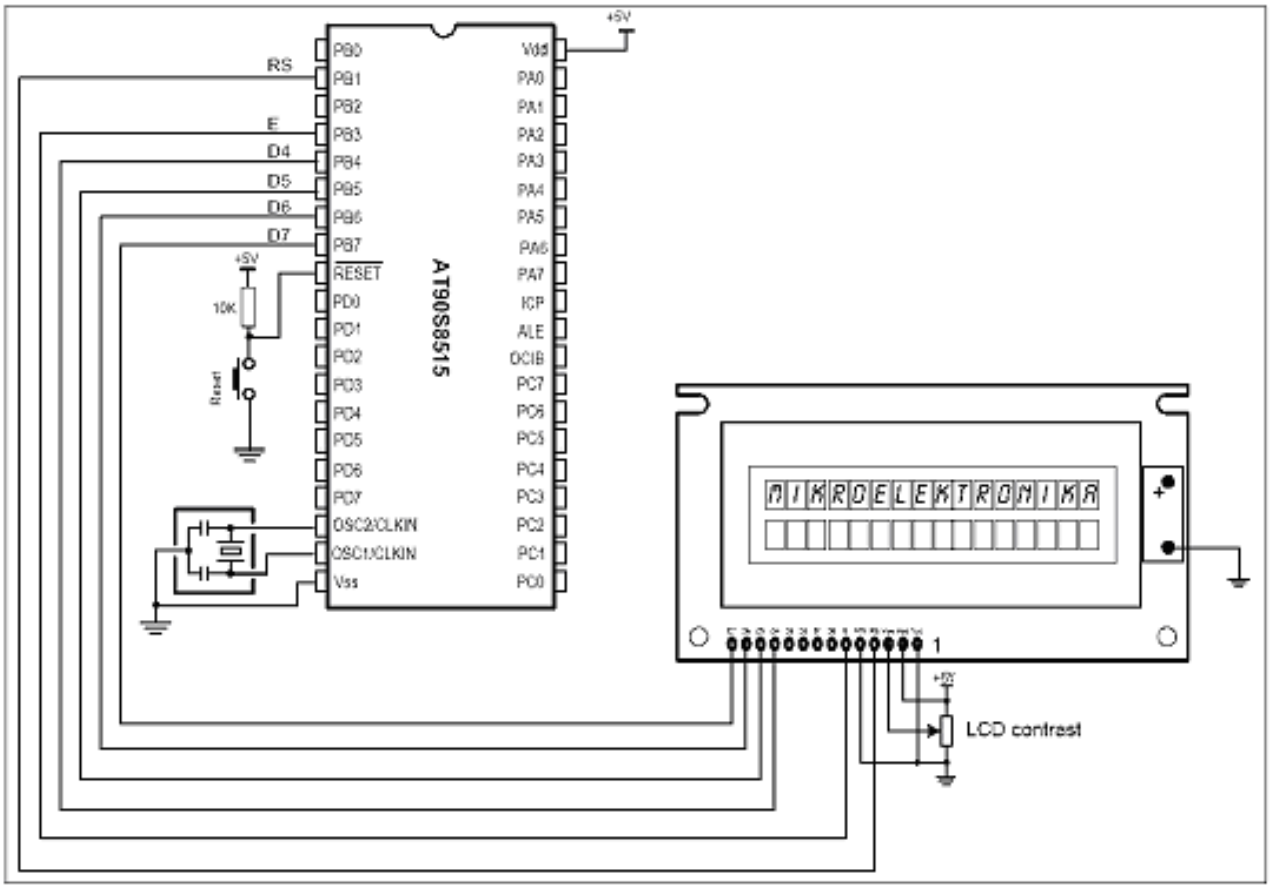


Рис. 1.5. Схема ввімкнення символьного рідкокристалічного індикатора до порту PORTB

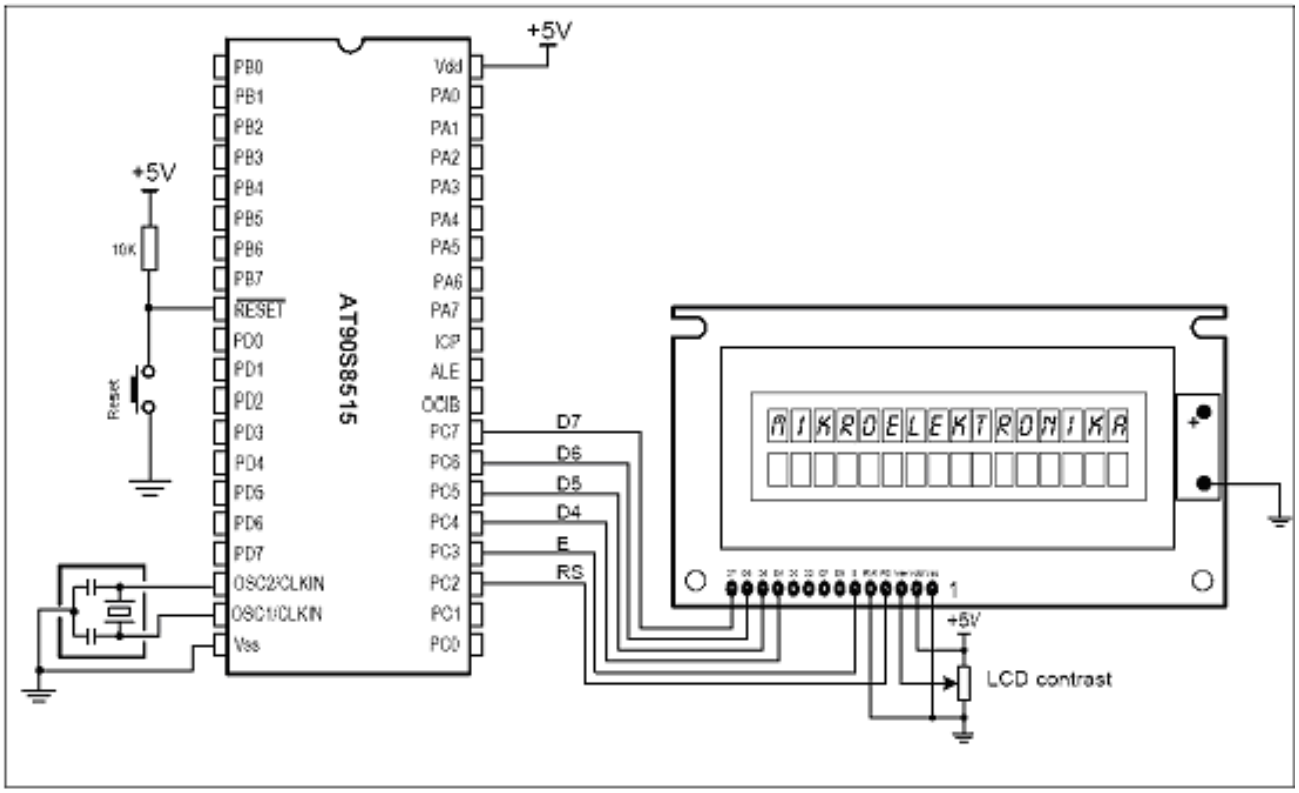


Рис. 1.5. Схема ввімкнення символьного рідкокристалічного індикатора до порту PORTB

Дана версія плати дозволяє працювати з мікроконтролерами в 8-вивідному корпусі,для цього необхідно встановити контролер в 20-ти вивідну панельку, перша ніжка контролера в перший контакт панельки і встановити перемичку JP4.

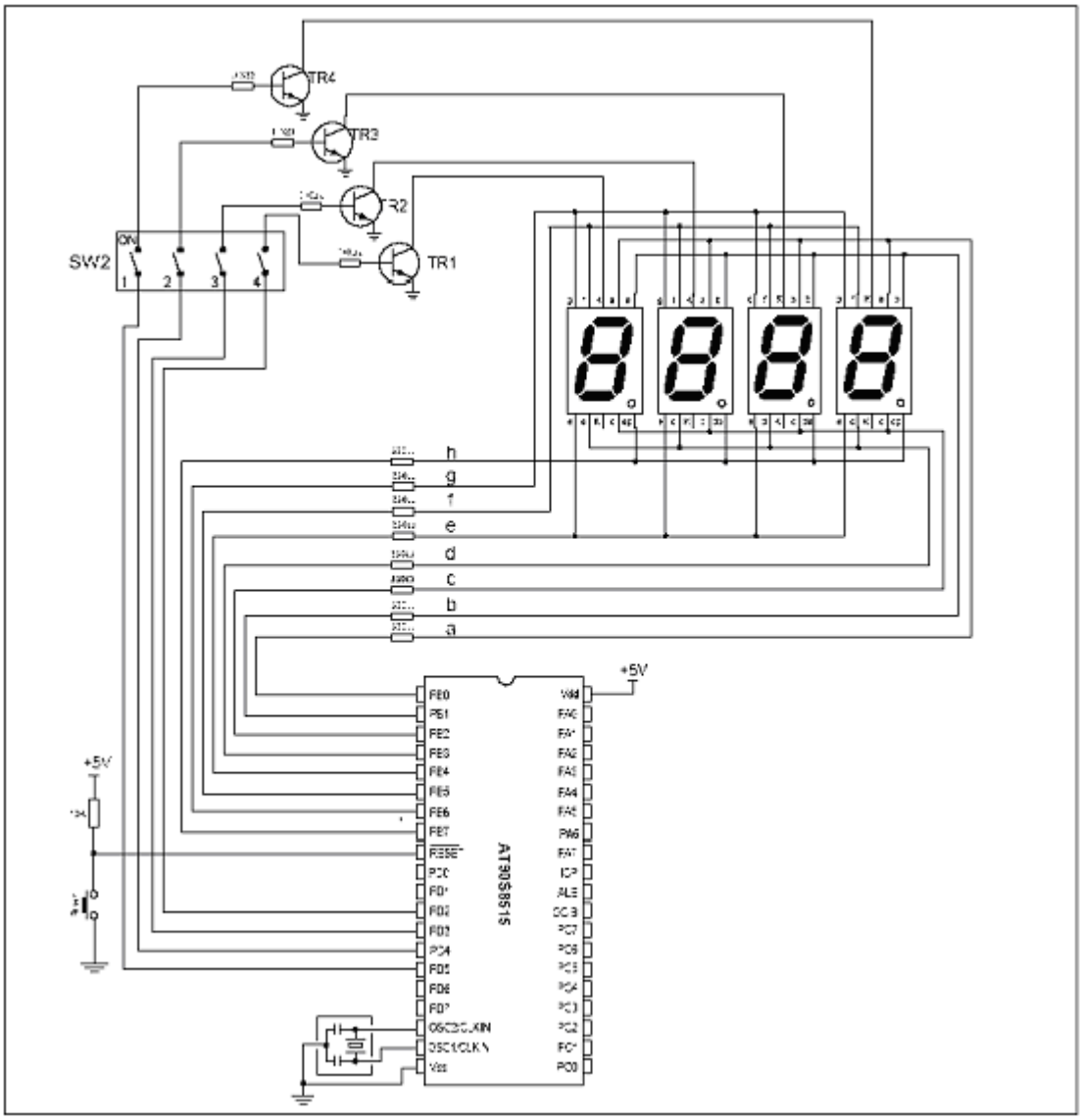


Рис. 1.6. Схема включення 7-ми сегментного індикатора

Плата містить чотирирозрядний 7-ми сегментний індикатор, який підключений до порту B. Знакомісця керуються з порту D розрядами 2, 3, 4 і 5 через перемикач SW2.

У використовуваному індикаторі застосований динамічний спосіб індикації, він полягає в тому, що в кожен момент часу відображається тільки один розряд індикатора.

На PORTB виводиться відображуваний символ, а на PORTD позиційний номер знакомісць (активний рівень "високий").

Частота регенерації повинна бути не менше 50 герц, інакше буде помітно мерехтіння індикатора. Мікроперемикачем SW2 можна відключити знакомісце,

яке не використовується.

У деяких мікроконтролерах міститься 10-розрядний АЦП. Для таких контролерів плата дозволяє задавати тестові впливу на піни PORTA.2 і PORTA.3. Також є можливість підключати до окремих висновків "аналогового" порту PULLUP і PULLDOWN резистори опором 10КОм за допомогою мікроперемикача SW1.

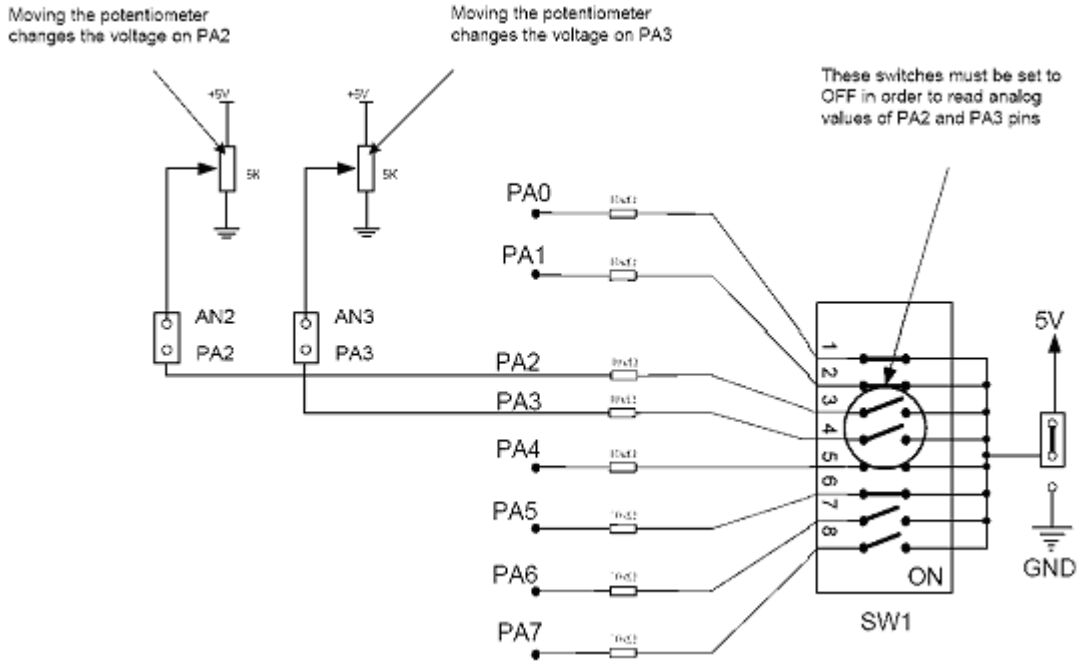


Рис. 1.7. Схема підключення змінних резисторів до ліній АЦП

На платі передбачений інтерфейс RS-232 з можливістю його відключення

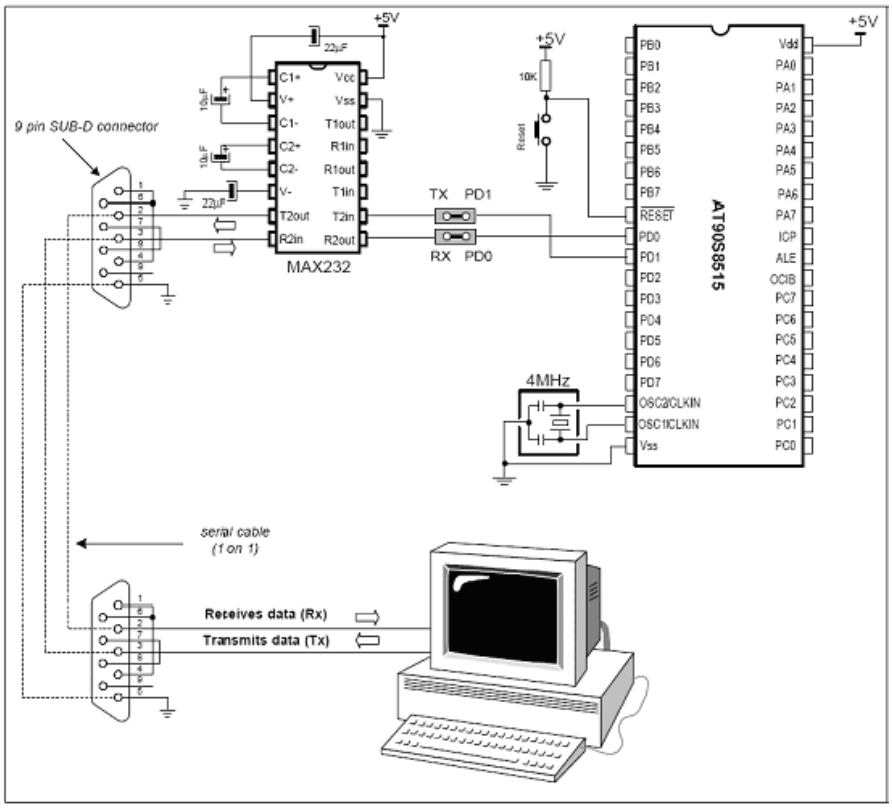


Рис. 1.8. Схема підключення НВП до комп’ютера

На платі передбачено підключення цифрового термометра DS18B20, який дозволяє вимірювати температуру від -55 ° С до 125 ° С. Датчик встановлюється в триконтактну панель і може бути підключений до ліній PC0 або PD6.

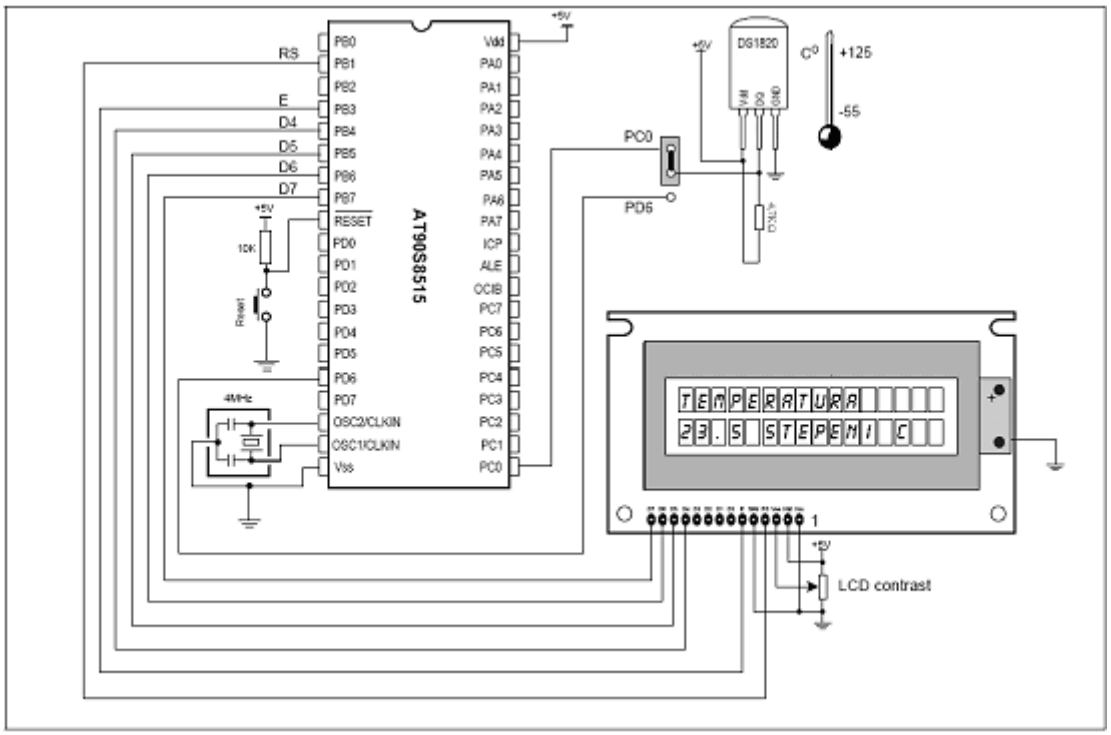


Рис. 1.9. Схема підключення цифрового термометра DS18B20

Встановивши перемички в певну позицію, ми можемо "підтягти" або "опустити" виводи портів. На портах B, C, D всі виводи приєднуються одночасно групою. Лінії порту A можна "підтягувати" або "опускати" вибірково.

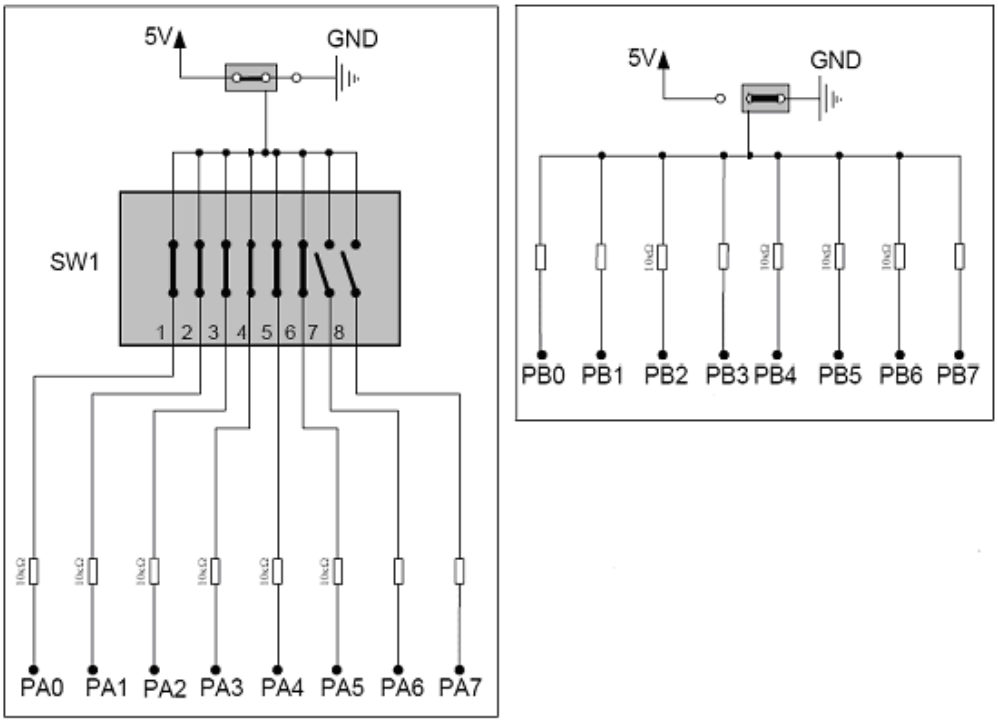


Рис. 1.10. Схеми включення PULL-UP PULLDOWN резисторів на НВП

Для програмування контролера можна скористатися будь-яким ATMEL-10 сумісним програматором. Плата розроблена таким чином, що після програмування програматор відключається, і лінії програмування можуть бути задіяні у вашій схемі.

Спочатку необхідно встановити перемичку JP3 в положення 28 або 20,40 в залежності від використовуваного контролера. Якщо програматор підтримує режим авто відключення – при переведенні перемички JP2 в положення AUTO, після програмування програматор автоматично відключиться. Якщо програматор не підтримує цей режим, перемичку JP2 потрібно встановити в положення PROG ON – програматор завжди підключений. Якщо після програмування контролера зняти JP2 – програматор відключиться, і можна використовувати виводи програмування в своєму проекті.

При використанні РКІ перемичку JP2 потрібно знімати обов’язково

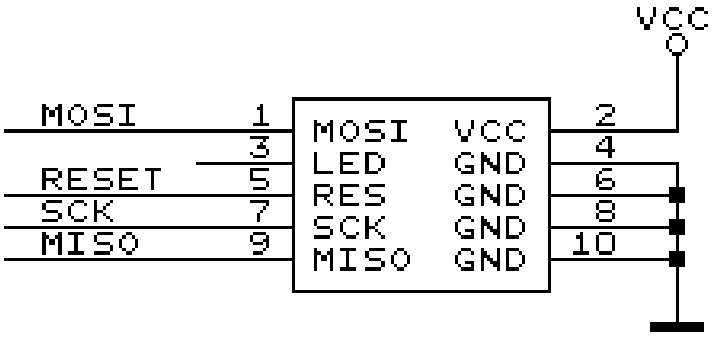


Рис. 1.11. Схема підключення програматора

РОЗДІЛ 2

ЛАБОРАТОРНІ РОБОТИ

Лабораторна робота № 1
ПЕРША ПРОГРАМА. УПРАВЛІННЯ СВІТЛОДІОДАМИ

Мета роботи: ознайомитись з принципом управління дискретними пристроями, що приєднані безпосередньо до портів мікроконтролера, а також програмним забезпеченням WinAVR, Codevision AVR, Flowcode for AVR. Набути навиків створення та компіляції власної програми для мікроконтролера.

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATtiny2313, ATmega8, ATmega16, Atmega8515; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал

Порти введення / виведення МК AVR мають число незалежних ліній "Вхід / Вихід" від 3 до 53. Кожен розряд порту може бути запрограмований на введення або на виведення інформації. Вихідні драйвери забезпечують струмову навантажувальну здатність 20 мА на лінію порту при максимальному значенні 40 мА, що дозволяє, наприклад, безпосередньо підключати до мікроконтролера світлодіоди і біполярні транзистори. Загальна струмова навантаження на всі лінії одного порту не повинна перевищувати 80 мА (всі значення наведено для напруги живлення 5 В).

Для роботи з будь-яким портом «x» існують три регістра управління:DDRx, PORTx, PINx.

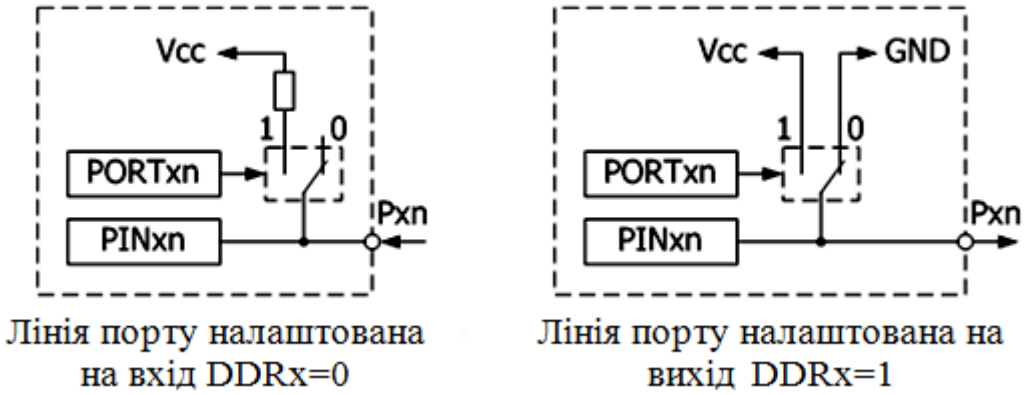


Рис. 2.1. Схеми налаштування портів на режими введення / виведення

DDRx (The Port X Data Direction Register) – визначає напрям передачі даних кожної лінії порту X: 1 - вихід, 0 - вхід. Регістр DDRx містить 8 біт DDx0 .. DDx7, кожен з яких відповідає за свою лінію порту, яка збігається з номером Px0 .. Px7.

PORTx (The Port X Data Register) – регістр даних порту «x». Принцип роботи з цим регістром залежить від того, в якому режимі, входу або виходу

працює лінія порту. Якщо лінія порту працює на вхід, то біти даного регістра відповідають за підключення до лінії внутрішнього опору, який підтягує напругу лінії до напруги живлення. Якщо лінія порту працює на вихід, то біти даного регістра керують станом вихідної лінії.

PINx (The Port X Input Pins Address) – діапазон адрес, читання по яким надає доступ до інформації з буферних регістрів на вході порту «x». Даний регістр призначений тільки для зчитування станів ліній порту.

Навчальна плата має 4 лінійки світлодіодів (8 світлодіодів, запаяних в рядок). Світлодіоди ввімкнені по схемі з спільним анодом з струмообмежувальним резистором для кожного світло діода окремо. Відповідно, катоди світлодіодів підключені безпосередньо до портів мікроконтролера. Щоб засвітити світлодіоди деяким чином, потрібно записати в “0” у відповідний біт, а для вимкнення – “1”.

При виконанні лабораторних робіт для розробки програм рекомендовано використання WinAVR.

Приклад програми

Наступна програма забезпечує таку функцію: світлодіод приєднано до виходу PB.4. Він блимає з частотою близько 1 Гц.

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/
#include <avr/io.h>
#include <util/delay.h>
int main(void)
{
  DDRB = 0xFF;
  PORTB = 0xFF;
  DDRD = 0xFF;
  PORTD = 0xFF;
    for(;;)
{
          //PORTB=0; //управляє одразу всим портом /
    //PORTB &= ~_BV(4);// управляє конкретним піном. Записує "0" в PB.4
    PORTB=0b00000000;
        for(unsigned char d = 50; d>0; d--) _delay_ms(10); //delay 500 ms

        //PORTB=0x10;//одиниця на PB.4.Всі інші піни - 0
        //PORTB |= _BV(4);// управляє конкретним піном. Записує "1" в
PB.4
        PORTB=0b00010000;
```

```
for(unsigned char d = 50; d>0; d--) _delay_ms(10); //delay 500 ms
  }
return 0;
}
```

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

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

Перш за все відбувається налаштування портів. Команда DDRx вказує напрям роботи. При передачі команді «1» – відповідний пін порту налаштовується на «вихід». При передачі «0» - відповідний пін порту налаштовується на «вхід». Команда PORTx налаштовує стан порту на початку програми.

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

Функція _delay_ms(10) з бібліотеки util/delay.h організує затримку 10 мілісекунд. Написати _delay_ms(500) не можна, бо максимальне значення аргументу 262.14 мс, поділене на тактову частоту в МГц. Безпечне значення для усіх мікроконтролерів AVR та тактових частот є 10 мс. Щоб зробити затримку 0,5 с (500 мс), потрібно повторити 50 разів затримку 10 мс.

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

Алгоритм роботи програми

Розглянемо, як працює наведена програма. На початку здійснюється налаштування портів мікроконтролера. Потім МК входить у нескінченний цикл. Вмикається світлодіод, потім пауза 500 мс, вимикається світлодіод, знову пауза 500 мс. Далі цикл повторюється до нескінченності.

Блимання деяких світлодіодів

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

число записане у шістнадцятковому вигляді. Щоб його отримати, запишемо значення всіх восьми бітів, починаючи з найстаршого.

Таблиця 2.1

Приклади кодування станів поту в шістнадцятко вій системі

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

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

Таблиця 2.2

Таблиця відповідності двійкових значень шістнадцятковим

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

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

Хід роботи

1. Розібратися з принципом роботи тестової програми та призначенням кожного оператора у програмі.
2. Створити свою програму згідно з індивідуальним завданням та скомпілювати її.
3. Перевірити дієздатність створеної програми в середовищі Proteus.
4. Під'єднати внутрішньосхемний програматор до плати та комп'ютера. Увімкнути плату.
5. Запрограмувати МК.
6. Перевірити правильність виконання програми.

Індивідуальні завдання

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

Таблиця 2.3

Варіанти індивідуальних завдань

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

Контрольні запитання

1. Кодування чисел в двійкову та шістнадцяткову системи числення;
2. Час виконання команд (поняття такту, машинного циклу);
3. Призначення портів введення/виведення AVR-мікроконтролерів;
4. Структура порту введення/виведення AVR-мікроконтролерів;
5. Регістри керування портами введення/виведення AVR-мікроконтролерів та їх призначення.

Зміст звіту

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

Лабораторна робота № 2
СТВОРЕННЯ ЗАПРОГРАМОВАНИХ ПОСЛІДОВНОСТЕЙ ВИХІДНИХ СИГНАЛІВ

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

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATtiny2313, ATmega8, ATmega16, Atmega8515; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал

Вогонь, що біжить

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

Ось програма, що створює ефект вогню, що біжить. У якості вогнів використовується лінійка світлодіодів порту B.

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

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

int main(void)
{
DDRB = 0xFF;
  PORTB = 0xFF;
  unsigned char i=0;
for(;;)
{
if(++i >= 8) i = 0;
PORTB =(1 << i);
for(  unsigned char d = 10; d>0; d--) _delay_ms(10); //delay 100 ms
}
return 0;
}
```

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

Ефекти зі зсувом значень

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

Рис. 2.2. Приклад зсуву значень вліво

Нове значення молодшого біту може бути:

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

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

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

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

int main(void)
```

```
{
DDRB = 0xFF;
  PORTB = 0xFF;
  unsigned char state=0;
  unsigned char next;
for(;;)
{ next = !(state & 0x40);
state <<= 1;
if(next) state |= 1;
PORTB = (state);
for(unsigned char d = 25; d>0; d--) _delay_ms(10); //delay 250 ms
}
return 0;
}
```

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

Запрограмована послідовність

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

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

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
unsigned char states[] = { 1, 3, 7, 0x0F, 0x1F, 0x3F, 0x7F,
0xFE, 0xFC, 0xF8, 0xF0, 0xE0, 0xC0, 0x80 };

unsigned char i=0;
for(;;)
{
 if(++i >= sizeof(states)) i=0;
PORTB=(states[i]);
for(unsigned char d = 25; d>0; d--) _delay_ms(10); //delay 250 ms
```

```
}
return 0;
}
```

Хід роботи

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

Індивідуальні завдання

Зробити описаний світловий ефект. Ефект має повторюватися до нескінченності.

1. Вогонь, що біжить, пробігає 3 рази в одну сторону і 3 рази в іншу.
2. Перший і останній вогонь біжать до центру, назустріч один одному, 3 рази. Потім вони розбігаються від центру до країв, також 3 рази.
3. Два вогні, що біжать, пробігають 2 рази в одну сторону і 2 рази в іншу.
4. Спочатку загораються перші 2 світлодіода, вимикаються, потім загораються два наступні, вимикаються, наступною вмикаєтьсяі вимикається третя пара, потім це ж повторює остання пара, потім загораються і вимикаються всі одразу. На останок один вогник пробігає зліва на право, а потім справа на ліво.
5. Вогні перший і п'ятий, що біжать, пробігають 2 рази в одну сторону і 2 рази в іншу.
6. Загоряється перший і останній світлодіод, потім другий і передостанній і так до центру. Потім всі світлодіоди погасають. Після цього світлодіоди загоряються з центру до країв по два, потім всі разом погасають.
7. Вогні перший, другий, п'ятий і шостий, що біжать, пробігають 2 рази в одну сторону і 2 рази в іншу.
8. Загоряється перший і п'ятий світлодіод, до них долучаються другий і шостий, потім третій і сьомий, далі четвертий і восьмий. Потім гаснуть послідовно з першого до останнього. Далі загоряються так само, а гаснуть з останнього до першого.
9. Перший і останній вогонь біжать до центру, назустріч один одному. Потім вони розбігаються від центру до країв.
10. Тінь, що біжить, пробігає 3 рази в одну сторону і 2 рази в іншу.
11. Перший і останній світлодіод блимають з частотою 0,25 Гц, другий і передостанній у 2 рази швидше (0,5 Гц) і так до середини частота блимання збільшується вдвічі.
12. Світлодіоди послідовно вмикаються, з першого до останнього, а потім погасають, також з першого до останнього. Потім вмикаються з останнього до першого та погасають з останнього до першого.

13. Світлодіоди послідовно вмикаються, з першого до останнього, а потім погасають у зворотньому напрямку, з останнього до першого.

14. Спочатку всі світлодіоди горять. Погасають перший і останній, потім другий і передостанній, і так до центру. Далі всі загоряються. Потім світлодіоди погасають з центру до країв, після чого всі загоряються.

15. Загоряється перший і другий світлодіод, до них долучаються сьомий і восьмий, потім третій і четвертий, далі п'ятий і шостий. Потім гаснуть послідовно з першого до останнього. Далі загоряються так само, а гаснуть з останнього до першого.

Контрольні запитання

1. Побітове додавання та віднімання.
2. Логічні операції & та | .
3. Використання виключаючого або ^.
4. Принцип зсуву бітів.
5. Привести приклад оголошення масиву значень.
6. Як визначається довжина масиву?

Зміст звіту

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

Лабораторна робота № 3
ОРГАНІЗАЦІЯ ДИНАМІЧНОЇ ІНДИКАЦІЇ

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

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATtiny2313, ATmega8, ATmega16, Atmega8515; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал

Дуже часто МК використовується не тільки для керування роботою конструкції, але й для того, щоб повідомити що-небудь користувачеві. Наприклад, електронний годинник, крім власне відліків часу, повинен його ще відображати, а також дозволяти змінювати покази (встановлювати точний час). Якщо вся "інформація" зводиться до мигання парою світлодіодів, яких-небудь спеціальних зусиль з відображення інформації з боку розробника конструкції не вимагається, але якщо таких світлодіодів виявляється два-три десятки, тут вже потрібне застосування додаткових засобів (як апаратних, так і програмних). Як правило, в цьому випадку відображення інформації виконують у режимі динамічний індикації – це найбільш економний за кількістю використовуваних ліній спосіб.

Найбільш часто динамічну індикацію застосовують при здійсненні відображення інформації на семисегментних індикаторах, в яких стилізоване зображення цифр (і деякого набору букв) складають із семи лінійних сегментів, розташованих у вигляді цифри вісім (рис. 2.3). Висвічування сегмента, що вибирається, чи групи сегментів при отриманні зображення знаку забезпечується ввімкненням їх в коло проходження струму.

Рис 2.3. Зображення знаків на основі семисегментного індикатора

Кожен сегмент світлодіодного індикатора е звичайним світло діодом. А їх спільне ввімкнення визначає тип індикатора (з спільним анодом чи катодом). Якщо використовується одноцифровий елемент – при підключенні можна обмежитись одним портом для управління сегментами і приєднанням спільного електрода до плюса чи мінуса. В разі використання декількох цифрових елементів – використовують динамічну індикацію. При такому режимі розряди

індикатора працюють не одночасно, а по черзі. Переключення розрядів відбувається з великою швидкістю (50 Гц), через це людське око не помічає, що індикатори працюють по черзі.

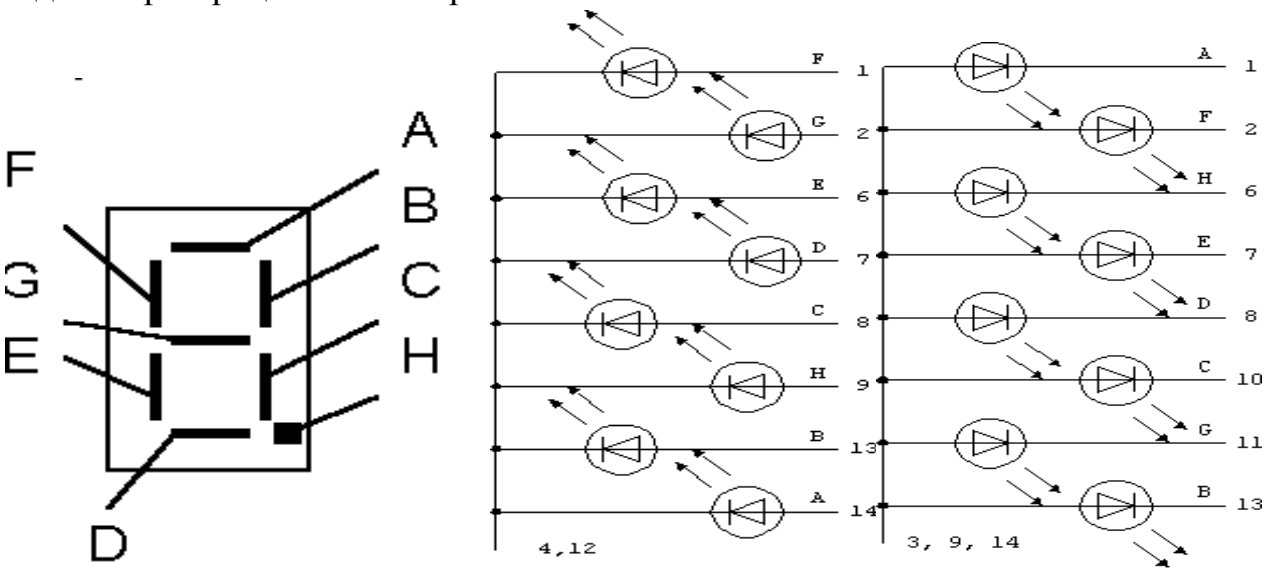


Рис. 2.4. Цифрові світлодіодні індикатори та їх схемні рішення

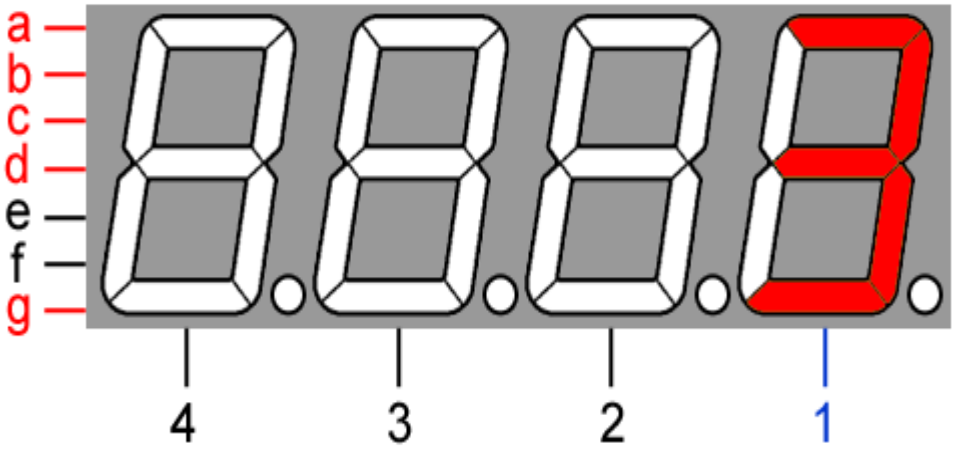


Рис. 2.5. Режим динамічної індикації

Так як у світлодіодів дуже мала інерційність, розряди, що змінюються зливаються в одне зображення. У цьому режимі в кожен момент часу працює тільки один розряд, вмикаються по черзі починаючи з першого закінчуючи останнім, потім все починається спочатку.

Динамічний спосіб відображення інформації базується на тому, що будь-який світловий індикатор є інерційним приладом, а для людського ока зображення на дисплеї , якщо його оновлювати із частотою приблизно 20 разів в секунду, представляється незмінним.

Схема реалізації динамічної індикація без додаткових елементів наведена на рис. 2.6. До порту В МК підключені катоди всіх світлодіодів матриці, а до порту А – аноди кожного з індикаторів, що створюють матрицю. На лініях порту А організовується одиниця, що "біжить". На лінії порту В при кожному положенні одиниці, що біжить, виводиться семисегментний код того символу,

який повинен горіти в даному знакомісці. Для індикаторів із загальним катодом замість одиниці, що біжить, використовується нуль, що біжить. Перевага такого способу індикації – у відсутності яких-небудь додаткових компонентів (окрім самих світлодіодних індикаторів), головний недолік – значна перевитрата ліній портів.

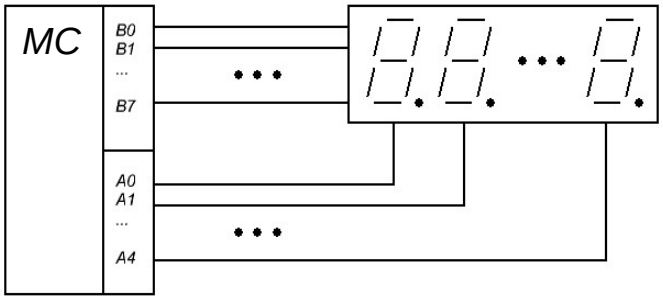


Рис. 2.6. Підключення світлодіодного індикатора без допоміжних елементів

На навчальній платі використовується семи сегментний чотирьох розрядний світлодіодний індикатор з спільним катодом, підключений до порту B. Це значить, що в разі його приєднання за схемою, приведеною на рис. 2.7 – керувати засвічуванням одного елемента ми повинні були низьким рівнем сигналу («0»).

Проте на платі використовується підхід з використанням драйвера на транзисторних ключах для управління засвічуванням кожного з елементів. Тобто, для засвічування кожного елемента потрібно використовувати високий рівень сигналу («1»), як у випадку світлодіодного індикатора з спільним анодом.

Виведення цифр на світлодіодний індикатор

Наведемо приклад виведення на світлодіодний індикатор номеру кожного з сегментів. (На першому сегменті – цифра «1», на другому – «2», і т.д.)

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

//  A   B     C     D     E     F     G     H(dp)
// B0  B1    B2    B3    B4    B5    B6    B7

//      Digit1      Digit2      Digit3      Digit4
//      D2          D3          D4          D5

#include <avr/io.h>
```

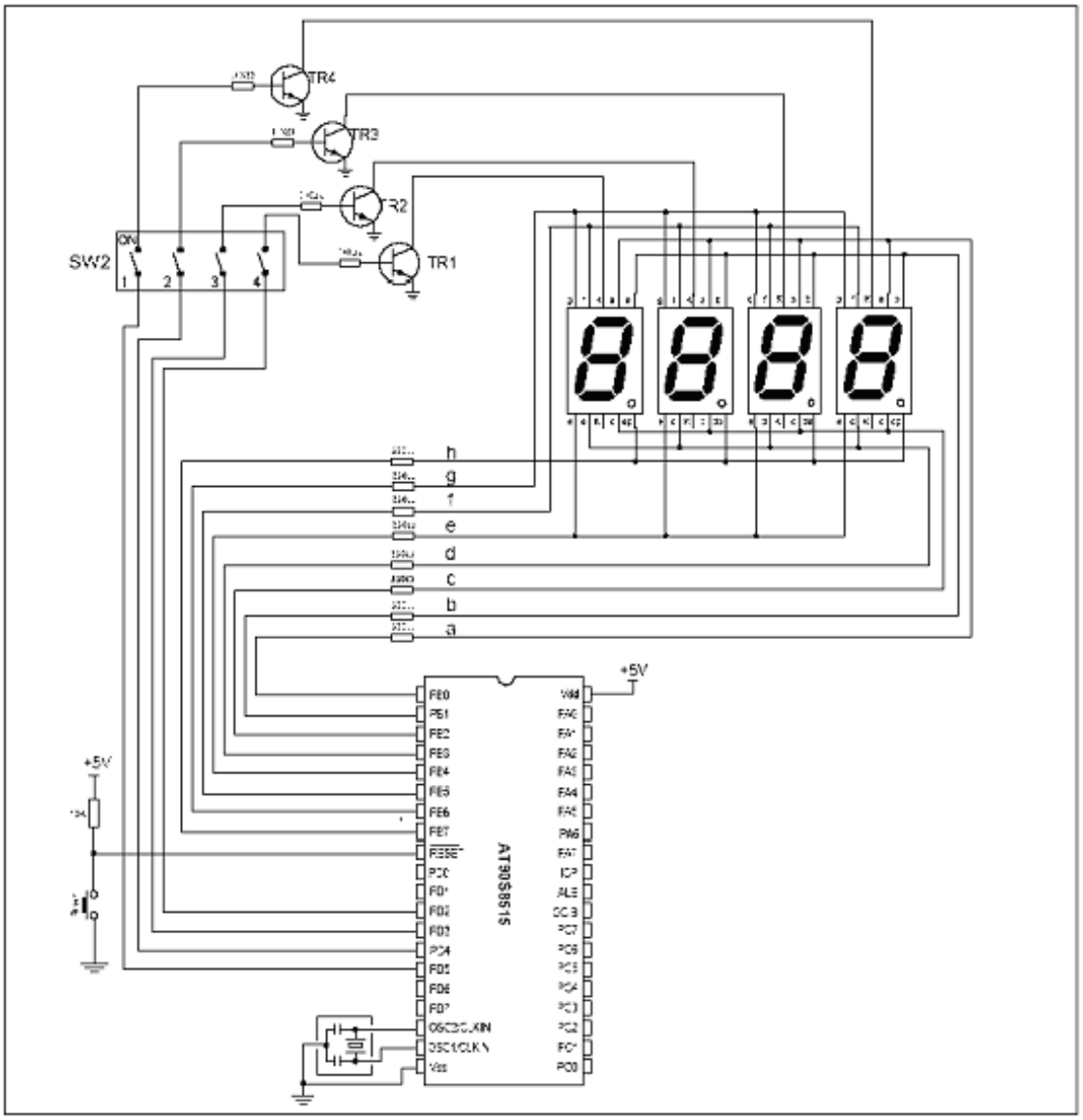


Рис. 2.7. Схема ввімкнення світлодіодного індикатора на навчальній платі

```
#include <util/delay.h>

int main(void)
{
DDRB = DDRD = 0xFF;
PORTB = PORTD = 0xFF;
unsigned char i=20; //затримка мікросекунд

for(;;)
{
  PORTD = 0b00000100; //вмикаємо розряд "1", всі інші - вимикаємо
        PORTB = 0b00000110;   // комбінація вмикає цифру "1"
  _delay_us(i);                 //часова затримка в мікросекундах на
свічення даного розряду

        PORTD = 0b00001000;   //вмикаємо розряд "2", всі інші - вимикаємо
```

```
        PORTB = 0b11011011;   // комбінація вмикає цифру "2"
        _delay_us(i);

        PORTD = 0b00010000;   //вмикаємо розряд "3", всі інші - вимикаємо
        PORTB = 0b01001111; // комбінація вмикає цифру "3"
        _delay_us(i);

        PORTD = 0b00100000;   //вмикаємо розряд "4", всі інші - вимикаємо
        PORTB = 0b01100110; // комбінація вмикає цифру "4"
        _delay_us(i);
    }
    return 0;
    }
```

На початку програми усі лінії портів B та D встановлюються як виходи з високим рівнем вихідного сигналу.

В нескінченному циклі відбувається почергове засвічення кожного з розряду індикатора (порт D) з одночасним вмиканням на портові B комбінації пінів, яка створює на екрані ту чи іншу цифру.

Хід роботи

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

Індивідуальні завдання

1. Пропонується вивести на світлодіодний індикатор дату свого народження в форматі «день народження, крапка, місяць народження».

Контрольні запитання

1. Назвати існуючі типи семи сегментних світлодіодних індикаторів.
2. Розкрити принцип динамічної індикації.
3. Як кодується зображення довільного символу на семисегиентному світлодіодному індикаторі?
4. Як розраховується час свічення кожного сегменту для динамічного методі відображення інформації?
5. Схеми ввімкнення одиничних семисегментних світлодіодних індикаторів.

Зміст звіту

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

Лабораторна робота № 4
ВИКОРИСТАННЯ КНОПОК ДЛЯ КЕРУВАННЯ РОБОТОЮ МІКРОКОНТРОЛЕРА

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

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATtiny2313, ATmega8, ATmega16, Atmega8515; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал

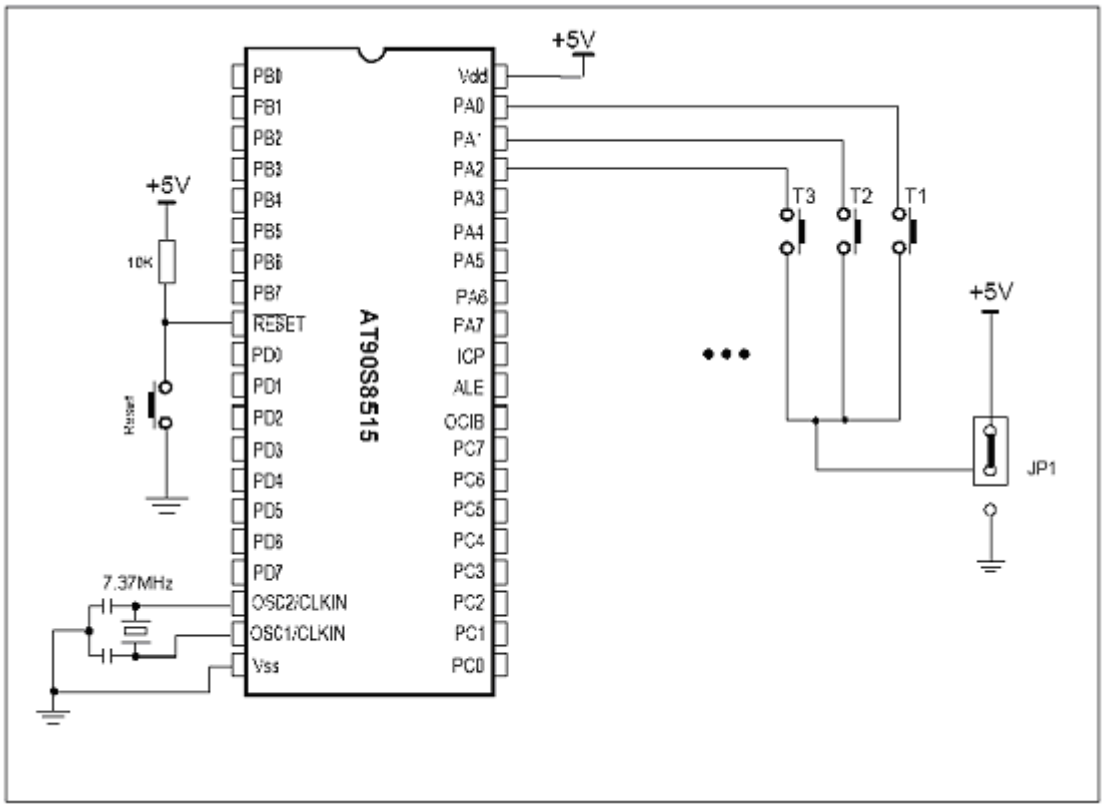


Рис. 2.8. Схема підключення кнопок до мікроконтролера на навчально-відлагоджувальній платі AVR-Easy

Навчальна відлагоджувальна плата містить кнопку початкової установки контролера RESET і 24 кнопки, що дозволяють симулювати вхідні дії на порти A, B, D. Загальний вивід кнопок може бути підключений до ланцюга живлення або "землі" за допомогою JP1. Це дає змогу посилати в порти мікроконтролера сигнали високого або низького рівня.

Управління світлодіодами за допомогою кнопок

Якщо натиснута кнопка D0, горить перша половина лінійки світлодіодів порту B, а якщо D1 – то друга половина. Кожна кнопка вмикає свою половину світлодіодів незалежно одна від одної.

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

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

int main(void)
{
DDRB = 0xFF;
PORTB = 0xFF;
DDRD = 0x00;
PORTD = 0xFF;
for(;;)
{
if(bit_is_clear(PIND, PD2)) PORTB = 0x0F;
if(bit_is_clear(PIND, PD3)) PORTB = 0xF0;
else PORTB=0xFF;
}
return 0;
}
```

На початку програми усі лінії порту B встановлюються як виходи з високим рівнем. Усі лінії порту D встановлюються як входи зі внутрішніми резисторами.

Управління індикаторами за допомогою кнопок

Якщо натиснута кнопка D0, на лівій частині індикатора горить число 99, інакше – 11. Якщо натиснута кнопка D1– то на правій частині горить 87, інакше – 23. Кожна кнопка управляє своєю половиною індикатора незалежно одна від одної. Завжди горять перша і четверта десяткові крапки.

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

// A   B      C      D      E      F      G      H(dp)
```

```
// B0  B1    B2    B3    B4    B5    B6    B7

// Digit1     Digit2Digit3Digit4
//      D2           D3           D4           D5

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

int main(void)
{
DDRB =  0xFF;
  DDRD = 0b00111100;
  PORTB = PORTD = 0xFF;
  unsigned char i=20; //затримка мікросекунд

for(;;)
{
PORTD |= _BV(2); PORTD &= ~_BV(3); PORTD &= ~_BV(4); PORTD &= ~_BV(5); //вмикаємо розряд "1", всі інші - вимикаємо
        if(bit_is_clear(PIND, PD0)) PORTB = 0b00000110; // комбінація вмикає цифру "1"
        else PORTB = 0b01101111;     // комбінація вмикає цифру "9"
        _delay_us(i);                //часова затримка в мікросекундах на свічення даного розряду

        PORTD &= ~_BV(2); PORTD |= _BV(3); PORTD &= ~_BV(4); PORTD &= ~_BV(5); //вмикаємо розряд "2", всі інші - вимикаємо
        if(bit_is_clear(PIND, PD0)) PORTB = 0b00000110; // комбінація вмикає цифру "1"
        else PORTB = 0b01101111;     // комбінація вмикає цифру "9"
        _delay_us(i);

        PORTD &= ~_BV(2); PORTD &= ~_BV(3); PORTD |= _BV(4); PORTD &= ~_BV(5); //вмикаємо розряд "3", всі інші - вимикаємо
        if(bit_is_clear(PIND, PD1))   PORTB = 0b01011011; // комбінація вмикає цифру "2"
        else PORTB = 0b01111111;     // комбінація вмикає цифру "8"
        _delay_us(i);

        PORTD &= ~_BV(2); PORTD &= ~_BV(3); PORTD &= ~_BV(4); PORTD |= _BV(5); //вмикаємо розряд "4", всі інші - вимикаємо
        if(bit_is_clear(PIND, PD1)) {PORTB = 0b01001111;} // комбінація вмикає цифру "3"
```

```
            else PORTB = 0b00000111;      // комбінація вмикає цифру "7"
            _delay_us(i);
    }
    return 0;
    }
```

Хід роботи

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

Індивідуальні завдання

1. На індикаторі світиться число 0. Коли натискають будь-яку кнопку, число збільшується на 1111 (вийдуть числа 1111, 2222 тощо) з інтервалом 300 мс. Після досягнення числа 9999 збільшення припиняється і програма повертається у початковий стан.
2. На індикаторі світиться число 1234. Коли натискають першу кнопку, число збільшується на 100, коли другу — на 1.
3. На індикаторі світиться число 9999. При натисненні першої кнопки остання цифра зменшується на 1 (виходить 9998), при наступному натисненні — третя цифра зменшується на 1 (виходить 9988), аналогічно для другої і першої цифри. Потім зменшується знову остання цифра.
4. Коли не натиснута жодна кнопка, світиться число 1111, коли натиснута перша — 8811, друга — 1188, обидві — 8888.
5. На індикаторі світиться число 4444. Коли натискають першу кнопку, число зменшується вдвічі, коли другу — збільшується на 123.
6. На індикаторі світиться число 9999. Коли натискають будь-яку кнопку, число зменшується на 1111 (вийдуть числа 8888, 7777 тощо) з інтервалом 300 мс. Після досягнення числа 0000 зменшення припиняється і програма повертається у початковий стан.
7. Коли не натиснута жодна кнопка, світиться число 9999, коли натиснута перша — 4949, друга — 9494, обидві — 4444.
8. На індикаторі світиться число 4320. Коли натискають будь-яку кнопку, число збільшується на 40 (вийдуть числа 4360, 4400 тощо) з інтервалом 300 мс. Після досягнення числа, більшого за 5000, збільшення припиняється і програма повертається у початковий стан.
9. На індикаторі світиться число 8675. Коли натискають будь-яку кнопку, число зменшується на 85 (вийдуть числа 8590, 8505 тощо) з інтервалом 300 мс. Після досягнення числа, меншого за 7500, зменшення припиняється і

програма повертається у початковий стан.

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

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

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

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

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

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

Контрольні запитання

1. Привести основні можливі схемні рішення для приєднання кнопок до мікроконтролерів.
2. Для кожного з приведених схемних рішень включення кнопок привести приклади попередньої ініціалізації порту та подальшого програмного методу опрацювання події натиснення на кнопку.

Зміст звіту

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

Лабораторна робота № 5
РІДКОКРИСТАЛІЧНИЙ ІНДИКАТОР

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

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATtiny2313, ATmega8, ATmega16, Atmega8515; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал

Алфавітно-цифрові рідкокристалічні індикатори

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

LCD на навчальній платі

На навчальній платі встановлено індикатор розміру 16x2.

LCD під'єднаний по 4-бітному інтерфейсу до порту B.

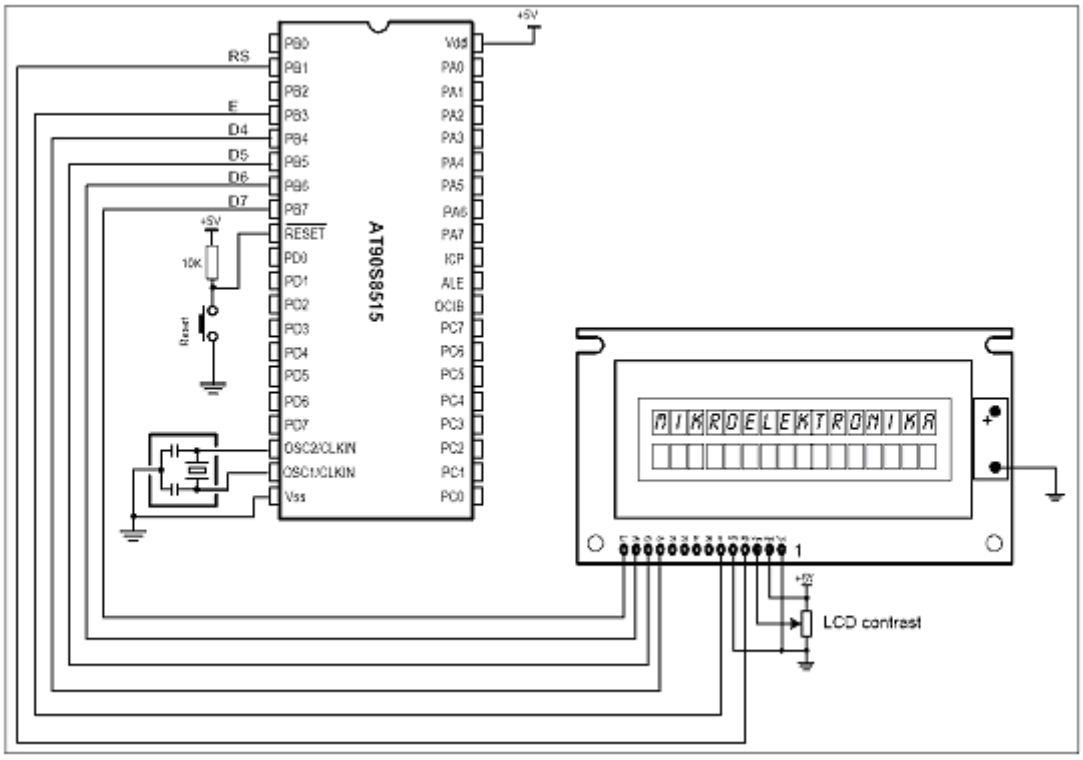


Рис. 2.9. Схема підключення алфавітно-цифрового рідкокристалічного індикатора до мікроконтролера на навчально-відлагоджувальній платі AVR-Easy

Бібліотека для роботи з РК-дисплеєм

Для зручності написання програм з використанням РК індикаторі – доречно винести всі команди і процедури для роботи з індикатором в окрему бібліотеку. Для комфортної роботи достатньо підготувати бібліотеку з трьома процедурами: процедура ініціалізації, запису даних, запису команди. Також існує ще можливість зчитування даних з індикатора, проте схема підключення індикатора на навчальній платі не передбачає використання таких процедур. Оскільки управляючий пін напряму передачі даних приєднано не до порту мікроконтролера, а до мінуса, що визначає його режим роботи тільки на запис. Для можливості зчитування даних з екрану передбачити можливість подачі сигналу високого рівня на даний пін. Особливу увагу слід приділяти написанню процедури ініціалізації дисплею. Без належної і правильної ініціалізації (попереднє налаштування режиму роботи) жоден екран працювати не буде. В приведеній бібліотеці приведена найбільш загальна процедура, описана в DATASHEET-і на HD44780. Приведена форма не є остаточною і може видозмінюватись в залежності від схеми та вимог розробника. Для функціонування даної бібліотеки потрібно щоб в основній програмі була підключена бібліотека затримок “util/delay.h”.

В бібліотеці передбачено настуані процедури:

lcd_init( ); – процедура ініціалізації РК-екрану. Викликається на почтку виконання програми перед початком нескінченного циклу.

lcd_com( ); – функція запису команди в РК-екран. Може використовуватись для переведення курсору в певне положення на екрані, очистки екрану, вибору форми відображення курсору і т.д. Наприклад, якщо передати функції код ‘0x80’, то результатом її виконання буде переведення курсору на початок першого рядка.

lcd_dat( ); – функція запису даних в РК-екран. Наприклад, якщо передати даній функції символ ‘A’ чи його код ‘0x41’, то результатом її виконання буде виведення на екрані в позиції курсору символу ‘A’.

Виведення тексту на LCD

Дана програма ініціалізує LCD і виводить текст “Hello, LCD!!! 16 chars, 2 lines.”

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

#include <avr/io.h>
#include <util/delay.h>
#include "LCD.C"                    //Бібліотека для роботи з РК-екраном

int main(void)
```

```
{
DDRB = PORTB = 0xFF;
unsigned char text[] =  "-=Hello, LCD!!=-16 chars,2 lines.";
lcd_init();    //Ініціалізація РК-екрану
lcd_com(0x80); //Переведення курсору на початок першого рядка

for(unsigned char i=0; i<32; i++)
{ if(i == 16) lcd_com(0xC0); //Переведення курсору на початок другого рядка, якщо текст виходить за межі першого
lcd_dat(text[i]); //Виведеня і-го символу з масиву
}

for(;;);

}
```

У нескінченному циклі процесор нічого не робить, але вилучити цикл не можна, щоб МК не скидався і не починав виконувати усю програму спочатку.

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

Символи кирилиці

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

Рядок, що біжить

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

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

#include <avr/io.h>
#include <util/delay.h>
#include "LCD.C"        //Бібліотека для роботи з РК-екраном

unsigned char text[ ] = { '-', 0x3D, 0x3D, 0x20, 0xA8, 0xBB, 0x61, 0xBD,
```

```
0x65, 0xBF, 0xB8, 0x20, 0x20, 0x3D, 0x3D, '-' };
        unsigned char text1[] = { 0x20, 0x20, 0x43, 0x6F, 0xBD, 0xC7, 0xC0, 0xBD,
0x61, 0x20, 0x63, 0xB8, 0x63, 0xBF, 0x65, 0xBC, 0x61, 0x20, 0xBC, 0x69, 0x63,
0xBF, 0xB8, 0xBF, 0xC4, 0x20, '8', 0x20, 0xBE, 0xBB, 0x61, 0xBD, 0x65, 0xBF,
0x3A, 0x20, 0x4D, 0x65, 0x70, 0xBA, 0x79, 0x70, 0x69, 0xB9, 0x2C, 0x20, 0x42,
0x65, 0xBD, 0x65, 0x70, 0x61, 0x2C, 0x20, 0xA4, 0x65, 0xBC, 0xBB, 0xC7, 0x2C,
0x20, 0x4D, 0x61, 0x70, 0x63, 0x2C, 0x20, 0xB0, 0xBE, 0x69, 0xBF, 0x65, 0x70,
0x2C, 0x20, 0x43, 0x61, 0xBF, 0x79, 0x70, 0xBD, 0x2C, 0x20, 0xA9, 0x70, 0x61,
0xBD, 0x2C, 0x20, 0x48, 0x65, 0xBE, 0xBF, 0x79, 0xBD,'.' };

        int main(void)
        {
        DDRB = PORTB = 0xFF;
        lcd_init();             //Ініціалізація РК-екрану

        lcd_com(0x80);      //Переведення курсору на початок першого рядка
        for(unsigned char i=0; i<16; i++) lcd_dat(text[i]);   //Виведення першого
напису

        for(;;)
        { for(unsigned char offset=0; offset<=sizeof(text1)-16; offset++)
          { lcd_com(0xC0);//Переведення курсору на початок другого рядка
            for(unsigned char i=0; i<16; i++) lcd_dat(text1[offset+i]); //Виведення 16-
ти символів
            for(unsigned char i=0; i<40; i++) _delay_ms(10); //delay 400 ms
          }
        }

        }
```

Оскільки значення першого рядка не змінюються, він передається у LCD один раз на початку програми. У нескінченному циклі передається лише значення другого рядка.

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

Хід роботи

1. Знайти на навчальній платі рідкокристалічний індикатор.
2. Розібратися з принципом роботи тестових програм та призначенням кожного оператора у програмі.

3. Створити свою власну програму, яка робить рядок, що біжить. Текст, який біжить, має складатися з Вашого прізвища та будь-якого речення кирилицею. Назва тексту відображається на першому рядку. Кнопка виконує функцію “Пауза/Продовження” (при одному натисненні рядок припиняє бігти, при повторному — продовжує з того ж місця, де зупинився). Скомпілювати програму.

4. Запрограмувати МК, перевірити правильність виконання.

Контрольні запитання

1. Якими можливостями володіють алфавітно-цифрові рідкокристалічні індикатори? Які ще є рідкокристалічні індикатори?
2. Привести існуючі схемні рішення для підключення алфавітно-цифрових рідкокристалічних індикаторів.
3. Яке призначення кожного з пінів алфавітно-цифрових рідкокристалічних індикаторів.
4. Який алгоритм роботи з алфавітно-цифровим рідкокристалічним індикатором?
5. Пояснити кожну з процедур для роботи з алфавітно-цифровим рідкокристалічним індикатором.

Зміст звіту

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

Лабораторна робота № 6
РІДКОКРИСТАЛІЧНИЙ ІНДИКАТОР

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

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATmega16, Atmega8515; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал
Додаткові можливості алфавітно-цифрових рідкокристалічних індикаторів

Кожний LCD-індикатор має вбудований знакогенератор, що являє собою область ПЗП об'ємом понад 8 Кб, яка прошивається на заводі виробнику. Традиційно перша половина ПЗП з адресами 00 – 7Fh містить зображення цифр, знаків пунктуації а також великих та малих букв латинського алфавіту. Друга половина відведена для національних алфавітів. В зв'язку з цим HD44780 має модифікації виконання з трьома основними варіантами прошивки знакогенератора:

- латиниця та європейські мови (European standard font или Euro);
- латиниця та японські ієрогліфи (Japanese standard font або Japan) ;
- латиниця і кирилиця (Custom font или Russian, Додаток).

Не всі з комірок знакогенератора заповнені. При зверненні до порожніх комірок на екрані буде виводитись довільна інформація. Перші 16 символів з адресами 0x00 – 0x0F відмічені " * ". при бажанні вони можуть бути самостійно запрограмовані користувачем.

Виведення довільних символів на LCD

Дана програма ініціалізує LCD і виводить на екран зафарбований прямокутник, дзвінок та знак відсотків.

```
/*
Target MCU: ATmega8
Target device: AVR-Easy
*/

#include <avr/io.h>
#include <util/delay.h>
#include "LCD.C"                    //Бібліотека для роботи з РК-екраном

int main(void)
```

```
{
DDRB = PORTB = 0xFF;
lcd_init();    //Ініціалізація РК-екрану
lcd_com(0x80); //Переведення курсору на початок першого рядка

lcd_dat(0xFF); //Виведеня чорного прямокутника
lcd_dat(0xED); //Виведеня символу “ дзвінка “
lcd_dat(0x25); //Виведеня символу ” % ”

for(;;);

}
```

У нескінченному циклі процесор нічого не робить, але вилучити цикл не можна, щоб МК не скидався і не починав виконувати усю програму спочатку.

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

Хід роботи

1. Знайти на навчальній платі рідкокристалічний індикатор та кнопки.
2. Повторити принципи роботи з рідкокристалічними індикаторами та кнопками.
3. Створити свою власну програму, яка забезпечує функції, описані в індивідуальному завданні, та скомпілювати її.
4. Запрограмувати МК, перевірити правильність виконання.

Індивідуальні завдання

1. При натисненні будь-якої з кнопок на індикаторі у верхньому рядку з’являється назва порту, до якого належить кнопка, а в нижньому – номер піну, до якого приєднана кнопка. При цьому в кожного напису передбачено своє власне місце.
2. На екрані написано слово «МАМА». Реалізувати програму, в якій, при натисненні однієї з чотирьох довільно обраних кнопок, слово переміщується в одному з чотирьох напрямків. (верх, низ, право, ліво).
3. Реалізувати модель друкарської машинки, в якій кожній з кнопок призначена певна буква. Результатом роботи повинне бути наступне: при послідовному натисненні кнопок – на екрані виводиться напис, що набирається. Одна з кнопок відповідає за очистку екрану.
4. Існує довільний текст з розміром не менше 50 символів. Реалізувати програму, що реалізує виведення тексту на екран. При ініціалізації роботи плати на екрані виводиться текст з початку. Передбачити можливість прокрутки тексту при натисненні довільно обраних кнопок.
5. Реалізувати модель інженерного калькулятора, в якій кожній з кнопок призначена певна цифра, чи деякий символ. Результатом роботи повинне бути наступне: при послідовному натисненні кнопок – на екрані

виводиться математична функція, що набирається. Одна з кнопок відповідає за очистку екрану. Обрахунки виводити не потрібно!!!

6. Реалізувати програму, в якій при натисненні однієї кнопки на екран послідовно виводяться всі можливі символи, що не є цифрами чи буквами. При заповненні екрану курсор переходить на перший рядок і продовжується виведення нових символів. Передбачити кнопку стирання остання останнього символу та кнопку повної очистки екрану.

7. Дві кнопки виконують роль лічильника. Одна кнопка збільшує лічильник, інша – зменшує. Розмірність лічильника – 16. Реалізувати програму, в якій посередині верхнього рядка виводиться значення лічильника, а нижній рядок заповнюється чорними квадратами пропорційно до значення лічильника.

8. Екран заповнений довільним текстом або символами. Реалізувати програму в якій довільно визначені чотири кнопки управляють блимаючим курсором. При цьому передбачена ще дна кнопка, яка стирає символ в позиції курсору.

9. Реалізувати ефект шахової дошки розміром 2x8 кліток. Передбачити можливість переміщення даного поля вздовж екрану за допомогою двох довільно обраних кнопок.

10. На екран виводиться довільний текст розміром 32 символи. Реалізувати програму в якій довільно визначені чотири кнопки управляють блимаючим курсором. При цьому передбачена ще одна кнопка, яка змінює регістр вибраної букви на протилежний.

11. Реалізувати програму яка б давала можливість при натисненні однієї з 8 кнопок одного порту вибрати довільний символ з діапазону 0xF0 – 0xFF, позначення якого виводиться на перше знакомісце екрану і залишається там допоки не буде зміненим. Довільно визначені чотири кнопки управляють курсором з нижнім підкресленням. При цьому передбачена ще одна кнопка, при натисненні на яку в положенні курсору виводиться вибраний символ.

Контрольні запитання

1. Якими додатковими можливостями володіють алфавітно-цифрові рідкокристалічні індикатори?
2. Як відбувається адресація на рядках екрану алфавітно-цифрових рідкокристалічних індикаторів?
3. Як кодується довільний символ з множини можливих існуючих в пам’яті контролера рідкокристалічного індикатора?

Зміст звіту

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

Лабораторна робота № 7
АНАЛОГО-ЦИФРОВИЙ ПЕРЕТВОРЮВАЧ В МК AVR

Мета роботи: ознайомитись з базовими типами мікроконтролерів AVR. Набути навиків роботи з вбудованим аналого-цифровим перетворювачем, способів обробки виміряних даних та їх виведення на засоби відображення інформації.

Обладнання: навчально-відлагоджувальна плата AVR-Easy; мікроконтролери ATmega16; середовища програмування WinAVR, Codevision AVR, Flowcode for AVR; внутрішньосхемний програматор; програма для прошивки мікроконтролерів AVR8 Burn-O-Mat.

Теоретичний матеріал

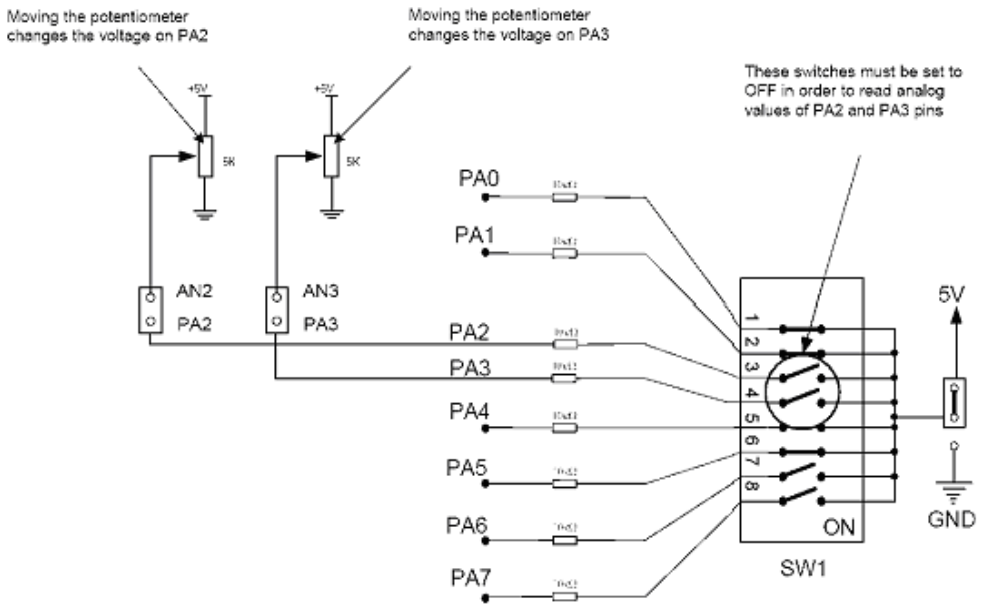


Рис. 2.10. Схема підключення подільників напруги до каналів АЦП мікроконтролерів на навчально-відлагоджувальній платі AVR-Easy

Аналого-цифровий перетворювач мікроконтролерів AVR

Аналого-цифрові перетворювачі (АЦП) є пристроями, які приймають вхідні аналогові сигнали та генерують відповідні їм цифрові сигнали, придатні для обробки мікропроцесорами та іншими цифровими пристроями. Багатоканальний АЦП входить в більшість сучасних моделей МК AVR. Зазвичай число каналів дорівнює 8, але в різних моделях воно може варіювати від 4 каналів в молодших моделях сімейства Tiny, 6 каналів в ATmega8, до 16 каналів в ATmega2560. Багатоканальність означає, що на вході єдиного модуля АЦП встановлений аналоговий мультиплексор, який може підключати цей вхід до різних виводів МК для здійснення вимірювань декількох незалежних аналогових величин з рознесенням по часу. Входи мультиплексора можуть працювати окремо (в несиметричному режимі для виміру напруги відносно "землі") або (в деяких моделях) об'єднуватися в пари для вимірювання

диференціальних сигналів. Іноді АЦП додатково забезпечується підсилювачем напруги з фіксованими значеннями коефіцієнта підсилення 10 і 200. Сам АЦП являє собою перетворювач послідовного наближення з пристроєм вибірки-зберігання і фіксованим числом тактів перетворення, рівним 13 (або 14 для диференціального входу: перше перетворення після ввімкнення потребує 25 тактів для ініціалізації АЦП). Тактова частота формується аналогічно тому, як це робиться для таймерів-за допомогою спеціального дільника тактової частоти МК, який може мати коефіцієнти розподілу від 1 до 128. Але на відміну від таймерів, вибір тактової частоти АЦП не зовсім довільний, так як швидкодія аналогових компонентів обмежена. Тому коефіцієнт ділення слід вибирати таким, щоб при заданій частоті роботи МК тактова частота АЦП укладалася в рекомендований діапазон 50-200 кГц (тобто максимум близько 15 тис. вимірювань в секунду). Збільшення частоти вибірки допустимо, якщо не потрібно досягнення високої точності перетворення. Роздільна здатність АЦП в МК AVR – 10 двійкових розрядів, чого для більшості типових застосувань досить. Абсолютна похибка перетворення залежить від ряду факторів і в ідеальному випадку не перевищує ± 2 молодших розряди, що відповідає загальній точності вимірювання приблизно 8 двійкових розрядів. Для досягнення цього результату необхідно приймати спеціальні заходи: не тільки правильно підбирати тактову частоту в рекомендований діапазон, але і знижувати по максимуму інтенсивність цифрових шумів. Для цього рекомендується, як мінімум, не використовувати невикористані виводи того ж порту, до якого підключений АЦП, для обробки цифрових сигналів, робити правильну розводку друкованої плати, а як максимум – додатково до того ще й включати спеціальний режим ADC Noise Reduction. АЦП може працювати у двох режимах: одиночного і безперервного перетворення. Другий режим доцільний лише при максимальній частоті вибірок. В інших випадках його слід уникати, оскільки обійти в цьому випадку необхідність паралельної обробки цифрових сигналів, як правило, неможливо, а це означає зниження точності перетворення.

Регістри управління АЦП

Для налаштування АЦП існує два регістри: ADCSR (регістр контролю та стану АЦП) та ADMUX (регістр мультиплексора АЦП). У всіх серій МК AVR призначення регістрів та їх бітів налаштування практично не відрізняється.

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Рис. 2.11. Розподілення розрядів в регістрі ADCSRA

ATmega8

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	–	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Всі інші моделі

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Рис. 2.12. Розподілення розрядів в регістрі ADMUX для різних серій мікроконтролерів AVR

Призначення кожного з бітів вказаних регістрів детально описано в даташитах до кожного мікроконтролера.

В ході роботи з АЦП МК потрібно пам’ятати наступне:

1. Не можна починати нове вимірювання, поки не завершилось попереднє.
2. Вимірювання з декількох каналів не може відбуватись одночасно. В будь-який момент часу можлива робота лише з одною, конкретно вибраною лінією АЦП за допомогою регістру ADMUX. Якщо пристрій працює лише з одним каналом АЦП, то достатньо на початку програми в блокові ініціалізації налаштувати потрібний канал. В противному разі потрібно буде перед кожним вимірюванням пере налаштовувати регістр ADMUX для вибору потрібного каналу.
3. При виборі зовнішнього опорного джерела напруги потрібно щоб відповідні виводи МК були до нього приєднані в схемі (виводи AVCC і AREF).

Вимірювання рівня вхідної напруги на каналі PA2 та виведення значення на LCD

Дана програма ініціалізує LCD, проводить вимірювання вхідного аналогового сигналу на каналі PA2 і виводить на екран в першому рядку назву вибраного каналу, а в другому – виміряне значення рівня сигналу.

```
/*
Target MCU: ATmega16
Target device: AVR-Easy
*/

#include <avr/io.h>
#include <util/delay.h>
#include "LCD.C"                    //Бібліотека для роботи з РК-екраном
```

```
unsigned long u=0;
unsigned long voltage=0;
unsigned char i;

 unsigned char text[] = { '-', 0x3D, 0x3D, 0x20, 'A', 'D', 'C', 0x20 ,'P', 'A', '2', 0x20, 0x3D, 0x3D, '-', 0x20, 0x20 };

 //=====Функція зчитування даних з попередньо вибраного каналу АЦП
unsigned int getADC(void)
{  unsigned int v; //локальна змінна
  ADCSRA|=(1<<ADSC);       //почати претворення (ст. 213 Datasheet)
  while       ((ADCSRA&_BV(ADIF))==0x00);       //Чекаємо       закінчення перетворення
  v=(ADCL|ADCH<<8); //Зчитуємо значення АЦП
  return v;
}
//Головна програма
int main(void)
{
DDRB = PORTB = 0xFF; //порт В на вихід, високий рівень
ADMUX =      (0<<REFS1)|  (1<<REFS0)|  (0<<ADLAR)|  (0<<MUX4)| (0<<MUX3)| (0<<MUX2)| (1<<MUX1)| (0<<MUX0);
//_____AVCC з коденсатором на AREF (REFS1 та REFS0) // розрядність 10 біт (ADLAR)//ADC2  (CT.211 Datasheet)
ADCSRA = (1<<ADEN)|(1<<ADPS2)|(0<<ADPS1)|(1<<ADPS0);
//       ADC in  //Тактова частота АЦП СК/32
lcd_init( );   //Ініціалізація РК-екрану
lcd_com(0x80); //Переведення курсору на початок першого рядка
for(unsigned char i=0; i<16; i++) lcd_dat(text[i]);  //Виведення напису

//Безкінечний цикл
while(1)
{
  u=getADC( );                    //Зчитуємо дані з вибраного каналу АЦП
  voltage= 5*u*1000/1024;   //Розрахунок значення напруги. Множення на 1000 для отримання значення в цілочисельному форматі

//=Виводимо отримане значення на РКІ. Обмежуємося 3 знаками після коми  //======== Обмежуємося 3 знаками після коми =========
//=Розкладаємо отримане значення на розряди, після першого ставимо кому ==
```

```
    lcd_com(0xC4);          //Переводимо курсор на другий рядок на 5-ту позицію
    lcd_dat(voltage/1000+0x30);
    lcd_dat(',');
    i=voltage/1000;
    u=voltage-i*1000;
    lcd_dat((u/100)+0x30);
    i=voltage/100;
    u=voltage-i*100;
    lcd_dat((u/10)+0x30);
    lcd_dat(voltage%10+0x30);
    lcd_dat(' ');
    lcd_dat('V');
    for(unsigned char d = 50; d>0; d--) _delay_ms(10); //Затримка 500 ms
  }
  }
```

У нескінченному циклі процесор викликає процедуру зчитування даних з вибраного каналу АЦП та виводить обраховане значення на екран.

Хід роботи

1. Знайти на навчальній платі рідкокристалічний індикатор, змінні резистори для регулювання рівня вхідного сигналу на канал АЦП, лінійки світлодіодів та кнопки.
2. Підключити потрібні канали АЦП за допомогою перемичок на платі.
3. Повторити принципи роботи з рідкокристалічними індикаторами та кнопками. Навчитись налаштовувати вбудований АЦП на різні режими роботи та обробляти виміряні дані.
4. Створити свою власну програму, яка забезпечує функції, описані в індивідуальному завданні, та скомпілювати її.
5. Запрограмувати МК, перевірити правильність виконання.

Індивідуальні завдання

1. Виміряні значення аналогових сигналів з каналів PA2 і PA3 виводяться в центрі РК-екрану. Рівень вхідного сигналу каналу PA2 також відображається на одному з світлодіодних рядків. Кількість лінійно засвічених світлодіодів повинна бути пропорційною до рівня вхідного сигналу.
2. Виміряні значення аналогових сигналів з каналів PA2 і PA3 виводяться в центрі РК-екрану. Рівень вхідного сигналу каналу PA2 або каналу PA3 також відображається на світлодіодному рядку. Передбачити в програмі 1 кнопку, яка б визначала який з каналів відображатиметься на світлодіодному рядку в поточний час. Перемикання каналу відбувається після відпускання

кнопки. При цьому в другому рядку повинна відображатись назва вибраного каналу.

3. Виміряні значення аналогових сигналів з каналів PA2 і PA3 виводяться в центрі РК-екрану. Рівень вхідного сигналу каналу PA3 також відображається на нижньому рядку РК-екрану у вигляді рядка з зафарбованих прямокутників. Кількість лінійно зафарбованих прямокутників повинна бути пропорційною до рівня вхідного сигналу.

4. Виміряні значення аналогових сигналів з каналів PA2 і PA3 виводяться на початку кожного з рядків РК-екрану відповідно. Навпроти кожного з цифрових значень виводиться лінійна шкала відповідного рівня сигналу з чорних прямокутників розмірністю 10. Кількість лінійно зафарбованих прямокутників повинна бути пропорційною до рівня вхідного сигналу.

5. Виміряні значення аналогових сигналів з каналів PA2 і PA3 виводяться в центрі РК-екрану. Рівень вхідного сигналу каналу PA2 або каналу PA3 також відображається на нижньому рядку РК-екрану у вигляді рядка з зафарбованих прямокутників розмірністю 10. Кількість лінійно зафарбованих прямокутників повинна бути пропорційною до рівня вхідного сигналу. Передбачити в програмі 1 кнопку, яка б визначала який з каналів відображатиметься на екрані в поточний момент. Перемикання зканалів відбувається після відпускання кнопки. При цьому на початку другого рядка повинна відображатись назва вибраного каналу.

Контрольні запитання

1. Вказати основні існуючі типи аналого-цифрових перетворювачів та охарактеризувати їх.
2. Характеристики та можливості аналого-цифрового перетворювача в мікроконтролерах AVR
3. Назвати регісти для роботи з АЦП та вказати їх призначення.
4. Що таке опорна напруга та як вона задається для роботи з АЦП?
5. Скільки каналів АЦП існує в МК AVR та як відбувається вимірювання на каналах? Чи можливе одночасне вимірювання на всіх каналах? Якщо ні – то як це подолати?
6. Як програмно задається канал вимірювання аналогового сигналу?
7. Як задається точність вимірювання АЦП?
8. В яких одиницях відбувається вимірювання аналогового сигналу?
9. Які межі вимірювання аналогового сигналу? Як розширити можливі межі вимірювання? Привести приклад схемного рішення.

Зміст звіту

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

РОЗДІЛ 3

ДОДАТОК

ДОВІДКОВІ ДАНІ ДЛЯ РОБОТИ З СИМВОЛЬНИМИ РІДКОКРИСТАЛІЧНИМИ ДИСПЛЕЯМИ

Таблиця 3.1

Розшифровка найбільш використовуваних команд для LCD

Команда LCD	HEX - код	Виконувані дії	Час виконання., мкс
Очистка дисплею	0x01	Порожній екран, очистка пам'яті, курсор в лівій верхній позиції	1640
Повернення курсору на початок	0x02	Курсор в лівій верхній позиції, пам'ять не очищається	1640
Здвиг курсору вліво	0x04	Після виводу чергового символу курсор автоматично здвигається на одне знакомісце вліво	40
Здвиг курсору вправо	0x06	Після виводу чергового символу курсор автоматично здвигається на одне знакомісце вправо	40
Вимкнення дисплею	0x08	Повна відсутність зображення на екрані LCD	40
Вимкнення курсору	0x0C	Дозволено виведення зображення, проте курсор не видно	40
Прямокутна форма курсору	0x0D	Дозволено виведення зображення, курсор у вигляді блимаю чого чорного прямокутника	40
Лінійна форма курсору	0x0E	Дозволено виведення зображення, курсор у вигляді нижньої підрядкової немигаючої лінії	40
Комплексна форма курсору	0x0F	Дозволено виведення зображення, курсор у вигляді блимаю чого чорного прямокутника з підкресленням	40
Інтерфейс 4 біта, 1 рядок	0x20	Зв'язок з однорядковим LCD через 4 лінії шини даних	40
Інтерфейс 4 біта, 2 рядки	0x28	Зв'язок з дворядковим LCD через 4 лінії шини даних	40
Інтерфейс 8 біт, 1 рядок	0x30	Зв'язок з однорядковим LCD через 8 ліній шини даних	40
Інтерфейс 8 біт, 2 рядки	0x38	Зв'язок з дворядковим LCD через 8 ліній шини даних	40
Доступ до ОЗП знакогенератора	0x40 - 0x7F	Запис даних по цим адресам дозволяє створити 16 власних символів	40
Установка позиції курсору	0x80 - 0xCF	Курсор встановлюється в позицію згідно рис. 2	40

Верхній рядок LCD

0x80	0x81	0x82	0x83	0x84	0x85	0x86	0x87	0x88	0x89	0x8A	0x8B	0x8C	0x8D	0x8E	0x8F
0xC0	0xC1	0xC2	0xC3	0xC4	0xC5	0xC6	0xC7	0xC8	0xC9	0xCA	0xCB	0xCC	0xCD	0xCE	0xCF

Нижній рядок LCD

Рис. 3.1. Розподіл адрес на рядках екрану

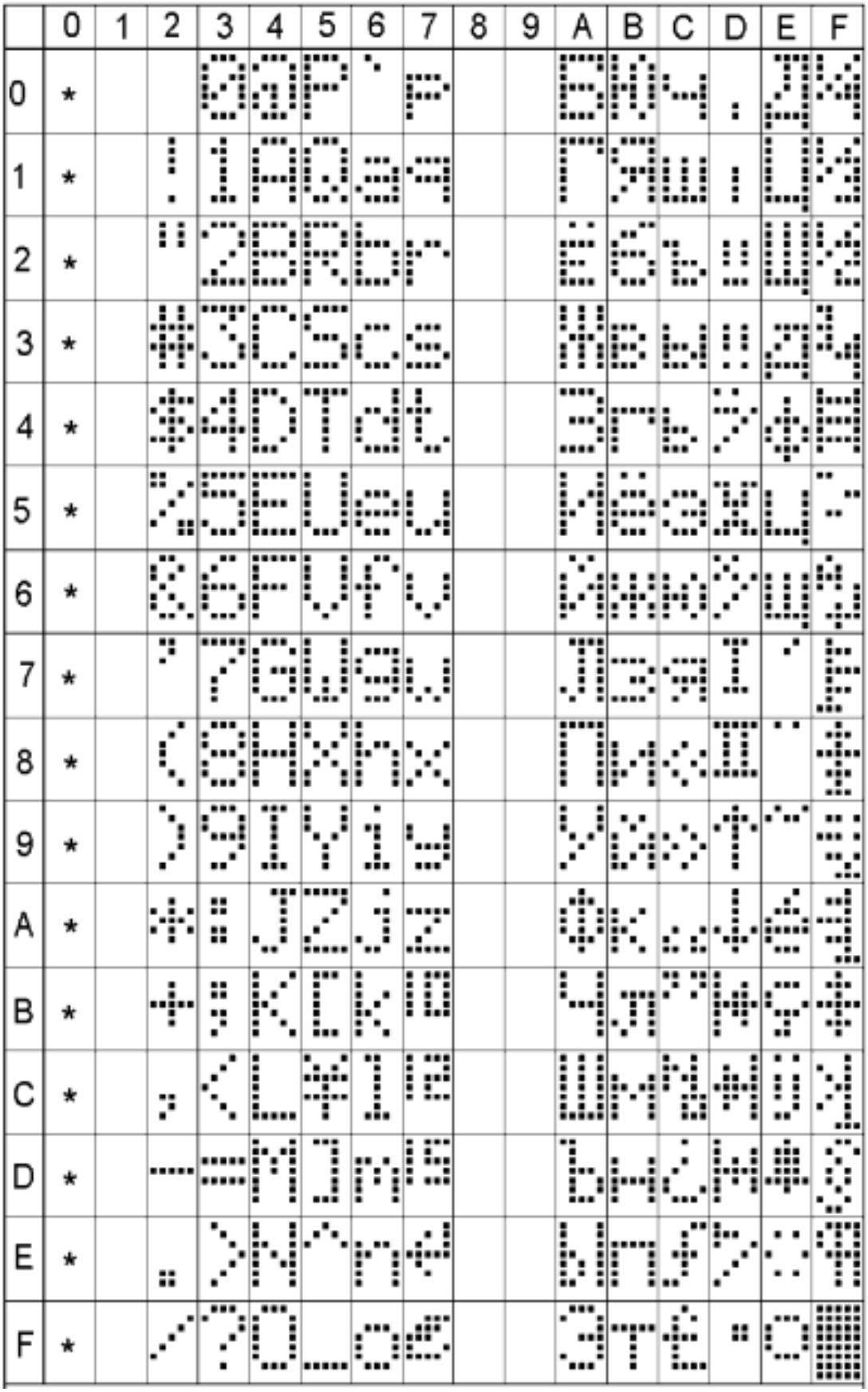


Рис. 3.2. Відповідність символів знакогенератора до адрес в пам’яті ПЗП
ПРИКЛАД: код 0x34 відповідає цифрі “ 4 “, код 0xB9 – букві “ й ”

FUSE-БІТИ В МІКРОКОНТРОЛЕРАХ AVR

У таблицях 3.2, 3.3 приведені fuse-біти часто використовуваних мікроконтролерів AVR. Зліва назви fuse-битов по даташиту, в перших двох рядках перераховані сімейства і типи конкретних МК, а на перетині рядків і стовпців стоїть знак плюс, якщо даний fuse-біт міститься в даному МК, або вказано назву, відмінну від стандартного. Якщо якийсь біт відсутній – у відповідній клітині нічого немає.

Таблиця 3.2

FUSE-біти в мікроконтролерах AVR сімейства ATtiny

	2313	25/45/85	13	26	261/461/861
RESERVED					
OCDEN					
JTAGEN					
SELFPRGEN	+	+	+		+
DWEN	+	+	+		+
EESAVE	+	+	+	+	+
SPIEN	+	+	+	+	+
WDTON	+	+	+		+
BODLEVEL2	+	+			+
BODLEVEL1	+	+	+		+
BODLEVEL0	+	+	+	BODLEVEL	+
BODEN				+	
RSTDISBL	+	+	+	+	+
CKDIV8	+	+	+		+
CKOUT	+	+			+
SUT1	+	+	+	+	+
SUT0	+	+	+	+	+
CKOPT				+	
CKSEL3	+	+		+	+
CKSEL2	+	+		+	+
CKSEL1	+	+	+	+	+
CKSEL0	+	+	+	+	+
PLLCK				+	
BOOTRST					
BOOTSZ1					
BOOTSZ0					

RESERVED – цей біт зарезервований для службових цілей фірмою Atmel. Ні за яких умов не рекомендується міняти його стан (тобто треба залишати його таким, як він встановлений при виготовленні МК). У цьому рядку зустрічаються біти з іншими назвами, як правило, це біти включення режиму сумісності з застарілими типами МК,

на зміну яким випущені нові. Зазвичай в кінці назви такого fuse-біта міститься символ C – від COMPATIBLE (сумісний).

OCDEN – fuse дозволяє роботу схеми внутрішнього відладчика (On Chip Debug ENable). Не залишайте встановленим цей біт в комерційних продуктах! Інакше вашу програму можна буде зчитати з пам'яті МК.

JTAGEN – fuse біт дозволяє роботу інтерфейсу програмування-налагодження JTAG. У порівнянні з SPI-інтерфейсом, JTAG володіє розширеними можливостями. Не рекомендується без необхідності залишати цей біт встановленим, тому що в цьому випадку споживаний МК струм зростає.

Таблиця 3.3

FUSE-біти в мікроконтролерах AVR сімейства ATmega

	8	16	48 /88 /168	128	169	329	8515	8535
RESERVED				M103C	+		S8515C	S8535C
OCDEN		+		+	+	+		
JTAGEN		+		+	+	+		
SELFPRGEN			+					
DWEN		+	+					
EESAVE	+	+	+	+	+	+	+	+
SPIEN	+	+	+	+	+	+	+	+
WDTON	+		+	+	+	+	+	+
BODLEVEL2			+		+			
BODLEVEL1			+		+	+		
BODLEVEL0	BODLEVEL	BODLEVEL	+	BODLEVEL	+	+	BODLEVEL	BODLEVEL
BODEN	+	+		+			+	+
RSTDISBL	+		+			+		
CKDIV8			+		+	+		
CKOUT			+		+	+		
SUT1	+	+	+	+	+	+	+	+
SUT0	+	+	+	+	+	+	+	+
CKOPT	+	+		+			+	+
CKSEL3	+	+	+	+	+	+	+	+
CKSEL2	+	+	+	+	+	+	+	+
CKSEL1	+	+	+	+	+	+	+	+
CKSEL0	+	+	+	+	+	+	+	+
PLLCK								
BOOTRST	+	+	+	+	+	+	+	+
BOOTSZ1	+	+	+	+	+	+	+	+
BOOTSZ0	+	+	+	+	+	+	+	+

SELFPRGEN – біт, що дозволяє програмі МК проводити запис в пам'ять програм, тобто проводити самопрограмування.

DWEN – fuse біт, що дозволяє роботу DebugWire – це інтерфейс налагодження по одному дроту. Не рекомендується залишати його встановленим в комерційних виробах.

EESAVE – fuse біт, після установки якого при стиранні пам'яті МК вміст EEPROM даних буде зберігатися недоторканим, тобто не буде стерто.

SPIEN – fuse біт, що дозволяє роботу інтерфейсу внутрішньосхемного програмування МК по SPI. Цей біт може бути легко переустановлений за допомогою паралельного програматора (або JTAG, якщо такий дозволений і мається на МК). Всі МК випускаються з встановленим бітом SPIEN, зняти його по інтерфейсу SPI неможливо.

WDTON – fuse біт, після установки якого сторожовий таймер WDT включається відразу після подачі живлення і не може бути відключений програмно. Якщо біт не встановлений, то включенням і відключенням WDT можна управляти програмно.

BODLEVEL – група fuse бітів. Може бути або один такий біт, або декілька, тоді вони нумеруються, починаючи з нуля. Значення цих fuse бітів визначає поріг спрацьовування схеми BOD – детектора рівня живлячої напруги, при зниженні напруги живлення нижче цього рівня відбудеться "скидання" МК.

BODEN – fuse біт, що включає схему апаратного детектора неприпустимого рівня живлячої напруги, тобто схему BOD.

RSTDISBL – fuse біт, що відключає сигнал зовнішнього скидання від виводу мікроконтролера і підключає до нього схему порту вводу-виводу. Цей біт є тільки в тих МК, у яких контакт апаратного скидання RESET суміщений з одним з портів введення-виведення. Помилкова установка цього fuse біта може відключити RESET і ви не зможете більше прошивати по ISP. Не встановлюйте цей біт, якщо маєте намір продовжувати працювати з МК за допомогою послідовних програматорів. "Оживити" МК з встановленим RSTDISBL можна тільки паралельним програматором і не для всіх МК.

CKDIV8 – fuse біт, що включає попереднє розподіл частоти кварцового (або іншого наявного) тактового генератора на 8. Тобто при включеному цьому біті і застосуванні кварцового резонатора на 8 МГц реальна тактова частота МК складе 1 МГц.

CKOUT – fuse біт, що дозволяє виведення тактової частоти на один з виходів МК (для тактування інших пристроїв).

SUT1 і **SUT0** – fuse біти, керуючі режимом запуску тактових генераторів МК. Пов'язані з нижчеописаними бітами, що визначають тип і частоту тактового генератора. При помилковому їх налаштуванні можливі ситуації нестійкого запуску генератора або неодноразового скидання МК в процесі подачі на нього живлення.

CKOPT – біт, що визначає режим роботи вбудованого генератора тактової частоти для роботи з кварцовими резонаторами. Реально змінює коефіцієнт підсилення вбудованого інвертора в схемі генератора і вихідну напругу на ніжці XTAL2. Помилкова установка може призводити до нестійкого запуску кварцевого генератора, аж до збудження його не на тій гармоніці, що потрібна (через цей біт кварц може запускатись або тільки при живленні МК напругою не вище 3,6 В, або тільки після дотику до виходу XTAL1 металевим предметом)

CKSEL0 ... CKSEL3 – група бітів fuse-бітів, комбінація яких визначає тип і частоту працюючого тактового генератора. Всього можливо до 16 комбінацій, однак не всі визначені для всіх типів МК. Помилкова установка комбінації цих бітів може зробити МК недієздатним – він не буде працювати в схемі без подачі тактового сигналу на ніжку XTAL1.

PLLCK – fuse біт, дозволяє використання вбудованого синтезатора частоти для тактування ядра МК.

BOOTRST – fuse біт, що визначає адресу, з якої буде розпочато виконання програми після скидання – якщо біт установлено, то початок програми буде не з адреси 0000h (як звичайно), а з адреси області завантажувача (Boot Loader).

BOOTSZ – два fuse біта, що визначають розмір області пам'яті програм, виділеної для завантажувача (Boot Loader). Комбінація цих бітів, зокрема, визначає точку початку виконання програми після скидання, якщо встановлений біт BOOTRST.

ЛІТЕРАТУРА

1. Рюмік С. М. Мікроконтролери. Крок 1-10 //журнал Радіоаматор № 3-12/2004, с. 35–39.

2. Рюмік С. М. Мікроконтролери AVR. Ступінь 1-10 //Радіоаматор №1-7, 9-11/2005, с. 35–39.

3. Евстифеев А. В. Микроконтроллеры AVR семейств Tiny и Mega фирмы “Atmel” — М.: Издательский дом “Додэка-XXI”, 2004. – 560 с.

4. Баранов В. Н. Применение микроконтроллеров AVR: схемы, алгоритмы, программы.—М.: Издательский дом “Додэка-XXI”, 2004. – 288 с.