**Лабораторная работа №3**

**Тема:** Матричный индикатор

**Задание:** 1. Реализовать динамическую индикацию на базе светодиодной матрицы 8x8. Частота обновления всего изображения должна составлять не менее 50 Гц.

2. Реализовать эффект бегущей строки с частотой около 2 Гц.

Тактовая частота МК - 8 МГц, таблицу перекодировки считать имеющейся, индицируемая строка - любая, но не менее 5 символов.

**Ход работы**

Динамическая индикация подразумевает, что в каждый конкретный момент времени будет индицироваться только часть изображения. И за счет быстрого чередования этих частей у человека складывается впечатление цельного изображения. В случае матричного индикатора 8x8 такой частью может быть один столбец, в котором присутствуют 8 светодиодов. Т.е. полный цикл отображения какой-то картинки занимает 8 операций: отобразить 1-ый столбец, отобразить 2-ой столбец, ....., отобразить 7-ой столбец, отобразить 8-ой столбец. Если есть требование по частоте обновления всего изображения, то можно получить требуемую частоту чередования столбцов 8\*50=400 Гц.

Для эффекта бегущей строки нужно смещать изображение строки относительно матрицы с небольшой скоростью - чтобы человек нормально видел это смещение. По заданию частота сдвига изображения на один столбец составляет около 2 Гц. Если ширина изображения составляет 40 столбцов, например, то сначала надо отображать с 1-го по 8-ой столбцы, через примерно 0,5 с нужно начать отображать со 2-го по 9-ый столбцы, еще через 0,5 с - с 3-го по 10-ый столбцы и т.д.

Задачу отсчета задержек можно возложить на таймеры. В нашем случае можно использовать 2 таймера. Для задачи динамической индикации рекомендую взять таймер 0, а для эффекта бегущей строки - таймер 1. Точно подбирать частоту прерываний таймеров не надо, достаточно получить больше 400 Гц и около 2 Гц. Частота прерываний вычисляется по формуле

, где - тактовая частота МК; - коэффициент предделителя таймера; n - разрядность таймера.

Для взаимодействия с матрицей понадобится 2 порта - пусть это будут PORTA и PORTB. PORTA - строки матрицы (1 - светодиод горит, 0 - не горит), PORTB - активация столбцов матрицы (1 - столбец активный, 0 - неактивный).

Алгоритм основной программы:

1. Настроить порты на выход и выдать 0 на оба порта

2. Настроить таймеры и разрешить от них прерывания по переполнению

3. Произвести перекодировку индицируемой строки и результат расположить в буфере изображения

4. Инициализировать необходимые переменные cnt=0, Abeg=Image\_addr

5. Бесконечный цикл

Алгоритм прерывания таймера 0:

1. Выдать на PORTB 0

2. Увеличить счетчик столцов cnt на 1

3. Проверить, не превышает ли cnt значение 7 (если превышает, то cnt=0)

4. Взять из буфера изображения значение столбца по адресу Abeg+cnt

5. Отправить полученное значение в PORTA

6. Сдвинуть 1 влево на cnt бит и полученное значение отправить в PORTB

Алгоритм прерывания таймера 1:

1. Увеличить начальный адрес индицируемого участка Abeg на 1

2. 3. Проверить, не превышает ли Abeg конечного значения (если превышает, то Abeg=Image\_addr)

**Код программы:**

.include "m16def.inc"

.def temp = R16

.def cnt\_temp = R17

.def cnt = R18

.def AbegL = R26 ;XL

.def AbegH = R27 ;XH

.equ String\_max\_length = 64

; FlashROM

.cseg

.org 0

jmp RESET ; Reset Handler

jmp EXT\_INT0 ; IRQ0 Handler

jmp EXT\_INT1 ; IRQ1 Handler

jmp TIM2\_COMP ; Timer2 Compare Handler

jmp TIM2\_OVF ; Timer2 Overflow Handler

jmp TIM1\_CAPT ; Timer1 Capture Handler

jmp TIM1\_COMPA ; Timer1 CompareA Handler

jmp TIM1\_COMPB ; Timer1 CompareB Handler

jmp TIM1\_OVF ; Timer1 Overflow Handler

jmp TIM0\_OVF ; Timer0 Overflow Handler

jmp SPI\_STC ; SPI Transfer Complete Handler

jmp USART\_RXC ; USART RX Complete Handler

jmp USART\_UDRE ; UDR Empty Handler

jmp USART\_TXC ; USART TX Complete Handler

jmp ADC\_C ; ADC Conversion Complete Handler

jmp EE\_RDY ; EEPROM Ready Handler

jmp ANA\_COMP ; Analog Comparator Handler

jmp TWSI ; Two-wire Serial Interface Handler

jmp EXT\_INT2 ; IRQ2 Handler

jmp TIM0\_COMP ; Timer0 Compare Handler

jmp SPM\_RDY ; Store Program Memory Ready Handler

RESET:

ldi temp,high(RAMEND) ; Main program start

out SPH,temp ; Set Stack Pointer to top of RAM

ldi temp,low(RAMEND)

out SPL,temp

ldi temp,0b11111111 ;настройка порта А на выход

out DDRA,temp

ldi temp, 0 ;начальная комбинация для порта А

out PORTA,temp

ldi temp,0b11111111 ;настройка порта C на выход

out DDRB,temp

ldi temp, 0 ;начальная комбинация для порта C

out PORTB,temp

;настройка таймеров

ldi temp,(0<<CS02|1<<CS01|1<<CS00) ; /64 --> 488 Гц при 8 МГц

out TCCR0, temp

ldi temp,(0<<CS12|1<<CS11|1<<CS10) ; /64 --> 1,9 Гц при 8 МГц

out TCCR1B, temp

ldi temp, (1<<TOIE0|1<<TOIE1) ;разрешения прерываний по переполнению таймеров T0 и T1

out TIMSK, temp

;перекодирование строки Out\_String, определение ее реальной длины и адреса ячейки после реального буфера изображения

ldi ZL,low(2\*Out\_String)

ldi ZH,high(2\*Out\_String)

rcall ROM\_String\_To\_Image

;настройка начальных значений переменных

clr cnt

ldi AbegL,low(Image)

ldi AbegH,high(Image)

sei ;глобальное разрешение прерываний

Start: ;бесконечный цикл

rjmp Start

ROM\_String\_To\_Image:

clr cnt\_temp

ldi YL,low(Image)

ldi YH,high(Image)

Next\_symbol:

lpm temp,Z+

cpi temp,0

breq End\_of\_string

inc cnt\_temp

push cnt\_temp

push ZL

push ZH

ldi ZL,low(2\*Symbol\_To\_Image)

ldi ZH,high(2\*Symbol\_To\_Image)

ldi cnt\_temp,8

mul temp,cnt\_temp

add ZL,R0

adc ZH,R1

ldi cnt\_temp,8

Load\_symbol\_image:

lpm temp,Z+

st Y+,temp

dec cnt\_temp

brne Load\_symbol\_image

pop ZH

pop ZL

pop cnt\_temp

cpi cnt\_temp,String\_max\_length

breq End\_of\_string

rjmp Next\_symbol

End\_of\_string:

cpi cnt\_temp,0

brne Not\_zero\_length

;какие-нибудь действия, если попалась строка нулевой длины

Not\_zero\_length:

sts Real\_string\_length,cnt\_temp

sts (Addr\_after\_real\_image+0),YL

sts (Addr\_after\_real\_image+1),YH

ret

TIM0\_OVF: ; Timer0 Overflow Handler

clr temp

out PORTB,temp

mov YL,AbegL

mov YH,AbegH

add YL,cnt

clr temp

adc YH,temp

push YL

push YH

lds temp,(Addr\_after\_real\_image+0)

sub YL,temp

lds temp,(Addr\_after\_real\_image+1)

sbc YH,temp

brlo No\_change\_pointer

ldi temp,low(Image)

add YL,temp

ldi temp,high(Image)

adc YH,temp

pop temp

pop temp

rjmp Out\_data

No\_change\_pointer:

pop YH

pop YL

Out\_data:

ld temp,Y

out PORTA,temp

ldi temp,0b00000001 ; \* ldi temp,0b10000000

cpi cnt,0

breq No\_shift

mov cnt\_temp,cnt

Shift:

lsl temp ; \* lsr temp

dec cnt\_temp

brne Shift

No\_shift:

out PORTB,temp

inc cnt

cpi cnt,8

brne No\_change\_cnt

clr cnt

No\_change\_cnt:

reti

TIM1\_OVF: ; Timer1 Overflow Handler

adiw AbegL,1

lds temp,(Addr\_after\_real\_image+0)

cp AbegL,temp

brne No\_change\_Abeg

lds temp,(Addr\_after\_real\_image+1)

cp AbegH,temp

brne No\_change\_Abeg

ldi AbegL,low(Image)

ldi AbegH,high(Image)

No\_change\_Abeg:

reti

EXT\_INT0: ; IRQ0 Handler

EXT\_INT1: ; IRQ1 Handler

TIM2\_COMP: ; Timer2 Compare Handler

TIM2\_OVF: ; Timer2 Overflow Handler

TIM1\_CAPT: ; Timer1 Capture Handler

TIM1\_COMPB: ; Timer1 CompareB Handler

TIM1\_COMPA: ; Timer1 CompareA Handler

SPI\_STC: ; SPI Transfer Complete Handler

USART\_RXC: ; USART RX Complete Handler

USART\_UDRE: ; UDR Empty Handler

USART\_TXC: ; USART TX Complete Handler

ADC\_C: ; ADC Conversion Complete Handler

EE\_RDY: ; EEPROM Ready Handler

ANA\_COMP: ; Analog Comparator Handler

TWSI: ; Two-wire Serial Interface Handler

EXT\_INT2: ; IRQ2 Handler

TIM0\_COMP: ; Timer0 Compare Handler

SPM\_RDY: ; SPM Ready Handler

reti

Out\_String:

.db ' ', 'n', 'u', 'o', 's', '.', 'e', 'd', 'u', '.', 'u', 'a', 0

Symbol\_To\_Image:

.include "font\_8x8.inc"

; RAM

.dseg

Real\_string\_length:

.byte 1

Addr\_after\_real\_image:

.byte 2

Image:

.byte (8\*String\_max\_length)

**Вывод:** в данной лабораторной работе мы научились реализовать динамическую индикацию на базе светодиодной матрицы 8x8 а также реализовать эффект бегущей строки с частотой около 2 Гц.