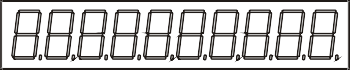
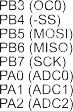
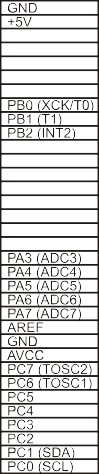
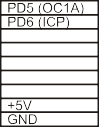
Лабораторная работа № 9 **Подключение модуля 7-сегментного жидкокристаллического индикатора**



Цель работы:

1. Изучение принципов подключения внешних устройств к МК

ATmega8535.

1. Программирование работы модуля жидкокристаллического индика- тора (ЖКИ).

Оборудование и программное обеспечение:

1. Лабораторный макет MK8535 в составе: модуль базовый МБ8535 +

модуль сменный МС03.

1. Кабель соединительный для программирования МК.
2. Персональный компьютер.
3. Интегрированная среда программирования CodeVisionAVR.

# Подготовка к работе

Функциональная схема лабораторного макета приведена на рис. 62. После изучения описания интерфейса МТ10Т7-7 целесообразно составить процедуры для передачи данных в модуль ЖКИ.

Создать проект в CodeWizardAVR и настроить линии 0…6 порта

A на вывод.

Потребуется таблица программного дешифратора, подобная той, что использовалась в лабораторной работе № 5 «Использование дина- мической индикации». С учетом того, как модуль ЖКИ подключен к порту A, а также того, что сегменты высвечиваются записью 1, в табл. 24 приведены комбинации бит для символов «0…9, пробел, минус».

Рис. 62. Функциональная схема лабораторного макета. Для модуля МС03 показана только та часть схемы, которая используется в данной работе

Комбинации бит для изображения символов

*Таблица 24*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Символ | 0 | 1 | 2 | 3 | 4 |
| 2-е зн-е | **11101110** | **01100000** | **00101111** | **01101101** | **11100001** |
| 16-е зн-е | **0xEE** | **0x60** | **0x2F** | **0x6D** | **0xE1** |
| Символ | 5 | 6 | 7 | 8 | 9 |
| 2-е зн-е | **11001101** | **11001111** | **01101000** | **11101111** | **11101101** |
| 16-е зн-е | **0xCD** | **0xCF** | **0x68** | **0xEF** | **0xED** |
| Символ | пробел | минус |  |  |  |
| 2-е зн-е | **00000000** | **00000001** |  |  |  |
| 16-е зн-е | **0x00** | **0x01** |  |  |  |

Ниже приведен пример программы для МК ATmega8535, где на ЖКИ выводятся символы «0123456789». В нем опущены те строки, автоматически сформированные CodeWizardAVR, которые не относятся непосредственно к данному примеру.

Коды 10 символов, предназначенных для вывода на ЖКИ, следует заносить в массив **LCDStr[10]**. Примем следующую кодировку сим- волов (табл. 25):

*Таблица* 25

Коды символов

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Символ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | пробел | минус |
| Код | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

Процедура **ShowLCD()**обеспечивает передачу 10 символов в мо- дуль ЖКИ. Наборы бит, необходимые для изображения символов, со- держатся в массиве **BitSet[12]**. В качестве индексов этого массива используются коды символов из **LCDStr[10]**.

Процедура **ClrLCD()**очищает табло индикатора путем заполне- ния его пробелами. Для разблокировки шины ЖКИ служит процедура **UnlockLCD()**.

**#include <mega8535.h> #include <delay.h>**

**unsigned char LCDStr[10] = {0,1,2,3,4,5,6,7,8,9};**

**// Разблокировка шины**

**void UnlockLCD(void)**

**{**

**PORTA.5 = 0; // ^WR2 = 0**

**PORTA.4 = 0; // WR1 = 0**

**PORTA.6 = 0; // A0 = 0**

**PORTA |= 0x0F; // адрес = 0x0F**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**PORTA.6 = 1; // A0 = 1**

**PORTA.0 = 1; // DB0 = 1**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**}**

# // Очистка индикатора

**void ClrLCD(void)**

**{**

**unsigned char i;**

**UnlockLCD(); // Разблокировка шины**

**PORTA.5 = 0; // ^WR2 = 0**

**PORTA.4 = 0; // WR1 = 0**

**// Запись 10 пробелов с адреса 0 PORTA.6 = 0; // A0 = 0**

**PORTA &= 0xF0; // адрес = 0**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**PORTA.6 = 1; // A0 = 1**

**PORTA &= 0xF0; // DB0..3 = 0**

**for (i=1; i<=20; i++)**

**{**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**}**

**}**

# void ShowLCD(void)

**{**

**static unsigned char BitSet[12]=**

**{0xEE,0x60,0x2F,0x6D,0xE1, // 0..4**

**0xCD,0xCF,0x68,0xEF,0xED, // 5..9**

**0x00,0x01 }; // пробел, минус**

**unsigned char i;**

**UnlockLCD(); // Разблокировка шины**

**PORTA.5 = 0; // ^WR2 = 0**

**PORTA.4 = 0; // WR1 = 0**

**PORTA.6 = 0; // A0 = 0**

**PORTA &= 0xF0; // адрес = 0**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**PORTA.6 = 1; // A0 = 1**

**for (i=0; i<10; i++)**

**{**

# void main(void)

**{**

# PORTA = 0x00; DDRA = 0x7F; // Port A init. ClrLCD();

**ShowLCD(); while (1);**

**}**

# Задание

1. Создать проект и реализовать рассмотренный выше пример программы.
2. Создать новый проект и организовать вывод на индикатор значений последовательного счета (1 раз/сек).
3. Создать новый проект для вывода на индикатор показаний часов в формате ЧАСЫ–МИНУТЫ–СЕКУНДЫ.

**PORTA &= 0xF0; // DB0..3 = 0**

**// DB0..3 = младшая тетрада**

**PORTA |= BitSet[LCDStr[i]] & 0x0F;**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**PORTA &= 0xF0; // DB0..3 = 0**

**// DB0..3 = старшая тетрада**

**PORTA |= BitSet[LCDStr[i]] >> 4;**

**PORTA.4 = 1; // Формирование**

**delay\_us(1); // импульса**

**PORTA.4 = 0; // WR1**

**delay\_us(1); // задержка 1 мкс**

**}**

**}**

# ОРГАНИЗАЦИЯ КЛАВИАТУРЫ ДЛЯ МИКРОКОНТРОЛЛЕРА

Если устройство на основе МК имеет множество управляющих кнопок (клавиатуру), то для их непосредственного подключения может не хватить выводов портов. В таких случаях клавиатуру можно организовать как матрицу кнопок, объединённых в N строк и M столбцов (рис. 63).



Рис. 63. Матричная организация клавиатуры

Теперь при расходе линий порта равном N+M число кнопок мо- жет достигать N×M. Чем больше кнопок, тем эффектнее экономия. Например, 101 кнопка клавиатуры персонального компьютера организо- вана в матрицу 8×16. Определить, какая клавиша нажата, можно перио- дическим опросом строк или столбцов. Способ основан на том, что МК опрашивает матрицу с частотой, много большей, чем может нажимать и отпускать кнопки человек.

На рис. 63 показан пример матричной организации клавиатуры из 12 кнопок, объединённых в 3 строки по 4 кнопки (4 столбца). Периоди- ческий опрос матрицы можно организовать, например, построчно, путем выдачи «бегущего нуля» последовательно на линии PC4, PC5, PC6 и считывая при этом столбцы на линиях PC0…PC4. Если не нажата ни одна клавиша, то слово PC0…PC4 будет равно 1111 (0x0F).

Если какая-либо клавиша нажата, то через замкнутый контакт столбец получит значение 0, и считанное слово PC0…PC4 будет отлично от 0x0F. В этом случае, зная номер текущей строки, и определив, какая

из линий PC0…PC4 не равна 1, можно выяснить номер нажатой клави- ши. Например, если при PC5 = 0 с линий PC0…PC4 считано 1101 (0x0D), то нажата была клавиша S7.

«Дребезг» контактов, как правило, устраняют программно, путем повторного чтения входов через определенную временную задержку.

Для того чтобы предотвратить протекание опасных токов при одновременном нажатии нескольких кнопок в столбце, строки подклю- чают к МК последовательно через развязывающие резисторы или диоды. С этой же целью можно использовать и другой метод опроса, при кото- ром все неактивные горизонтальные шины, кроме шины «бегущего нуля», программно назначаются входами.

Другой пример подключения клавиатуры с использованием мень- шего числа линий порта приведен на рис. 64. Если в схеме на рис. 63 для матрицы 4×3 требуются 7 линий порта, то в этом примере, за счет более сложной программной обработки, понадобится всего 4.



Рис. 64. 4-проводное подключение клавиатуры

Видно, что в такой схеме требуются дополнительные внешние элементы – диоды. При опросе матрицы поочередно на одной из четы- рех линий устанавливают 0, а другие программируют как входы. При этом они обязательно должны быть подтянуты резисторами к высокому уровню (в этом примере использованы внутренние резисторы). По ком- бинации на входах вычисляется нажатая кнопка.

Большинство современных микроконтроллеров содержат анало- го-цифровой преобразователь (АЦП). С использованием АЦП можно

построить схему клавиатуры, требующую для подключения всего лишь две линии. На рис. 65 показан пример 5-кнопочной клавиатуры.



*Таблица 26*

Идентификация кнопок

|  |  |
| --- | --- |
| Напряжение | Кнопка |
| 5,00 В | Нет  нажатых |
| 0,00 В | S1 |
| 0,77 В | S2 |
| 1,12 В | S3 |
| 1,45 В | S4 |
| 1,80 В | S5 |

Рис. 65. Клавиатура на основе резистивного делителя

Кнопки подключены к отводам резистивного делителя R1…R5 и при нажатии разных кнопок с делителя на вход АЦП поступают, соот- ветственно, разные уровни напряжения (см. табл. 26). Напряжения сле- дует оценивать с некоторым допуском, с учетом отклонений номиналов резисторов.