Житомирська політехніка	Міністерство освіти і науки України Державний університет «Житомирська політехніка

Затверджено науково-методичною радою ЖДТУ
протокол від «__»_____ 20__ р. №__

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ

для проведення лабораторних робіт
з навчальної дисципліни

«ПРОГРАМУВАННЯ МІКРОПРОЦЕСОРНИХ ЗАСОБІВ ВИМІРЮВАЛЬНОЇ ТЕХНІКИ»

для студентів освітнього рівня «бакалавр»
спеціальності 152 «Метрологія та інформаційно-вимірювальна техніка»
освітньо-професійна програма «Комп'ютеризовані інформаційно-вимірювальні системи»
факультет комп'ютерно-інтегрованих технологій, мехатроніки і робототехніки
кафедра метрології та інформаційно-вимірювальної техніки

Розглянуто і рекомендовано
на засіданні кафедри метрології
та інформаційно-вимірювальної
техніки

протокол від 17 червня 2020р., № 3

Розробники: к.т.н., доц. кафедри метрології
та інформаційно-вимірювальної техніки Чепюк Л.О.
асистент кафедри метрології
та інформаційно-вимірювальної техніки Лугових О.О.

Житомир
2020 н.р.

Лабораторна робота № 1

ЗВЕРТАННЯ ДО EEPROM ПРИ ЧИТАННІ/ЗАПИСУ

Мета роботи: вивчення організації EEPROM мікроконтролера AT90S2313, отримання навиків запису та читання EEPROM мікроконтролерів AVR

Теоретичні відомості

Постійний запам'ятовуючий пристрій EEPROM призначений для збереження даних, записаних при програмуванні мікроконтролера й одержуваних у процесі виконання програми.

EEPROM має відособлений адресний простір. При звертанні до EEPROM адреса записується в регістр адреси EEAR ($1E). Байт, призначений для запису, заноситься в регістр даних EEDR ($1D). Байт, одержуваний при читанні, надходить у цей самий регістр. Для керування процедурами запису і читання використовується регістр керування EECR ($1C).

Таблиця 6.1 – Регістр керування EEPROM

Біти	7	6	5	4	3	2	1	0	
$1C ($3C)	-	-	**EEPM1**	**EEPM0**	**EERIE**	**EEMPE**	**EEPE**	**EERE**	**EECR**
Читання/Запис	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Початковий стан	0	0	X	X	0	0	X	0	

Біти 5,4 - EEPM1 і EEPM0: Розряди вибору режиму EEPROM.

Біти установки режиму програмування EEPROM визначають, яким способом виконуватиметься команда програмування, якщо скинутий прапор EEPE. Після системного скидання розряди EEPMn будуть встановлені в 0b00 у тому випадку, якщо в цей час не відбувається процес програмування EEPROM. Будь-які значення EEPMn ігноруються, коли установлений прапор EEPE.

Таблиця 6.2 – Біти вибору режиму EEPROM

EEPM1	EEPM0	Час програмування	Операція
0	0	3,4 мс	Стирання і запис за одну операцію (атомарна дія)
0	1	1,8 мс	Тільки стирання
1	0	1,8 мс	Тільки запис
1	1	-	Зарезервовано

Біт 3 - EERIE: Дозвіл переривання від EEPROM. Якщо цей розряд встановлений в одиницю, переривання дозволені (якщо прапор I регістра SREG також установлений в одиницю).

Біт 2 - EEMPE: Управління дозволом програмування EEPROM. Значення біта EEMPE визначає функціонування прапора EEPE. Якщо біт EEMPE встановлений (рівний 1), установка біта EEPE в одиницю викликає програмування EEPROM за вибраною адресою. Впродовж чотирьох машинних циклів відразу після установки EEMPE треба робити установку EEPE. Інакше біт EEMPE буде апаратно скинутий, а програмування виявиться неможливим.

Біт 1 - EEPE: Дозвіл програмування EEPROM. Цей біт управляє процесом програмування EEPROM. Установка біта EEPE в одиницю викликає один з варіантів програмування EEPROM у відповідності зі значеннями бітів EEPMn. Перед тим, як записувати в EEPE одиницю, необхідно раніше встановити в одиницю біт EEMPE. Інакше процес програмування EEPROM не почнеться.

Після закінчення процесу програмування біт EEPE автоматично скидається в нуль. Відразу після установки EEPE в одиницю робота CPU призупиняється на два машинних цикли.

Біт 0 - EERE: Дозвіл читання EEPROM

Для запису байта в EEPROM необхідно:

- записати адресе в регістр адреси;
- записати байт у регістр даних;

- установити в одиничний стан розряд EEMPE регістра EECR,
- при EEMPE = 1 встановити в одиничний стан розряд EEPE регістра EECR.

Процедура запису виконується в залежності від величини напруги живлення за 2,5-4 мс. При завершенні запису розряд EEPE регістра EECR апаратно скидається в нульовий стан.

Розряд EEMPE зберігає одиничний стан протягом 4-х тактів після установки і апаратно скидається в нульовий стан.

Для читання байта з EEPROM необхідно:

- записати адреса в регістр адреси;
- установити в одиничний стан розряд EERE регістра EECR. Прочитаний байт надходить у регістр даних. Розряд EERE регістра EECR апаратно скидається в нульовий стан.

```
;* This Application note shows how to read data from and write data to the
;* EEPROM. Both random access and sequential access routines are listed.
;*
;***************************************************************************
.include "2313def.inc"
       rjmp    RESET           ;Reset Handle
;***************************************************************************
;* EEWrite
;* This subroutine waits until the EEPROM is ready to be programmed, then
;* programs the EEPROM with register variable "EEdwr" at address "EEawr"
;* Ця підпрограма очикування, доки EEPROM не буде готова до програмування
;***************************************************************************
;***** Subroutine register variables

.def    EEdwr   =r16            ;data byte to write to EEPROM
.def    EEawr   =r17            ;address to write to

;***** Code
EEWrite:
       sbic    EECR,EEWE       ;if EEWE not clear
       rjmp    EEWrite         ;    wait more
       out     EEAR,EEawr      ;output address
       out     EEDR,EEdwr      ;output data
       sbi     EECR,EEWE       ;set EEPROM Write strobe
                               ;This instruction takes 4 clock cycles since
                               ;it halts the CPU for two clock cycles
       ret
;***************************************************************************
;* EERead
;*
;* This subroutine waits until the EEPROM is ready to be programmed, then
;* reads the register variable "EEdrd" from address "EEard"
;***************************************************************************
;***** Subroutine register variables
```

```
.def    EEdrd   =r0            ;result data byte
.def    EEard   =r16           ;address to read from

;***** Code


EERead:
        sbic    EECR,EEWE       ;if EEWE not clear
        rjmp    EERead          ;    wait more
        out     EEDR,EEard      ;output address
        sbi     EECR,EERE       ;set EEPROM Read strobe
                                ;This instruction takes 4 clock cycles since
                                ;it halts the CPU for two clock cycles
        sbi     EECR,EERE       ;set EEPROM Read strobe 2nd time
                                ;This instruction takes 4 clock cycles since
                                ;it halts the CPU for two clock cycles
        in      EEdrd,EEDR      ;get data
        ret

;****************************************************************************
;*
;* EEWrite_seq
;*
;* This subroutine increments the EEPROM address by one and waits until the
;* EEPROM is ready for programming. It then programs the EEPROM with
;* register variable "EEdwr_s".
;*
;****************************************************************************
;***** Subroutine register variables

.def    EEwtmp  =r0            ;temporary storage of address
.def    EEdwr_s =r16           ;data to write

;***** Code

EEWrite_seq:
        sbic    EECR,EEWE       ;if EEWE not clear
        rjmp    EEWrite_seq     ;    wait more
        in      EEwtmp,EEAR     ;get address
        inc     EEwtmp          ;increment address
        out     EEAR,EEwtmp     ;output address
        out     EEDR,EEdwr_s    ;output data
        sbi     EECR,EEWE       ;set EEPROM Write strobe
                                ;This instruction takes 4 clock cycles since
                                ;it halts the CPU for two clock cycles
        ret

;****************************************************************************
;* EERead_seq
;*
;* This subroutine increments the address stored in EEAR and reads the
;* EEPROM into the register variable "EEdrd_s".
;*
;****************************************************************************

;***** Subroutine register variables

.def    EErtmp  =r0            ;temporary storage of address
```

```
.def    EEdrd_s =r1             ;result data byte

;***** Code

EERead_seq:
        in      EErtmp,EEAR     ;get address
        inc     EErtmp          ;increment address
        out     EEAR,EErtmp     ;output address
        sbi     EECR,EERE       ;set EEPROM Read strobe
                                ;This instruction takes 4 clock cycles since
                                ;it halts the CPU for two clock cycles
        sbi     EECR,EERE       ;set EEPROM Read strobe 2nd time
                                ;This instruction takes 4 clock cycles since
                                ;it halts the CPU for two clock cycles
        in      EEdrd_s,EEDR    ;get data
        ret

;************************************************************************
;*
;* Test/Example Program
;*
;************************************************************************
;***** Main Program Register variables

.def    counter =r18
.def    temp    =r19

;***** Code

RESET:
;***** Program a random location
        ldi     EEdwr,$aa
        ldi     EEawr,$10
        rcall   EEWrite         ;store $aa in EEPROM location $10
;***** Read from a random locations
        ldi     EEard,$10
        rcall   EERead          ;read address $10
        out     PORTB,EEdrd     ;output value to Port B

;***** Fill the EEPROM with bit pattern $55,$aa,$55,$aa,...
        ldi     counter,64      ;init loop counter
        ser     temp
        out     EEAR,temp       ;EEAR <- $ff (start address - 1)

loop1:  ldi     EEdwr_s,$55
        rcall   EEWrite_seq     ;program EEPROM with $55
        ldi     EEdwr_s,$aa
        rcall   EEWrite_seq     ;program EEPROM with $aa
        dec     counter         ;decrement counter
        brne    loop1           ;and loop more if not done

;***** Copy 10 first bytes of EEPROM to r2-r11

        ldi     temp,$ff
        out     EEAR,temp       ;EEAR <- $ff (start address - 1)

        ldi     ZL,2            ;Z-pointer points to r2

loop2:  rcall   EERead_seq      ;get EEPROM data
```

```
        st      Z,EEdrd_s       ;store to SRAM
        inc     ZL
        cpi     ZL,12           ;reached the end?
        brne    loop2           ;if not, loop more

forever:rjmp    forever         ;eternal loop
```

Завдання до лабораторної роботи

У комірки EEPROM, починаючи з ADR1, записати N чисел. Скопіювати N-2 чисел до регистрів, починаючи з R2. Перевірити виконання програми в AVR STUDIO.

При написанні програми необхідно замість EECR можна підставляти $1C, EEAR - $1E, EEDR - $1D, EEWE – 1, EERE – 0. Результати виконання програми в AVR STUDIO знаходяться в Memory –I/O.

Таблиця 6.3 – Варіанти завдань

Номер	ADR1	N	Коди чисел
01	$10	E	6F
02	$12	C	63
03	$14	D	75
04	$16	B	86
05	$18	A	68
06	$1A	9	6D
07	$1C	8	6F
08	$1E	7	7A
09	$20	6	7C
10	$22	5	74
11	$24	4	73
12	$26	E	78
13	$28	C	8C
14	$2A	D	6C
15	$2C	B	6E
16	$2E	A	8A
17	$30	9	6A
18	$32	8	6B
19	$34	7	9B
20	$36	6	7B

21	$38	5	69
22	$3A	4	87
23	$3C	E	85
24	$3E	C	83
25	$0A	D	AD
26	$0C	B	A5
27	$0E	A	AE

Зміст звіту

Завдання до лабораторної роботи.

Алгоритм програми з поясненням.

Текст програм з поясненням.

Результати виконання програми в AVR STUDIO.

Контрольні запитання

1. Дайте характеристику основних станів регістра адреси EEPROM Address Register (EEAR).
2. Дайте характеристику основних станів регістра даних EEPROM Data Register (EEDR).
3. Дайте характеристику основних станів регістра керування EEPROM Control Register (EECR).
4. Наведіть порядок дій для запису числа в EEPROM.
5. Наведіть порядок дій для читання числа з EEPROM.
6. Наведіть порядок дій для читання числа з FLASHROM.

Лабораторна робота № 2

СХЕМИ ВІДОБРАЖЕННЯ ІНФОРМАЦІЇ

Мета роботи: Отримання навичок виведення інформації на статичні індикатори. Розробка програм для мікроконтролера AT90S2313 для відображення цифрової інформації на індикаторах статичного типу.

Теоретичні відомості

1 Системи відображення інформації

Практично кожен мікропроцесорний пристрій містить елементи індикації. Як індикатори в даний час найчастіше застосовуються світлодіоди. На ринку є величезний вибір світлодіодів найрізноманітніших видів і розмірів.

У МПП світлові індикатори можуть служити для відображення різних режимів роботи: попередження про критичні ситуації, відображення ходу прийому сигналів керування тощо. Підключити одиночний світлодіодний індикатор до МК дуже просто. На рисунку 1 наведена схема підключення світлодіода безпосередньо до виводу порту МК.

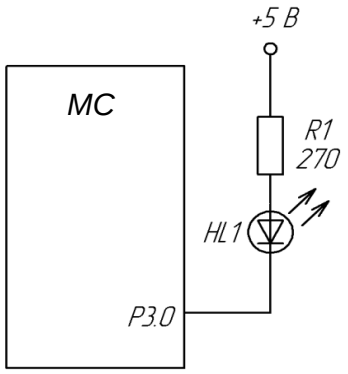

Рисунок 1 – Схема підключення світлодіодного індикатора

Усі вихідні каскади МК мають достатню навантажувальну здатність

для того, щоб витримати підключення одного світлодіодного індикатора зі споживаним струмом у робочому режимі не більше 20 мА.

Для керування двома світлодіодами одним виходом у МК передбачено активні вихідні каскади, і для перемикання режиму роботи (введення або виведення) слугує спеціальний регістр. Таким чином, сигнал кожного виходу будь-якого порту може мати 3 значення – "0", "1" і високоімпедансний ("Z") стан. Це дозволяє керувати двома світлодіодами за допомогою одного виводу (рис. 2).

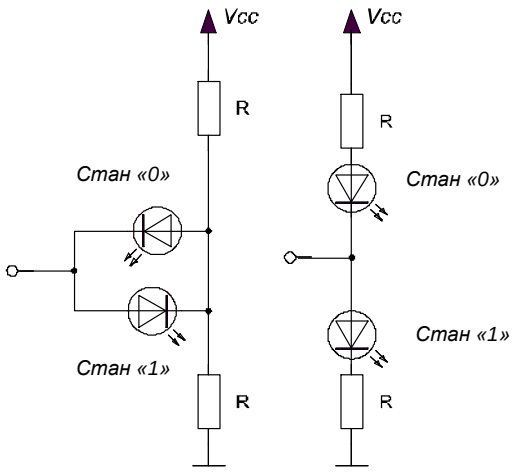

Рисунок 2 – Керування двома світлодіодами одним виходом МК

При роботі порту в режимі виходу, залежно від стану "0" або "1" горить відповідно верхній або нижній світлодіод. При перемиканні в Z-стан, і при відповідному виборі резисторів струм через світлодіоди дуже малий і їх світіння майже непомітно.

Цікавим є включення трьох пар зустрічно паралельних світлодіодів за схемою "зірка" (рис. 3).

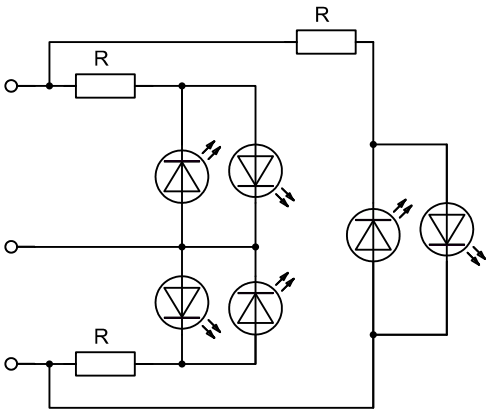

Рисунок 3 – Включення трьох пар зустрічно паралельних світлодіодів

Відповідними сигналами МК можна домогтись світіння будь-якого з шести світлодіодів, а також їх комбінацій. Використовуючи динамічну індикацію, можна отримати будь-які комбінації.

2 Паралельні порти введення-виведення

Всі порти введення/виведення мікроконтролера в режимі цифрового вводу-виводу являють собою двонаправлені порти, в яких кожен із виводів може працювати як на ввід, так і на вивід інформації.

На рисунку 4 зображено еквівалентну схему вхідних кіл розряду порту введення/виведення. Всі виводи порту мають індивідуальні резистори навантаження, вхідні схеми кожної лінії мають по два захисних діоди.

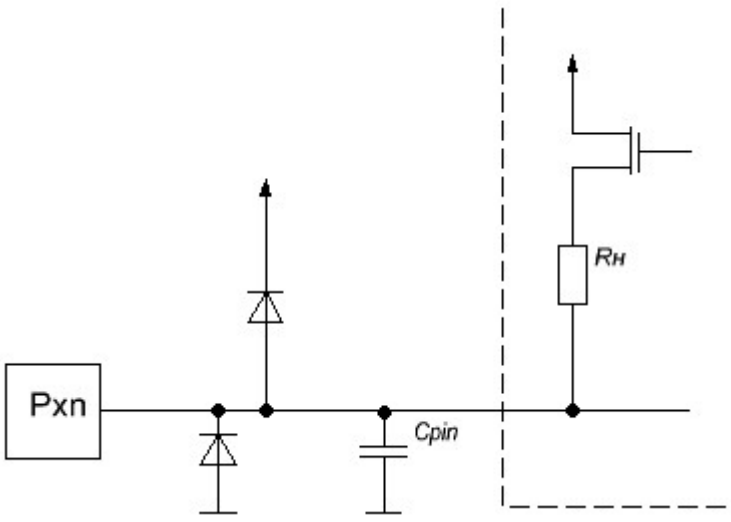

Рисунок 4 – Еквівалентна схема вхідних кіл розряду порту введення/виведення

Для кожного порту вводу-виводу в МК існує три спеціальних регістри:

– PORTx – регістр даних;
– DDRx – регістр керування;
– PINx – регістр безпосереднього читання стану лінії порту,
де «x» – назва порту.

Кожен розряд порту зв'язаний з трьома розрядами спеціальних регістрів:

– DDxn – розряд номеру n регістра DDRx;
– PORTxn – розряд номеру n регістра PORTx;
– PINxn – розряд номеру n регістра PINx.

Біт DDRxn регістру DDRx обирає напрямок передачі інформації. При DDxn = «1», розряд Pxn працює як вихід, при DDxn = «0» – як вхід.

Якщо розряд порту настроєно на вхід, то встановлення біту PORTxn в «1» підключить внутрішній навантажувальний резистор. Для його відключення необхідно в PORTxn записати «0». Одразу після системного скидання всі виводи всіх портів переходять в третій стан.

Якщо розряд порту настроєно на вихід, то встановлення біту PORTxn в «1» викличе появу «1» на виході порту. Якщо в розряд PORTxn записано «0», то і на виході буде «0».

PortD	$12	PortB	$18	PortA	$1B
DDRD	$11	DDRB	$17	DDRA	$1A
PinD	$10	PinB	$16	PinA	$19

Завдання до лабораторної роботи

Написати програму для мікроконтролера AT90S2313, яка:

a) виконує пересилку до регістру R числа з SRAM і відображає його на статичному світлодіодному індикаторі, який підключений до порта B мікроконтролера;
b) відображає на індикаторі по черзі молодшу та старшу частину числа з регістру R на індикаторі (схему розробити самостійно).

Перевірити виконання програми в AVR STUDIO. Зібрати схему лабораторного макету у середовищі Proteus. Перевірити працездатність розробленої програми.

Таблиця 1 – Таблиця варіантів завдань

Номер	*SRAM*	*CONST1*	*R*
01	71	FE	0
02	62	FF	1
03	83	FD	2
04	94	FC	3
05	A5	FB	4
06	A6	FA	5
07	C7	F1	6
08	78	F2	7
09	69	F3	3
10	6A	F4	0
11	7B	F5	1
12	8C	F6	2
13	9D	F7	3
14	AE	F8	4
15	7F	F8	5
16	6E	F9	6

17	8A	F1	7
18	8D	F2	0
19	9C	F3	1
20	D6	F4	2
21	D5	FF	3
22	67	1F	4
23	CF	2E	5
24	6D	3D	6
25	8A	4C	7
26	9E	5B	1
27	7F	6A	2

Зміст звіту

1. Схема лабораторного макета (схему розробити самостійно).
2. Завдання до лабораторної роботи.
3. Текст програми з поясненнями.
4. Результати виконання програми в AVR STUDIO.
5. Результати роботи лабораторного макета в середовищі Proteus.

Приклад виконання завдання

Написати програму для мікроконтролера AT90S2313, яка:

a) виконує пересилку до регістру R2 числа $6A з комірки SRAM з адресою $7F і відображає його на статичному світлодіодному індикаторі, який підключений до порта B мікроконтролера;
b) відображає на індикаторі по черзі молодшу та старшу частину числа $6A Перевірити виконання програми в AVR STUDIO. Зібрати схему лабораторного макету у середовищі Proteus. Перевірити працездатність розробленої програми.

Виконання першої частини завдання

```
.def temp =R16 ; присвоїти temp значення R16
clr R27;         очистити старший байт X
LDI R26,$7F;     встановити $7F у молодший байт X
LD R1,X;         завантажити в R1 зміст SRAM за адресою 7F
```

```
LDI temp,$FF;     ініціалізація порта B
OUT $17,temp;     порт B настроєний на вивід
M1: OUT $18,R1;   вивід через порт B змісту R1 (загорання
                ;  світлодіодів)
RJMP M1
```

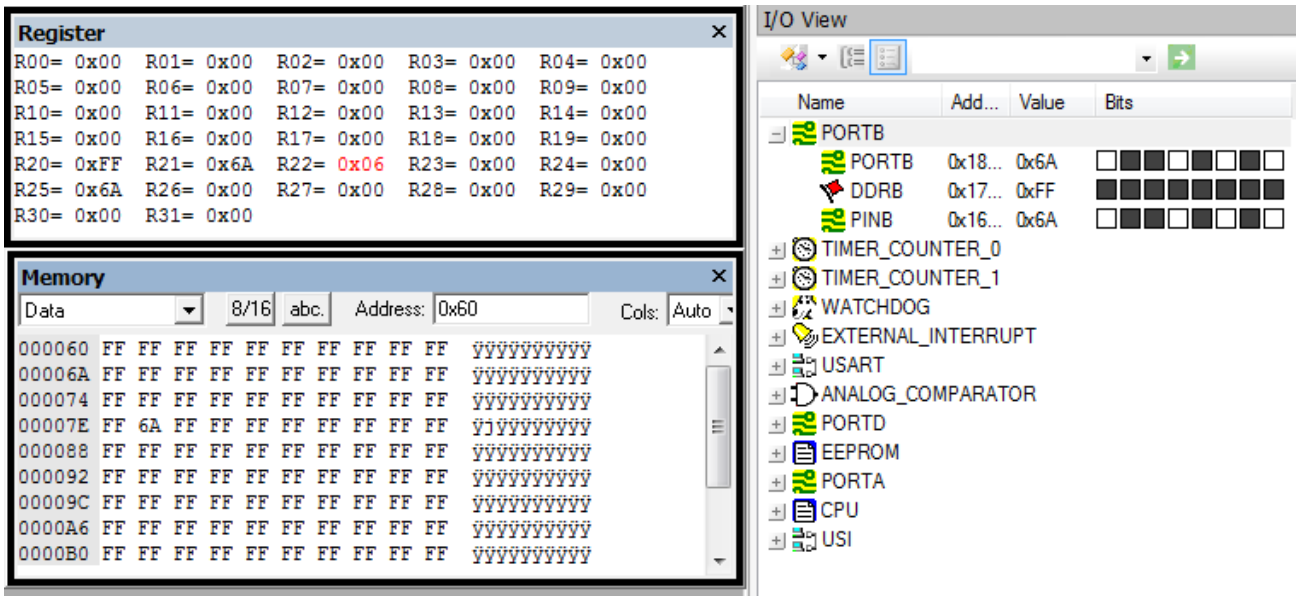

Рисунок 5 – Результат виконання першої частини завдання в AVR Studio

Програма виконання другої частини завдання.

```
.include "2313def.inc"
.def temp =R16              ;присвоїти temp значення R16
.def fine =R18          ;мітка затримки
.def medium =R19            ;мітка затримки
.def coarse =R20            ;мітка затримки
   LDI R17, RamEND ;налаштування стеку
       OUT SPL, R17
       LDI R17, $6A  ;запис в комірку SRAM $7F
       STS $7F, R17  ;числа $6A
       clr R27       ;очистити старший байт X
   LDI R26,$7F              ;встановити $7F у молодший байт X
  LD R1,X           ;завантажити в R1 зміст SRAM за адресою $7F
  LDI temp,$FF              ;ініціалізація порта D, B
  OUT $11,temp              ;порт D налаштований на вивід
```

```
   OUT $17,temp              ;порт B налаштований на вивід
L1: MOV R17,R1               ;копіювання інформації в R17 з R1
   ANDI R17,$0F              ;маскування старшого байту числа
   OUT $18,R17               ;запис в порт B молодшої частини R17 (загорання ;світлодіодів)
   OUT $12, R17              ;запис в порт D молодшої частини R17 (на індикатор)
   RCALL delay               ;виклик підпрограми затримки
   MOV R17,R1                ;копіювання інформації в R17 з R1
   ANDI R17,$F0              ;маскування молодшого байту числа
   OUT $18,R17               ;запис в порт B старшої частини R17 (загорання ;світлодіодів)
    LSR R17              ;зсув усіх біт праворуч (для виводу на індикатор)
        LSR R17                ;зсув усіх біт праворуч
        LSR R17                ;зсув усіх біт праворуч
        LSR R17                ;зсув усіх біт праворуч
        OUT $12,R17   ;запис в порт D старшої частини R17
        RCALL delay   ;виклик підпрограми затримки
   RJMP L1

delay:   LDI coarse,8;
cagain:     LDI medium,255     ;отримання затримки 1/2 секунди
magain:        LDI fine,255    ;при 4МГц тактовій частоті
fagain:        dec fine;
               brne fagain;
            DEC medium;
           BRNE magain;
        DEC coarse;
        BRNE cagain;
        RET
```

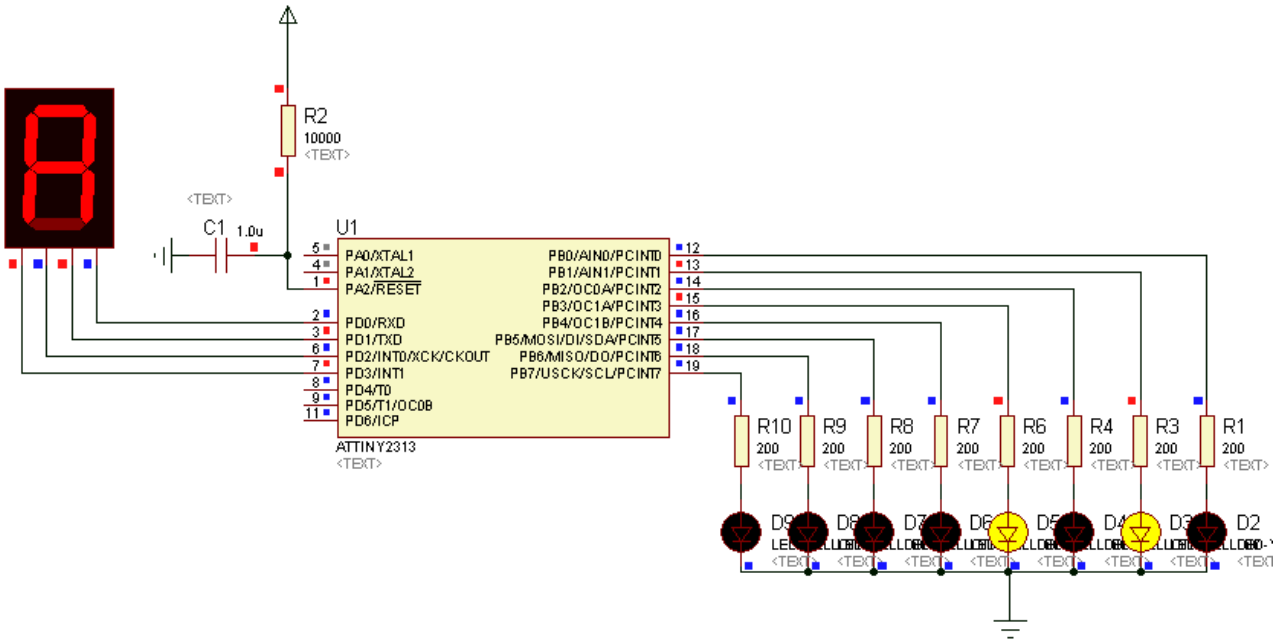

Рисунок 6 – Результат виконання другої частини завдання (виведення молодшої частини числа)

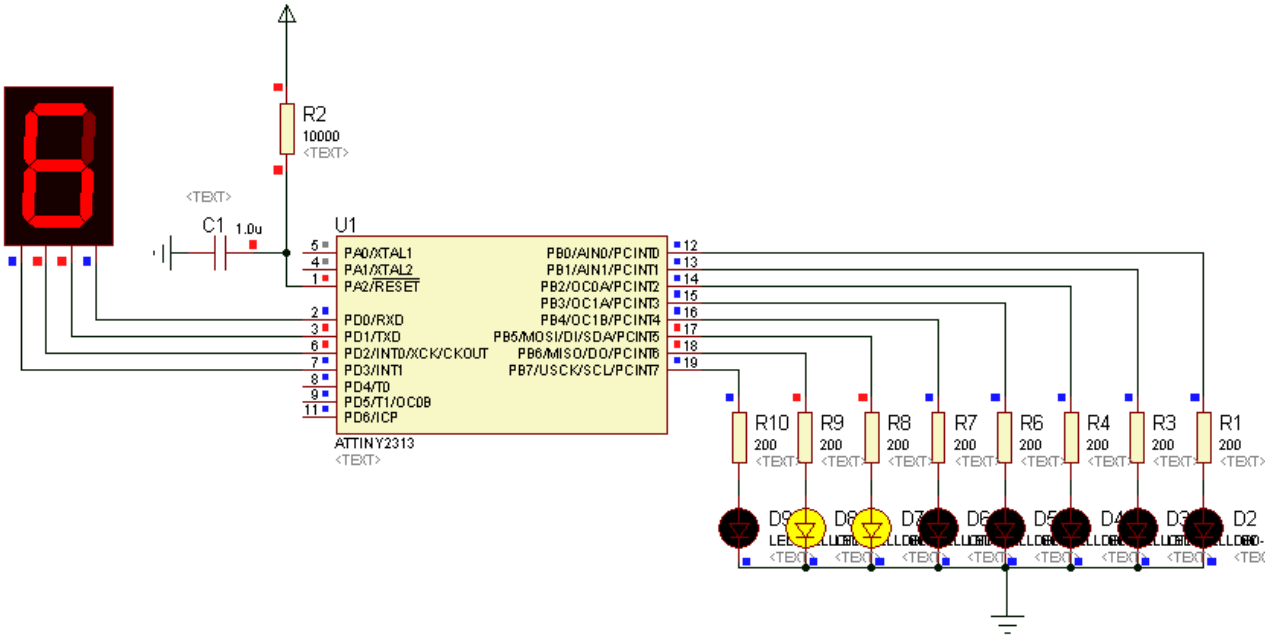

Рисунок 7 – Результат виконання другої частини завдання (виведення старшої частини числа)

Контрольні запитання

1. Наведіть схеми відображення інформації в цифрових пристроях.
2. Що таке статична індикація? Наведіть схемну реалізацію.
3. Чим відрізняється статична індикація від динамічної?
4. Розкажіть про характеристики порту B мікроконтролера.
5. Охарактеризуйте порт D мікроконтролера.

Лабораторна робота № 3

ОПИТУВАННЯ ДИСКРЕТНИХ ДАТЧИКІВ

Мета роботи: надбання навичок програмно-керованого введення інформації з дискретних датчиків та клавіатури до мікроконтролера AT90S2313.

Теоретичні відомості

Практично жоден мікропроцесорний пристрій не обходиться без кнопок і простих датчиків. За допомогою цих периферійних елементів в МПП надходить різна інформація, яка використовується для зміни алгоритму роботи програми. Схема підключення контактного датчика до МК наведена на рисунку 1. У наведеному прикладі датчик підключений до лінії *PD0* порту *D* МК. Через цей вхід МК проводить зчитування стану датчика. Датчик можна підключити і до будь-якої іншої лінії будь-якого з портів МК.

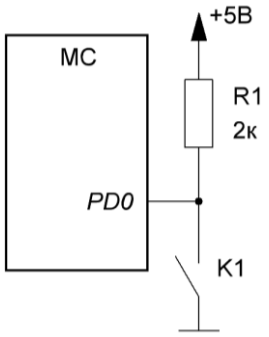

Рисунок 1 – Підключення контактного датчика до МК

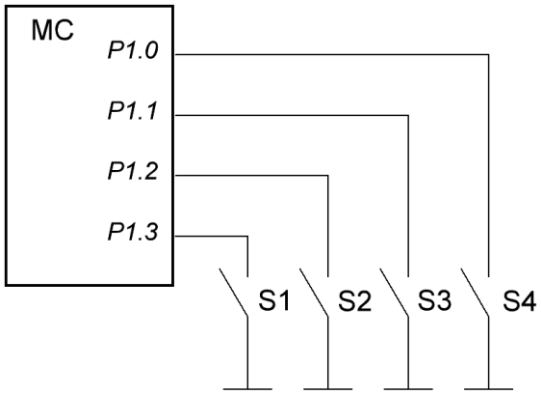

Рисунок 2 – Підключення кнопок або простих датчиків до МК

У початковому стані контакти датчика розімкнені. На вхід МК через резистор R1 прикладається напруга від джерела живлення + 5 В. МК сприймає цю напругу як сигнал логічної одиниці. При спрацьовуванні датчика контакти замикаються і з'єднують вивід МК із загальним дротом. Тепер мікросхема сприймає вхідний рівень сигналу як логічний нуль. Резистор R1 при цьому служить струмообмежувальним елементом, запобігаючи короткому замиканню між шиною живлення і загальним дротом. Деякі МК мають свої внутрішні резистори навантаження, які можуть замінити зовнішній резистор. Схема підключення декількох датчиків або кнопок до МК зображена на рисунку 2.

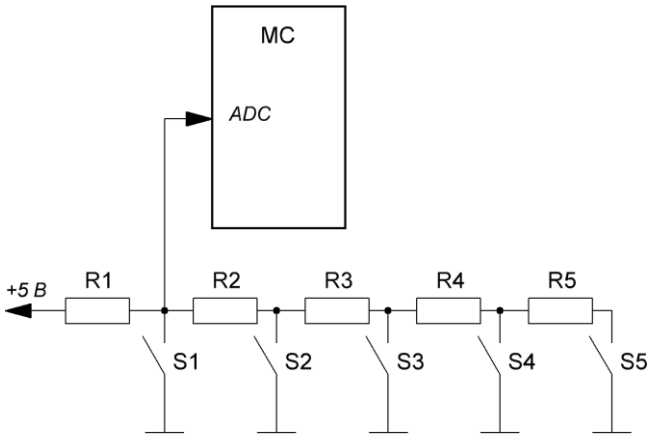

Рисунок 3 – Підключення кнопок зміною напруги на аналоговому вході МК

У схемі, що зображена рисунку 3, при натисканні однієї з клавіш змінюється постійна напруга на відповідному вході процесора, яка розпізнається процесором і дешифрується в певну команду. Ця напруга максимальна (приблизно 5 В), коли кнопки не натиснуті, і мінімальна (0 В) при натиснутій клавіші S1.

Існує два види клавіатур, що підключаються до МК: зі скануванням і з кодуванням. Блок-схема 12-клавішної клавіатури зі скануванням показана на рисунку 4.

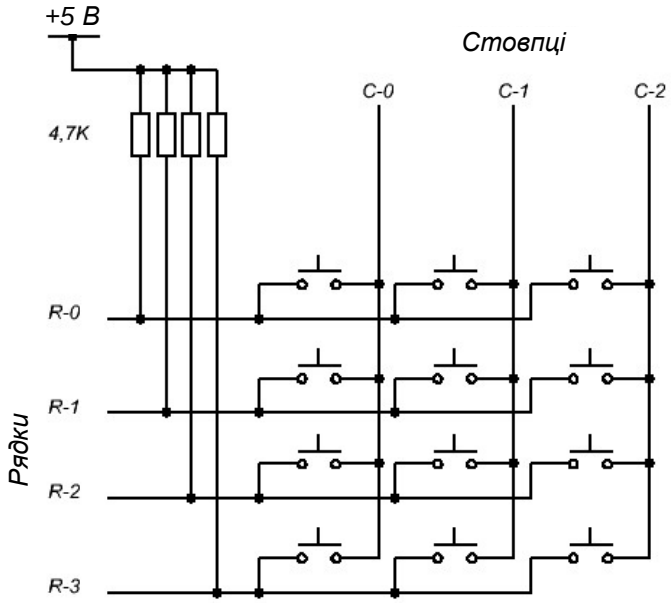

Рисунок 4 – Матрична клавіатура 4×3

Клавіші розташовані у вузлах матриці, у якої чотири лінії рядків і три лінії стовпців. На лінії стовпців по черзі подається негативний імпульс (логічний "0"). У цей момент перевіряється стан чотирьох ліній рядків. Якщо натиснутих клавіш немає, всі лінії рядків мають високий рівень (вони підключені до напруги *+5 В* через резистори). Якщо ж клавіша натискається, і на лінії стовпця, відповідного натиснутій клавіші, все ще нуль, то адекватна лінія рядка також стає рівною нулю. Знаючи номери стовпця і рядка, можна отримати позицію натиснутої клавіші.

У клавіатурі з кодуванням застосовують спеціалізовані мікросхеми, які виявляють натискання клавіші і передають її код. Прикладом такого пристрою є мікросхема MM74C922 (National Semiconductors).

Окрім розпізнавання положення натиснутої клавіші слід програмно захиститися від "тремтіння" контактів, тобто від впливу перехідних процесів, а також від ситуацій, пов'язаних з одночасним натисканням декількох клавіш.

Для того, щоб запобігти протіканню небезпечних струмів при одночасному замиканні декількох клавіш у одному стовпці, в лініях R-0...R-3 зазвичай встановлюють послідовно резистори або діоди. З цією ж метою можна використовувати і інший метод сканування, при якому всі неактивні горизонтальні шини, окрім шини нуля, що "біжить", програмно

призначаються входами. Впливу перехідних процесів можна уникнути, якщо повторно зчитувати стан входів матриці сканування через певну часову затримку.

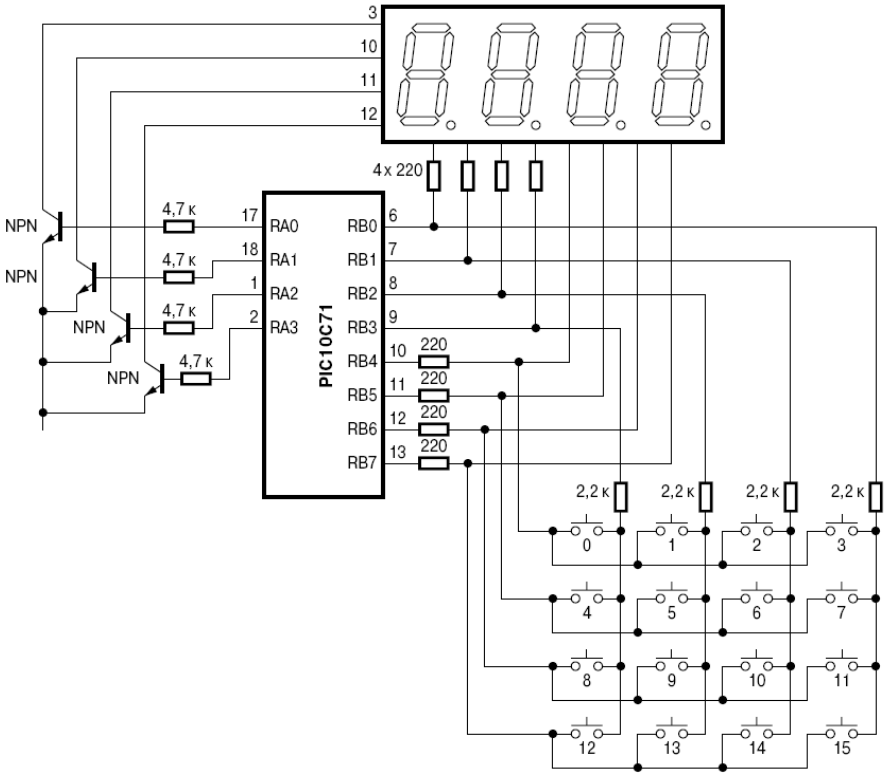

Рисунок 5 – Мультиплексорне керування матричною клавіатурою та семисегментним світлодіодним табло

Для раціонального використання ресурсу портів їх виводи, які використовуються для сканування клавіатури, можна використовувати і для інших функцій, наприклад, для підтримки динамічної світлодіодної індикації. На рисунку 5 показаний приклад такого мультиплексування портів МК.

Схема клавіатури з дешифратором наведена на рисунку 6. У цій схемі для вибору одного з чотирьох стовпців клавіатури використовується дешифратор *DD2* типу К555ИД4. У такій схемі для сканування стовпців МК повинен подавати на виходи *PD0* і *PD1* двійковий код номера стовпця. Код надходить на входи *A0* і *A1* дешифратора. В результаті один з його виходів (той, номер якого відповідає коду, що надійшов) прийме нульове значення. На решті виходів буде одиниця. Так, при коді 00 на вході дешифратора вихід *Q0* (вивід 9) приймає нульове значення. Для коду 01 – нуль буде на виході *Q1*

і так далі. Таким чином, МК може перебирати всі чотири стовпці, використовуючи всього два розряди.

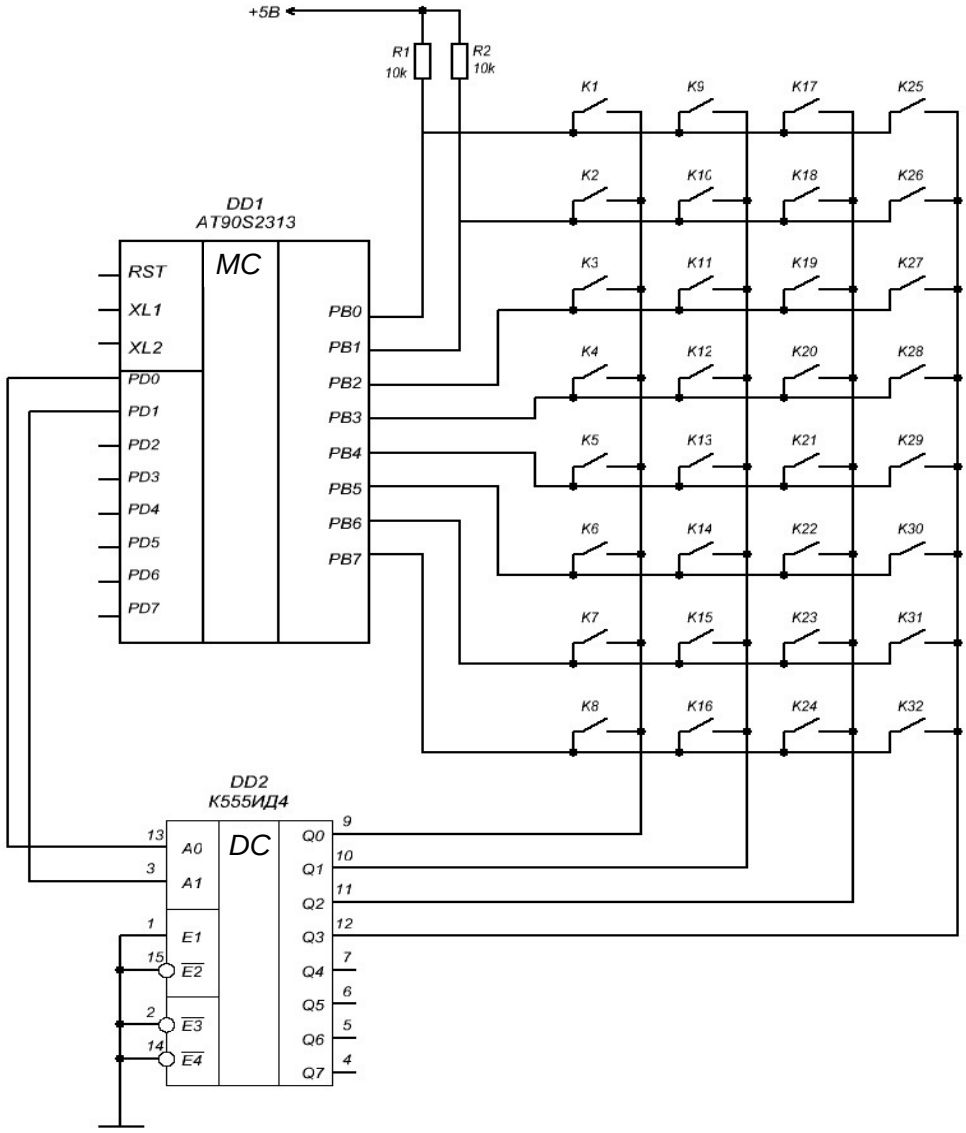

Рисунок 6 – Підключення клавіатури з використанням дешифратора

Існують й інші способи зменшення числа використання виводів МК, призначених для сканування клавіатури. Реалізація одного з таких способів наведена на рисунку 7. У звичайних матричних клавіатурах при замиканні кожного контакту утворюється електричне з'єднання між вихідною і вхідною ланками схеми сканування. Але якщо змінити топологію розташування клавіш так, щоб при замиканні контакту забезпечувалася зміна потенціалів групи шин, то число кодованих клавіш можна значно збільшити. Така конструкція клавіатури забезпечує кодування більшого числа клавіш, ніж

звичайна матрична, при однаковому числі шин, що використовуються для кодування. Збільшення досягнуте за рахунок того, що додаткові кодові комбінації утворюються парами шин кодування.

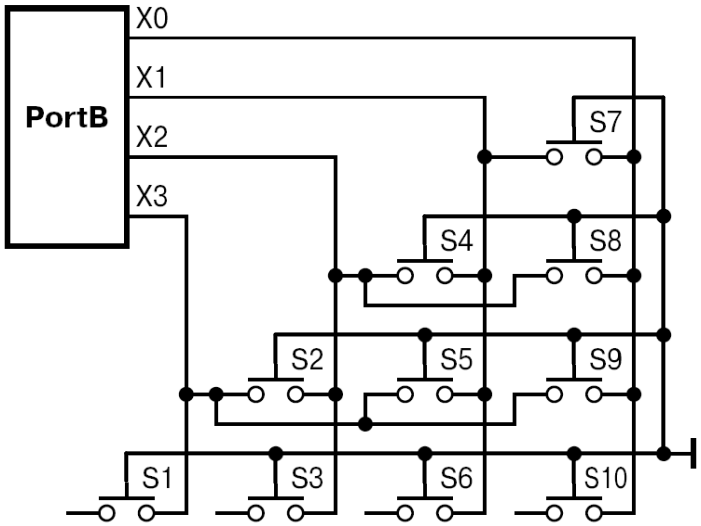

Рисунок 7 – Топологія клавіатури з одним контактором

Для подібного варіанта топології сканування не потрібне. Всі шини є входами з підтягуванням до напруги живлення резисторами. Контактор з'єднаний з загальною шиною. При замиканні контакту потенціал контактора передається на одну або дві кодові шини, на перетині яких він знаходиться. Якщо ввести другий замикальний контактор з потенціалом живлення, то можна вдвічі збільшити число сканованих клавіш. Для цього варіанту необхідно проводити сканування, наприклад, з періодом 20-30 мс. При скануванні використовується ефект пам'яті на паразитній ємкості шин. У першій фазі сканування всі шини визначаються виходами, і на них подається потенціал логічного 0. Потім виводи порту перевизначаться як входи, і проводиться зчитування їх стану. Факт замикання фіксується за зміною потенціалу, після чого проводиться друга фаза сканування. Цього разу на шини подається потенціал живлення, проводиться аналіз можливих станів шин і визначаються позиції замкнутих контактів.

```
                    ;Підпрограма опитування клавіатури 3*5
;Опитування першого стовпця
Klava:
        ldi temp, 0b11110111        ;код стовпця 1
```

```
        out PortB, temp                 ;
        sbis PinB,0                     ;Перевіряємо  клавішу 1
        ldi sklav,1                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,1                     ;Перевіряємо  клавішу 6
        ldi sklav,6                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,2                     ;Перевіряємо  клавішу 11
        ldi sklav,11                    ;Якщо натиснута зберігаємо  код в
                                        ;клавіші в регістрі стану клавіатури

; Опитування  другого стовпця

        ldi temp, 0b11101111        ;код стовпця 2
        out PortB, temp                 ;
        sbis PinB,0                     ;Перевіряємо  клавішу 2
        ldi sklav,2                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,1                     ;Перевіряємо  клавішу 7
        ldi sklav,7                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,2                     ;Перевіряємо  клавішу 12
        ldi sklav,12                    ;Якщо натиснута зберігаємо  код в
                                        ;клавіші в регістрі стану клавіатури
; Опитування  третього стовпця

        ldi temp, 0b11011111        ;код стовпця 3
        out PortB, temp                 ;
        sbis PinB,0                     ;Перевіряємо  клавішу 3
        ldi sklav,3                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,1                     ;Перевіряємо  клавішу 8
        ldi sklav,8                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,2                     ;Перевіряємо  клавішу 13
        ldi sklav,13                    ;Якщо натиснута зберігаємо  код в
                                        ;клавіші в регістрі стану клавіатури

; Опитування  четвертого стовпця

        ldi temp, 0b10111111        ;код стовпця 4
        out PortB, temp                 ;
        sbis PinB,0                     ;Перевіряємо  клавішу 4
        ldi sklav,4                     ;Якщо натиснута зберігаємо  код
                                        ;клавіші в регістрі стану клавіатури
        sbis PinB,1                     ;Перевіряємо  клавішу 9
```

```
        ldi sklav,9                 ;Якщо натиснута зберігаємо  код
                                    ;клавіші в регістрі стану клавіатури
        sbis PinB,2                 ;Перевіряємо  клавішу 14
        ldi sklav,14                ;Якщо натиснута зберігаємо  код в
                                    ;клавіші в регістрі стану клавіатури

; Опитування  п'ятого  стовпця

        ldi temp, 0b01111111        ;код стовпця 5
        out PortB, temp             ;
        sbis PinB,0                 ;Перевіряємо  клавішу 5
        ldi sklav,5                 ;Якщо натиснута зберігаємо  код
                                    ;клавіші в регістрі стану клавіатури
        sbis PinB,1                 ;Перевіряємо  клавішу 10
        ldi sklav,10                ;Якщо натиснута зберігаємо  код
                                    ;клавіші в регістрі стану клавіатури
        sbis PinB,2                 ;Перевіряємо  клавішу 15
        ldi sklav,15                ;Якщо натиснута зберігаємо  код в
                                    ;клавіші в регістрі стану клавіатури
        rcall drebezg               ;Виклик  підпрограми захисту від  тремтіння
                                    ;контактів
; Достатьо 4 NOP. Потрібно для того, щоб перед перевіркою дати ніжці час на
; те, щоб зайняти потрібний рівень. Річ у тому, що реальна схема має деяке
; значення C і L, яке робить неможливою миттєву зміну рівня, невелика затримка
; все ж є. А на швидкості в 8Мгц і вище процесор клацає команди з такою, що
; швидкістю напруга на виводі  ще не спала, а ми вже перевіряємо стан виводу.

        ret                         ;Вихід з підпрограми
```

Завдання до лабораторної роботи

До порта B підключені 8 світлодіодів або індикатор. До порта D підключена клавіатура 3x3. Написати програму для мікроконтролера AT90S2313 (режим переривання не використовується), яка виконує завдання згідно таблиці 1. Зібрати схему до лабораторної роботи у середовищі Proteus та перевірити виконання програми.

Таблиця 1 – Варіанти індивідуальних завдань

№	Текст індивідуального завдання

1	Натиснути будь-які кнопки на клавіатурі, відображати номер кнопок на індикаторі.
2	Натиснути кнопку S2, здійснити періодичне миготіння числа 43
3	У двійковому виді відображати на одиничних індикаторах номер натиснутої кнопки.
4	Здійснити включення/вимикання одиничних індикаторів HL1-HL8 у кількості відповідному номеру натиснутої клавіші .
5	На індикаторі відобразити число 10. По натисканню кнопки S5, здійснювати зменшення на одиницю відображуваного числа.
6	На індикаторі відобразити число 12. По натисканню кнопки S6, здійснювати збільшування на одиницю відображуваного числа.
7	На індикаторі відобразити число 14. По натисканню кнопки S7, здійснювати зменшування на два відображуваного числа.
8	Натиснути кнопку 1 на клавіатурі. Відобразити на індикаторі число 1, здійснити періодичне миготіння числа 4.
9	Натиснути будь-яку кнопку на клавіатурі. На всіх розрядах здійснити відображення номера кнопки.
10	Натиснути кнопку S5 відобразити на індикаторі число 10. Натиснути S6 відобразити на індикаторі число 11.
11	Відобразити на індикаторі число 21. По натисканню кнопки S1 зменшувати на одиницю відображуване число, по натисканню S2 збільшувати на одиницю відображуваного числа.
12	Відобразити на індикаторі число 8. По натисканню кнопки S9 зменшувати на одиницю відображуване число, по натисканню S8 збільшувати на одиницю відображуваного числа.
13	Якщо натиснута кнопка від 0 до 4 відобразити на індикаторі номер цієї кнопки, в усіх інших випадках горять всі індикатори.

14	Сформувати світіння індикаторів HL1-HL8 у залежності від номера натиснутої кнопки на клавіатурі.
15	Якщо натиснута кнопка від 5 до 8 відобразити на індикаторі номер цієї кнопки, в усіх інших випадках горять всі індикатори.

Зміст звіту

1. Схема до лабораторної роботи у середовищі Proteus.
2. Завдання до лабораторної роботи.
3. Текст програми з поясненнями.
4. Результати роботи схеми та програми у середовищі Proteus (screenshot).

Приклад виконання завдання

До порта *D* підключений статичний індикатор. До порта *B* підключена клавіатура 4x3. Написати програму для мікроконтролера AT90S2313 (режим переривання не використовується), яка виконує відображення на індикаторі номера натиснутої клавіші від 0 до 9.

```
.include "2313def.inc"
ldi R20, ramEnd                 ;налаштування стеку
OUT SPL, R20                    ;
ldi r20, 0xFF                   ;налаштування на режим роботи порта D
out ddrd, r20                   ;
ldi r20, 0b00000111             ;налаштування на режим роботи порта B
out ddrb, r20                   ;
ldi r20, 0b11111111             ;включення резисторів «Pull-up»
out portb, r20                  ;

l2:                             ;перевірка натискання «1»
cbi portb,0
rcall DELAY
in r21, pinb
andi r21, 0b01111000

cpi  r21, 0b01110000            ;перевірка натискання «4»
brne rr1
ldi r22, 0x01
out portd, r22
rr1:

cpi r21, 0b01101000             ;перевірка натискання «7»
```

```
brne rr4
ldi r22, 0x04
out portd, r22
rr4:

cpi r21, 0b01011000
brne rr7
ldi r22, 0x07
out portd, r22
rr7:

sbi portb,0
clr r21
                                        ;перевірка натискання «2»
cbi portb,1
rcall DELAY
in r21, pinb
andi r21, 0b01111000

cpi r21, 0b01110000
brne rr2
ldi r22, 0x02
out portd, r22
rr2:

cpi r21, 0b01101000                     ;перевірка натискання «5»
brne rr5
ldi r22, 0x05
out portd, r22
rr5:

cpi r21, 0b01011000                     ;перевірка натискання «8»
brne rr8
ldi r22, 0x08
out portd, r22
rr8:

cpi r21,  0b00111000                    ;перевірка натискання «0»
brne rr0
ldi r22, 0x00
out portd, r22
rr0:

sbi portb,1
clr r21

cbi portb,2                             ;перевірка натискання «3»
rcall DELAY
in r21, pinb
andi r21, 0b01111000

cpi r21,  0b01110000
brne rr3
ldi r22, 0x03
out portd, r22
rr3:

cpi r21,  0b01101000                    ;перевірка натискання «6»
```

```
brne rr6
ldi r22, 0x06
out portd, r22
rr6:

cpi r21,  0b01011000                ;перевірка натискання «9»
brne rr9
ldi r22, 0x09
out portd, r22
rr9:

sbi portb,2
clr r21

rjmp l2
nop

                                    ;підпрограма затримки для програмного ;усунення
DELAY:                              тремтіння контактів
ldi r18,0x15
d1:
clr r19
d2:
dec r19
brne d2
dec r18
brne d1
ret
```

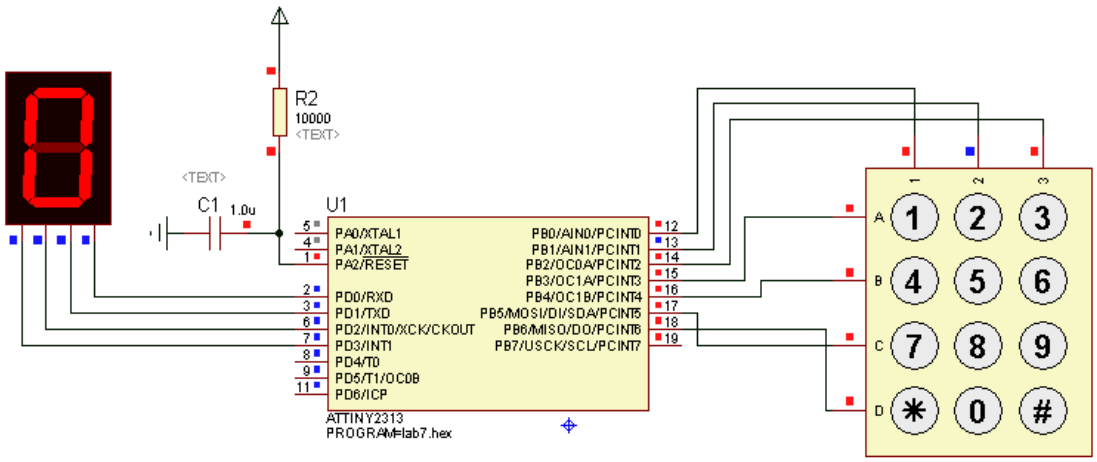

Рисунок 8 – Початковий режим роботи схеми

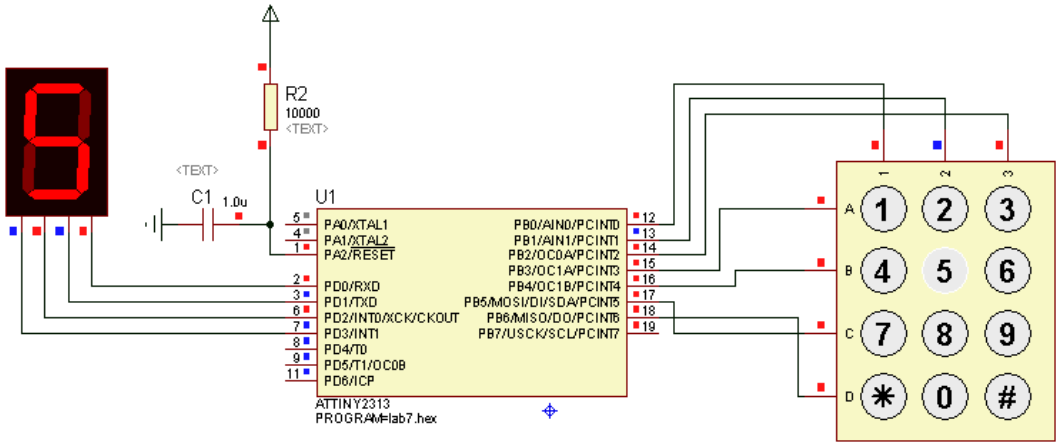

Рисунок 9 – Натиснення клавши та відображення її на індикаторі

Контрольні запитання

1. Організуйте клавіатуру на мікроконтролері AT90S2313 на 16 клавіш.
2. Організуйте клавіатуру на мікроконтролері AT90S2313 на 10 клавіш з використанням 4 портів мікроконтролера.
3. Організуйте клавіатуру на мікроконтролері AT90S2313 на 16 клавіш на IMC MM74C922.
4. Організуйте клавіатуру на мікроконтролері AT90S2313 на 32 клавіші.
5. Як організувати клавіатуру з використанням регістр зсуву SN74198N?

Лабораторна робота № 4

ОРГАНІЗАЦІЯ ДИНАМІЧНОЇ ІНДИКАЦІЇ

Мета роботи: Вивчення особливостей роботи динамічних цифрових індикаторів. Розробка програм для мікроконтролера AT90S2313 відображення інформації на індикаторах динамічного типу.

Теоретичні відомості

1 Організація динамічної індикації

Дуже часто МК використовується не тільки для керування роботою конструкції, але й для того, щоб повідомити що-небудь користувачеві і/або отримати від нього які-небудь вказівки про подальшу роботу. 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.

Для організації динамічної індикації застосовується матриця, що складається з ліній рядків і ліній стовпців (рис. 1). На перетині стовпця і рядка матриці розташований індикаторний елемент – світлодіод.

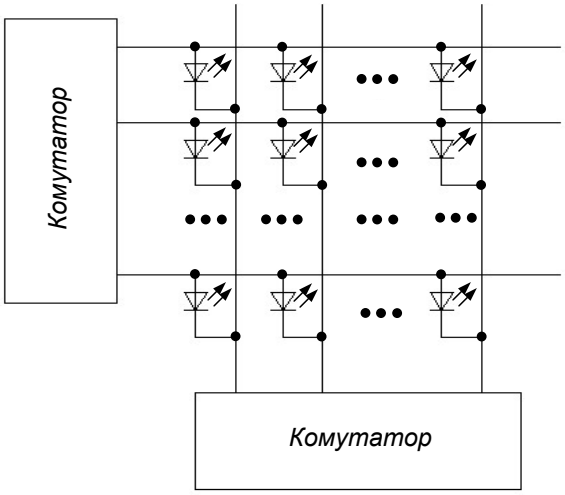

Рисунок 1 – Організація динамічної індикації

Для того, щоб запалити той або інший елемент, необхідно подати на матрицю не один, як у звичайних індикаторах, а два сигнали: логічна 1 на відповідному рядку і логічний 0 на відповідному стовпці матриці. Через односторонню провідність світлодіода кожна комбінація сигналів на входах рядків і стовпців однозначно включає рівно один індикаторний елемент.

Головна перевага динамічної індикації – невелике число ліній, що керують: для матриці світлодіодів розміром N×N елементів потрібно всього 2N сигналів, що керують. За таку економію, втім, доводиться платити – справа в тому, що при почерговому виведенні інформації на кожен світлодіод матриці його яскравість світіння буде в N^2 разів нижча, ніж при безпосередньому виведенні інформації на один світлодіод, що "окремо стоїть". Тому в пристроях, що використовують динамічну індикацію, виведення інформації здійснюється не на кожен світлодіод окремо, а на один рядок або на один стовпець повністю – у цьому випадку яскравість світіння світлодіодів падає тільки в N разів.

Розглянемо декілька варіантів реалізації динамічної індикації із застосуванням МК. Як індикаторний елемент передбачається застосування семисегментних індикаторів, але кожен такий індикатор з легкістю можна замінити і групою світлодіодів.

Схема реалізації динамічної індикація без додаткових елементів наведена на рисунку 2. До порту *B* МК підключені катоди всіх світлодіодів матриці, а до порту *A* – аноди кожного з індикаторів, що створюють матрицю. На лініях порту *A* організовується одиниця, що "біжить". На лінії порту *B* 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. Таке рішення для МК може забезпечити роботу не більше 5 семисегментних індикаторів одночасно, і при цьому МК стає "глухим і сліпим", оскільки жодної вільної лінії у нього не залишається. При використанні інших МК з великою кількістю ніжок вказана проблема знімається.

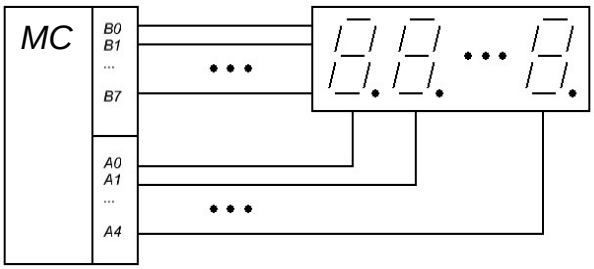

Рисунок 2 – Схема реалізації динамічної індикація без додаткових елементів

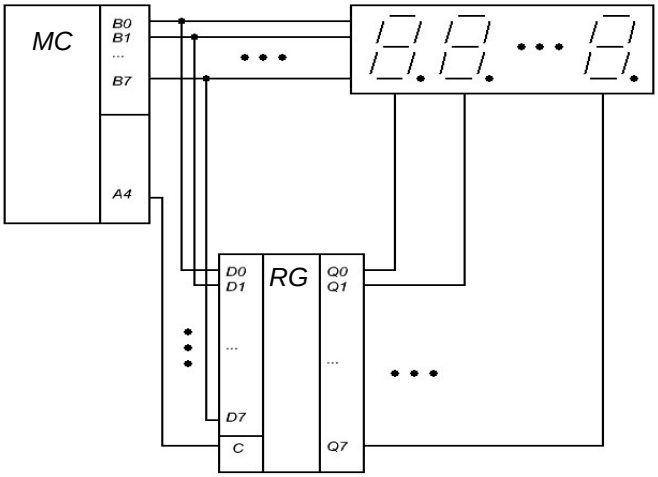

Рисунок 3 – Схема реалізації динамічної індикації з одним додатковим елементом

Схема реалізації динамічної індикації з одним додатковим елементом наведена на рисунку 3. До порту *B* МК підключені катоди всіх світлодіодів матриці, а також – входи буферного регістра. До виходів же буферного регістра підключені аноди кожного з семисегментних індикаторів. На порт *B* спочатку виводиться одиниця, що біжить, яка записується в буферному регістрі сигналом *C*, що утворюється однією з ліній порту *A*. Потім на лінії порту *B* видається семисегментний код символу, який повинен горіти в знакомісці, визначуваному сигналом на виході буферного регістра. В даному випадку лінії порту *B* використовуються в режимі часового мультиплексування, тобто, ними по черзі передається і код символу, і номер знакомісця. Перевага такого способу індикації – менша витрата ліній портів МК (окрім порту *B* – всього одна додаткова лінія) і можливість роботи до 8 індикаторів одночасно. Ще один варіант індикації з одним додатковим елементом наведений на рисунку 4.

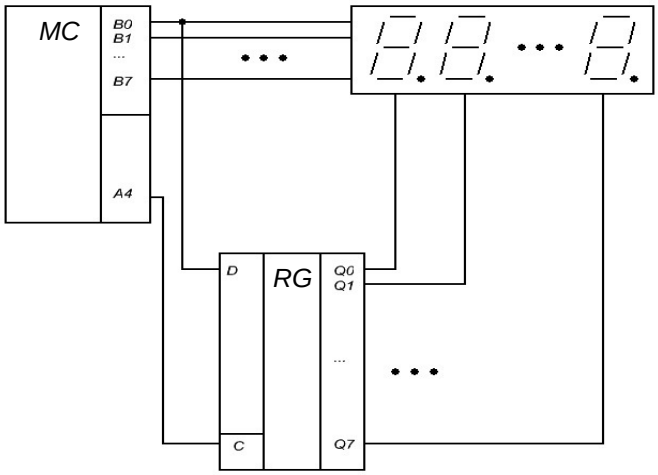

Рисунок 4 – Схема реалізації динамічної індикації з регістром зсуву

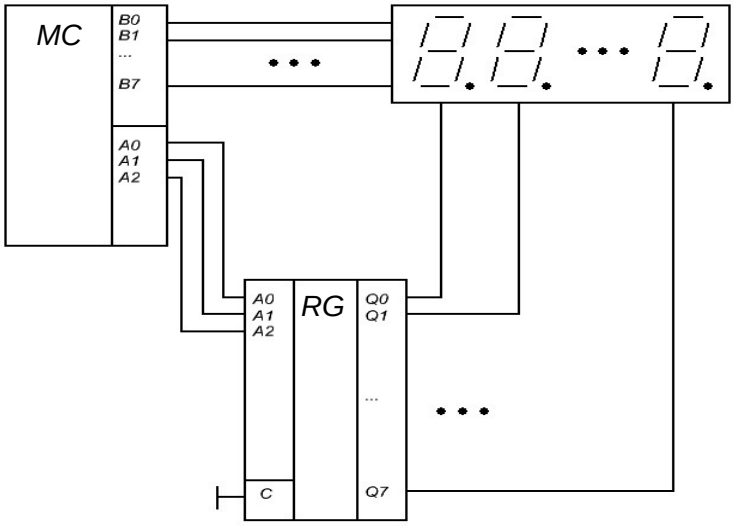

Рисунок 5 – Схема реалізації динамічної індикації з дешифратором

У цьому варіанті одиниця, що біжить, реалізується за допомогою регістра зсуву. Порт *B* у цій схемі також використовується в режимі часового мультиплексування – як для видачі символу, так і для занесення чергового біта до регістра зсуву. Схема також вимагає всього одну додаткову лінію (крім порту *B*), але за габаритами виходить трохи менше попереднього варіанта. Ще один варіант реалізації схеми з одним додатковим елементом наведений на рисунку 5. Така схема придатна тільки для індикаторів із загальним катодом, оскільки для організації нуля, що біжить, в ній використовується дешифратор. Схема вимагає три додаткові лінії (крім порту *B*), проте у багатьох випадках вона може виявитися цілком корисною.

Схема реалізації динамічної індикації з двома додатковими елементами наведена на рисунку 6.

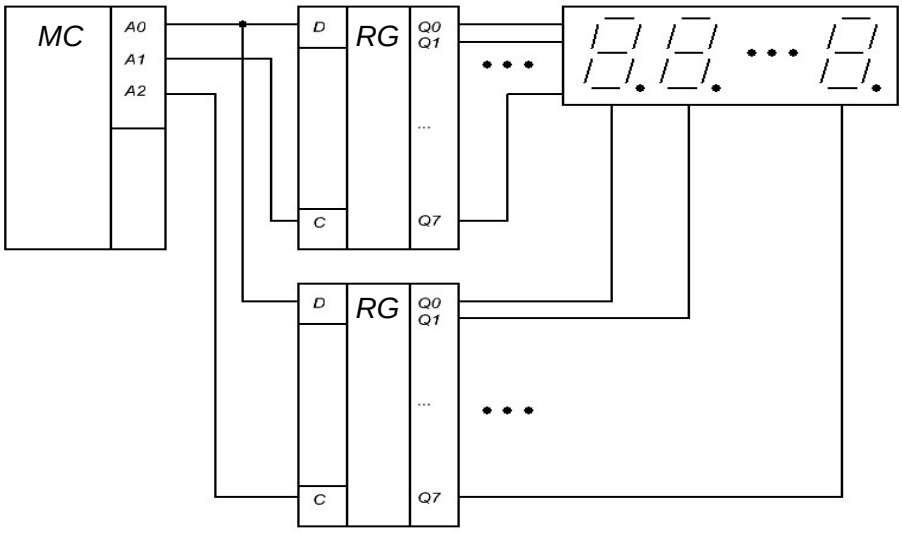

Рисунок 6 – Схема реалізації динамічної індикації з двома додатковими елементами

У схемі використовуються два послідовні регістри зсуву: один – для розгортки зображення по стовпцях (замінює порт *A*), інший – для розгортки зображення по рядках (замінює порт *B*). Перевага такого рішення – всього три лінії порту МК. Крім того, при такому варіанті реалізації блок динамічної індикації легко оформити і у вигляді окремої плати. Недолік такого рішення – два додаткових компоненти.

Ще одна схема реалізації наведена на рисунку 7. Перевага такої схеми – всього дві лінії порту. Недолік – складніша програма керування і велика тривалість формування вихідних сигналів, що викликає деяке паразитне підсвічування індикаторів (помітно лише в темряві).

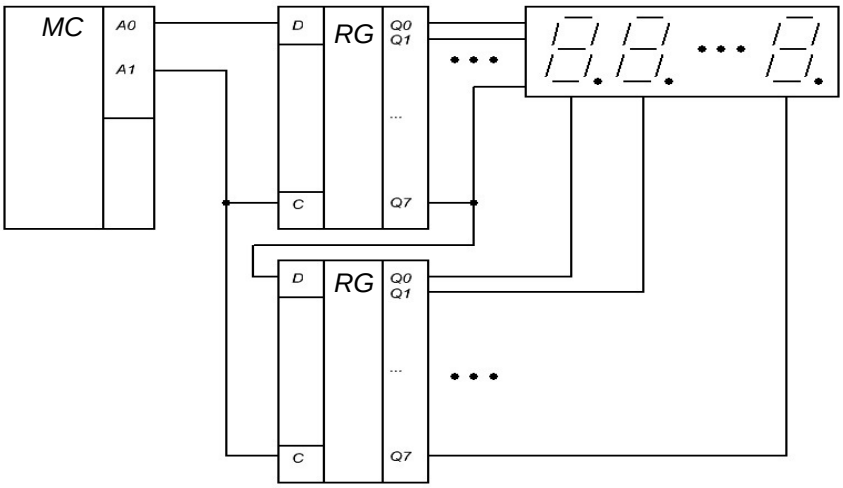

Рисунок 7 – Схема динамічної індикації з двома лініями керування

Практична схема підключення динамічного індикатора на МК AT90S2313за схемою, що наведена на рисунку 5, зображена на рисунку 11. Символ, що буде світитись на індикаторі визначається сигналом логічної одиниці на вивід COM1, 2, 3, 4 індикатора. На шині даних (A, B, C …) активним рівнем є рівень логічного 0.

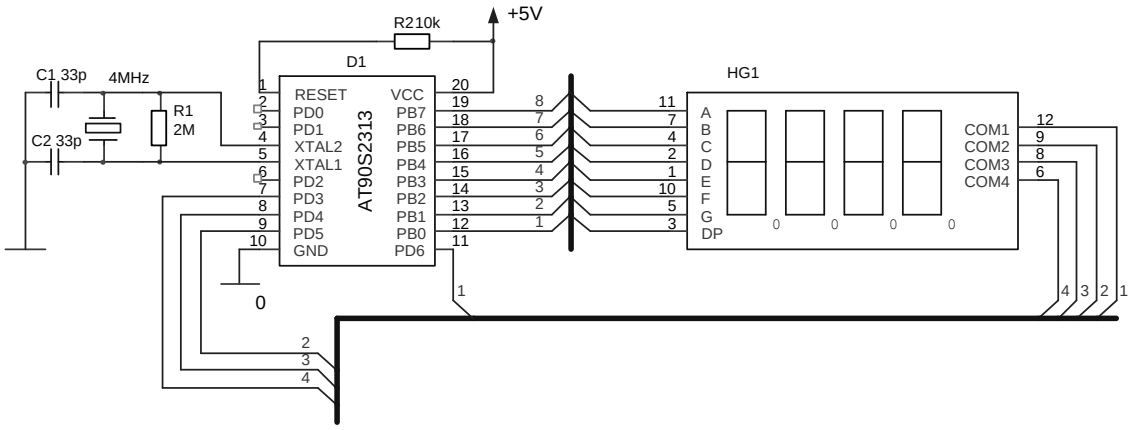

Рисунок 8 – Підключення динамічного індикатора до МК AT90S2313

2 Масиви чисел та їх застосування в мікроконтролерах AVR

У будь-якому місці програми можемо написати:

```
MyArray:
.db 1,15,4,9,12,145,67,90
```

Це є масив "MyArray", що складається з 8 елементів. Масив розміщується у пам’яті програм починаючи з адреси, на якій стоїть мітка (MyArray). Тобто, адресі мітки відповідає 0-й елемент масиву.

Щоб отримати доступ до 1-го, 2-го, 3-го і т.д. елементу, потрібно прочитати осередок FLASH пам’яті програм за адресою, яка більше адреси мітки відповідно на 1, 2, 3. Розглянемо програму, яка читає послідовно елементи масиву MyArray і виводить їх в порт D.

```
        ldi Temp,0          ;ініціалізація регістра
                            ;внутрішньої адресації масиву
ReadArray:
        ldi ZH,High(MyArray*2)      ;завантаження адреси 0-го
        ldi ZL,Low(MyArray*2)       ;елементу у рег. пару Z
        ldi Temp1,0
        add ZL,Temp                 ;додавання
        adc ZH,Temp1            ;внутр. адреси
        lpm                     ;завантаження з пам’яті програм
        mov Temp1,R0            ;копіювання
        out PortD,Temp1         ;вивід у порт
        inc Temp                ;збільш. внутр. адреси
        rjmp ReadArray          ;на початок циклу
MyArray:
.db 12,16,3,4,10,17,255,37,158,14,13,98
.db 14,85,30,9,145,52,64,49,119,72,209,46
```

Розглянемо структуру:

```
ldi ZH,High(MyAray*2)
```

Командою завантажуємо старшу частину адреси в старшу частину пари Z (ZH) по мітці MyArray. Що значить "*2"? Кожна команда містить два байти інформації і займає, таким чином, два осередки ПЗП. Тому, лічильник команд рахує 2 адреси як одну. Мітка містить саме дані для лічильника команд. Щоб одержати реальну адресу ПЗП, необхідно збільшити адресу мітки в 2 рази. Що і робимо. По-друге: у масиві два рядки. Це не означає, що масив - двомірний. Просто, розбиваємо один довгий рядок на декілька коротших - для зручності. Рядків може бути скільки завгодно. *При розбитті, не допускається непарної кількості елементів в рядку, інакше після останнього елементу поточного рядка перед першим елементом наступної, читається один "неіснуючий" елемент.* Це також пов'язано з особливостями адресації програмної пам'яті. Непарним може бути тільки останній рядок. По-третє, дані у масиві можуть бути представлені у вигляді

десяткових, шістнадцятирічних, двійкових чисел, а також, еквівалентних значень символів ASCII.

```
.db 3,4,5,75,32,12              ; десяткові числа
.db 0x2A,0x34,0x17,0xDF             ; шістнадцяткові числі
.db 0b01101001,0b11011100 ; двійкові числа
.db "Тут був Вася"              ; еквіваленти ASCII
```

Завдання до лабораторної роботи

1. Запустити в Proteus файл Din_IND. Підключити файл Din_IND.HEX. Перевірити як працює схема. Розібрати особливості програми Din_IND.ASM. Скласти алгоритм роботи програми.
2. Запустити в Proteus файл Din_IND_KLAV. Підключити файл INDICA~1.HEX. Перевірити як працює схема. Розібрати особливості програми INDICATE+KEYBOARD.ASM. Скласти алгоритм роботи програми.
3. Запустити в Proteus файл Din_IND. Підключити файл DIN_OGON.HEX. Перевірити як працює схема. Розібрати особливості програми DIN_OGON.ASM. Підключити файл DIN_TEN.HEX. Перевірити як працює схема. Розібрати особливості програми DIN_TEN.ASM. Підключити файл. DIN_TO.HEX. Перевірити як працює схема. Розібрати особливості програми DIN_TO.ASM.
4. Використовуючи програми DIN_OGON.ASM, DIN_TEN.ASM, DIN_TO.ASM написати програму, яка створює ефект «вогник, що біжить» по зовнішньому колу (для непарних варіантів) та ефект «тінь, що біжить» по зовнішньому колу (для парних варіантів) – оцінка задовільно. Виконати п.4 для 6 динамічних індикаторів – оцінка добре. Виконати п.4 для 6 динамічних індикаторів з використанням масивів – оцінка відмінно.

Зміст звіту

1. Алгоритм роботи програми та схема до п.1
2. Алгоритм роботи програми та схема до п.2
3. Схема, алгоритм, програма та результати виконання п.4 або п.5 у середовищі Proteus

Приклад виконання завдання

Програма, що створює ефект «вогник, що біжить» по зовнішньому колу для 4 розрядного динамічного індикатора (для схеми Din_IND)

```
        .include "2313def.inc"
        .def Temp=R18
        .def Temp1=R19
ldi R20,0b11111111          ;настройка портів
out ddrb,R20
ldi R20,0b00111111
out ddrd,R20
ldi R16,RamEnd                  ;ініціалізація стека
out SPL,R16
ldi Temp,0                      ;ініціалізація регістра
                            ;внутрішньої адресації масиву
        cli
        CLR R27
        LDI R26,$60
ReadArray:
      ldi ZH,High(MyArray*2)    ;завантаження адреси 0-го
      ldi ZL,Low(MyArray*2)     ;елементу в рег. пару Z
      ldi Temp1,0
      add ZL,Temp               ;додавання
      adc ZH,Temp1          ;внутр. адреса

      lpm                   ;завантаження з пам'яті програм
      ST X, R0              ;копіювання
      INC R26
      inc Temp                  ;збільшення. внутр. адреси
      CPI TEMP,45
      BRNE ReadArray        ;на початок  циклу
        RJMP IndicCycle
MyArray:
.db $0E,$09,$09,$39
.db $0D,$09,$09,$39
.db $0B,$09,$09,$39
.db $0F,$01,$09,$39
.db $0F,$09,$01,$39
.db $0F,$09,$09,$31
.db $0F,$09,$09,$29
.db $0F,$09,$09,$19
.db $0F,$09,$09,$38
.db $0F,$09,$08,$39
.db $0F,$08,$09,$39

;*********************************************************
;MAIN
;*********************************************************
IndicCycle:    LDI R26,$60
                  rcall Display      ;цикл індикації
```

```
                rjmp IndicCycle
;*********************************************************
;послідовний вивід на динамічний індикатор
Display:
                …
                RET
```

Контрольні запитання

1. Розкажіть про принцип динамічної індикації та наведіть його схему.
2. Наведіть схемну реалізацію динамічної індикації без додаткових елементів.
3. Наведіть схемну реалізацію динамічної індикації з одним додатковим елементом.
4. Наведіть схемну реалізацію динамічної індикації з регітром зсуву.
5. Наведіть схемну реалізацію динамічної індикації з дешифратором.
6. Наведіть схемну реалізацію динамічної індикації з двома лініями керування.
7. Для яких задач доцільно використовувати масиви чисел у програмі?
8. Наведіть масив чисел для перекодування двійкового числа від 0 до 15 в код семисегментного індикатора з загальним катодом.