**3 ЛАБОРАТОРНАЯ РАБОТА №3**

**ДИНАМИЧЕСКИЙ СЕМИСЕГМЕНТНЫЙ СВЕТОДИОДНЫЙ ИНДИКАТОР**

**Цель работы:** Ознакомиться с функциональными возможностями и внутренней структурой отладочного стенда EV8031/AVR. Изучить внутреннюю организацию динамического семисегментного светодиодного индикатора, научиться отображать информацию на нём.

**3.1 Краткие теоретические сведения**

При построении подсистем отображения информации различают два подхода – динамическая и статическая схемы построения индикации.

Сущность динамической индикации заключается в поочередном циклическом подключении каждого индикатора HLi к источнику информации через общую шину данных (см. рисунок 3.1). Перебор индикаторов должен производиться достаточно быстро, в этом случае за счёт инерции человеческого зрения изображение будет казаться цельным и немигающим.

Выбор индикатора осуществляется дешифратором DA. В регистре RD хранится цифровой код, предназначенный для отображения. В регистре RA хранится адрес индикатора.

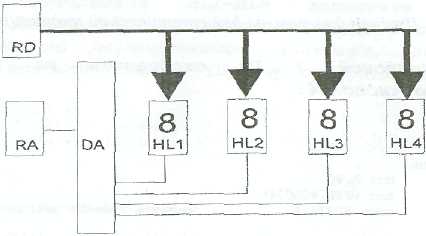


Рисунок 3.1 – Структурная схема динамической индикации

RD – регистр данных, предназначен для временного хранения семисегментного кода отображаемого символа (цифры или буквы).

RA – регистр адреса, предназначен для временного хранения номера индикатора, на котором отображаются данные.

DA – дешифратор, предназначенный для выработки сигнала выбора индикатора HLi, в зависимости от заданного в двоичном коде номера индикатора на входе.

HL1 – HL4 – семисегментные светодиодные индикаторы.

При таком включении значительно уменьшаются аппаратные затраты. Но необходимо обеспечить достаточное время свечения одного индикатора, для того чтобы не уменьшалась яркость. Также, как было указано выше, необходимо обеспечить такую частоту перебора индикаторов, чтобы не было заметно мерцаний. Преимущества такого способа отображения информации заметны при количестве разрядов индикации больше 5.

В учебно-отладочном стенде EV8031/AVR динамическая индикация реализована на плате расширения с помощью одного четырехразрядного семисегментного индикатора HL2. Управление динамической индикацией осуществляется с помощью порта В микросхемы системного контроллера, сигналы выбора соответствующего знакоместа индикатора поступают по линиям 0 и 1 порта С микроконтроллера (РСО и РС1 соответственно) к дешифратору адреса DD3.

**3.2 Порядок выполнения работы**

3.2.1 Запустите IDE AVR Studio 4.

3.2.2 Активируйте создание нового проекта посредством выбора пункта New Project из пункта Project главного меню AVR Studio 4.

3.2.3 Выберите тип создаваемого проекта: проект будет создаваться на языке ассемблер (пункт Atmel AVR Assembler) или на языке С (пункт AVR GCC).

3.2.4 Ниже, в поле Location, укажите путь по которому будет располагаться проект. Для удобства дальнейших манипуляций с проектом необходимо сохранить его в отдельном каталоге (рекомендуется D:\AVR\_projects).

3.2.5 Указать имя проекта (на английском языке) в поле Project name.

3.2.6 Принудительно (если это не сделано самой AVR Studio 4) установить флажок Create initial file.

3.2.7 В появившемся окне Debug platform выбрать пункт AVR Simulator. В окне Device выбрать пункт ATmega8515. Выбор любых отличающихся от приведенных выше пунктов в указанных окнах приведёт к дальнейшим ошибкам и невозможности выполнения лабораторной работы!

3.2.8 Нажать кнопку Finish для завершения создания проекта.

3.2.9 В появившемся окне написать программу на языке С или ассемблер с учётом варианта задания, который указан в таблице 3.1.

Таблица 3.1 – Варианты заданий

|  |  |
| --- | --- |
| № | Описание задания |
| 10 | На динамическом семисегментном индикаторе реализовать "бегущую" строку, состоящую из десятичных цифр. Для этого организовать отображение информации по следующему алгоритму: 9876, 8765, 7654, 6543, 5432, 4321, 3210, 2109, 1098, 0987. Задержка на отображение одного состояния равна 1,5 с и организовывается программно. Семисегментные коды символов (или знакогенератор) задаются в виде констант (в начале программы), которые необходимо сохранять в SRAM (ОЗУ), начиная с адреса 0060h. |

3.2.10 Произвести компиляцию проекта посредством выбора пункта Build из одноимённого меню (либо нажать кнопку F7).

3.2.11 При наличие сообщений об ошибках или предупреждениях вернуться к предыдущему пункту и внести необходимые исправления.

3.2.12 Проверьте наличие подключения USB-кабеля программатора к соответствующему разъёму системного блока.

3.2.13 Для загрузки откомпилированного проекта в микроконтроллер выберите пункт меню Tools -> Program AVR -> Connect...

3.2.14 В появившемся окне Platform выберите пункт AVRISP mkII. В окне Port выберите пункт USB. Выбор любых отличающихся от приведенных выше пунктов в указанных окнах приведёт к невозможности программирования микроконтроллера!

3.2.15 Нажмите на кнопку Connect...

3.2.16 В появившемся окне сразу перейдите на вкладку Program. В поле Location укажите путь к файлу с расширением \*.hex.

3.2.17 Нажмите кнопку Program. При этом происходит программирование (полное стирание и запись) Flash-памяти (памяти программ) микроконтроллера.

3.2.18 Визуально оцените правильность работы динамического семисегментного светодиодного индикатора.

**3.3 Результаты выполнения работы**

.include "m8515def.inc"

.def segm = r16 ;ðåãèñòð âðåìåííîãî õðàíåíèÿ

.def nmb = r19 ;íîìåð çíàêîìåñòà

.def counter = r17 ;ñ÷åò÷èê öèêëà â ïîäïðîãðàììå ãåíåðàöèè çàäåðæêè

.def counter1 = r18 ;ñ÷åò÷èê öèêëà äëÿ îòîáðàæåíèÿ ñåãìåíòîâ

.def counter\_state = r20

.def temp = r21 ;âðåìåííûå äàííûå

.def long\_delay\_low = r24 ;ìëàäøèé áàéò ñ÷åò÷èêà äëèííîé çàäåðæêè

.def long\_delay\_high = r25 ;ñòàðøèé áàéò ñ÷åò÷èêà äëèííîé çàäåðæêè

.EQU dyn\_7seg\_data = 0x8001

.EQU synthes\_5x7\_row\_dyn\_7seg\_control = 0x8002

.EQU delay = 75

.EQU nmb9 = 0b11110110

.EQU nmb8 = 0b11111100

.EQU nmb7 = 0b11100000

.EQU nmb6 = 0b10111110

.EQU nmb5 = 0b10110110

.EQU nmb4 = 0b01100110

.EQU nmb3 = 0b11110010

.EQU nmb2 = 0b11011010

.EQU nmb1 = 0b01100000

.EQU nmb0 = 0b11111100

;\*\*\*\*\* Íà÷àëî ïðîãðàììû \*\*\*\*\*

.CSEG ;îïðåäåëÿåì íà÷àëî ñåãìåíòà êîäà

.ORG 0x0000 ;îïðåäåëÿåì àäðåñ íà÷àëà ñåãìåíòà êîäà â ïàìÿòè ïðîãðàìì

;\*\*\* Âåêòîð ïðåðûâàíèé êîíòðîëëåðà \*\*\*

rjmp Init ; âåêòîð ïðåðûâàíèÿ ïî ñáðîñó

reti;rjmp EXT\_INT0 ; IRQ0 Handler

reti;rjmp EXT\_INT1 ; IRQ1 Handler

reti;rjmp TIM1\_CAPT ; Timer1 Capture Handler

reti;rjmp TIM1\_COMPA ; Timer1 Compare A Handler

reti;rjmp TIM1\_COMPB ; Timer1 Compare B Handler

reti;rjmp TIM1\_OVF ; Timer1 Overflow Handler

reti;rjmp TIM0\_OVF ; Timer0 Overflow Handler

reti;rjmp SPI\_STC ; SPI Transfer Complete Handler

reti;rjmp USART\_RXC ; USART RX Complete Handler

reti;rjmp USART\_UDRE ; UDR0 Empty Handler

reti;rjmp USART\_TXC ; USART TX Complete Handler

reti;rjmp ANA\_COMP ; Analog Comparator Handler

reti;rjmp EXT\_INT2 ; IRQ2 Handler

reti;rjmp TIM0\_COMP ; Timer0 Compare Handler

reti;rjmp EE\_RDY ; EEPROM Ready Handler

reti;rjmp SPM\_RDY ; Store Program memory Ready

Init:

ldi temp,low(RAMEND)

out SPL,temp

ldi temp,high(RAMEND)

out SPH,temp ;óñòàíîâêà SP íà ïîñëåäíèé àäðåñ SRAM

sbi ACSR,7 ;îòêëþ÷åíèå ïèòàíèÿ àíàëîãîâîãî êîìïàðàòîðà

ldi temp, 0b10000000

out MCUCR, temp

ldi YL, low(dyn\_7seg\_data)

ldi YH, high(dyn\_7seg\_data)

ldi XL, low(synthes\_5x7\_row\_dyn\_7seg\_control)

ldi XH, high(synthes\_5x7\_row\_dyn\_7seg\_control)

ldi ZH, high(numbers)

ldi ZL, low(numbers)

ldi r16, nmb9

st Z+, r16

ldi r16, nmb8

st Z+, r16

ldi r16, nmb7

st Z+, r16

ldi r16, nmb6

st Z+, r16

ldi r16, nmb5

st Z+, r16

ldi r16, nmb4

st Z+, r16

ldi r16, nmb3

st Z+, r16

ldi r16, nmb2

st Z+, r16

ldi r16, nmb1

st Z+, r16

ldi r16, nmb0

st Z+, r16

ldi r16, nmb9

st Z+, r16

ldi r16, nmb8

st Z+, r16

ldi r16, nmb7

st Z+, r16

Infinite\_loop:

ldi ZH, high(numbers)

ldi ZL, low(numbers)

ldi nmb, 0

ldi counter1, 0

ldi counter\_state, 0

state\_loop:

cpi counter\_state, 10

breq Infinite\_loop

cpi counter1, 75

breq end\_state

nmb\_loop:

cpi nmb, 4

breq end\_nmb\_loop

ld segm, Z+

st Y, segm

st X, nmb

inc nmb

rcall long\_delay

rjmp nmb\_loop

end\_nmb\_loop:

ldi nmb, 0

sbiw ZH:ZL, 4

inc counter1

rjmp state\_loop

end\_state:

ldi counter1, 0

adiw ZH:ZL, 1

inc counter\_state

rjmp state\_loop

long\_delay:

ldi long\_delay\_low,0x2E ;çàãðóçêà â ðåãèñòðîâóþ ïàðó êîýôôèöèåíòà çàäåðæêè

ldi long\_delay\_high,0x00 ;(002Eh), ýòî áóäåò çàäåðæêà íà 4,99 ìñ

long\_loop: ;òåëî öèêëà çàíèìàåò 796 + 2 + 2 = 800 òàêòîâ

rcall short\_delay ;êîðîòêàÿ çàäåðæêà

sbiw long\_delay\_high:long\_delay\_low,0b00000001 ;âû÷èòàíèå èç ïàðû ÷èñëà 1 (äåêðåìåíò äëèííîãî ñ÷åò÷èêà)

brne long\_loop ;åñëè íå 0,ïîâòîðèòü öèêë

ret ;âîçâðàò â îñíîâíóþ ïðîãðàììó

short\_delay: ;âñÿ ïîäïðîãðàììà çàíèìàåò ðîâíî 796 òàêòîâ âìåñòå ñ rcall è ret

nop

ldi counter,0xC5 ;ñ÷åò÷èê öèêëà

short\_loop:

nop

dec counter

brne short\_loop ;êîìàíäà âåòâëåíèÿ ïî ôëàãó íóëÿ (çàöèêëèâàíèå)

ret ;âîçâðàò â îñíîâíóþ ïðîãðàììó

.DSEG

.ORG SRAM\_START

numbers:

.EXIT ;êîíåö ïðîãðàììû

**3.4 Особенности IDE AVRStudio выявленные в ходе выполнения лабораторной работы**

В ходе лабораторной работы никаких новых особенностей IDE AVRStudio не было выявлено.

**Выводы**

В ходе лабораторной работы были изучены принципы отображения информации на динамическом семисегментом индикаторе в учебно-отладочном стенде EV8031/AVR. Была написана программа, которая выводит бегущую строку состоящую из цифр на динамический семисегментный индикатор, считывая при этом данные из SRAM. Преимуществом динамического семисегментного индикатора является уменьшения потребления тока, но при этом усложняется алгоритм вывода информации.