МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Методические указания

к выполнению лабораторных работ по дисциплине «Системотехнические методы и средства отображения информации» для студентов специальности 7.090803 ЭлС (Электронные системы)

Рассмотрено
на заседании кафедры
электронной техники.
Протокол №3 от17.10.2007г
Утверждено на заседании
учебно-издательского совета
ДОННТУ протокол№263
Пр. №2 от12. 12. 2007г

Донецк, 2007

Методические указания к выполнению лабораторных работ по дисциплине «Системотехнические методы и средства отображения информации» для студентов специальности 7.090803 ЭлС (Электронные системы)/ В.В.Калашников, Д.Н. Кузнецов.– ДонНТУ, Донецк, 2007. – 62 с.

Приведены цель и методика выполнения лабораторных работ, необходимые теоретические и справочные данные к их выполнению, порядок выполнения, требования к содержанию отчетов и контрольные вопросы для самоконтроля знаний студентов.

Составители: ст. преп. В.В. Калашников
доцент Д.Н.Кузнецов

Рецензент:

СОДЕРЖАНИЕ

ОСНОВНЫЕ ПОЛОЖЕНИЯ

Методические указания предназначены для студентов специальности «Электронные системы» и являются руководством к лабораторным работам по дисциплине «Системотехнические методы и средства отображения информации».

Целью этого курса является закрепление лекционного материала, освоение интегрированной среды разработки и отладки программного обеспечения ProView, освоение программы имитационного моделирования микропроцессорных устройств Proteus и специализированного лабораторного стенда EV8031/AVR.

Лабораторные работы проводятся в компьютерной аудитории. При их выполнении используются методы математического и имитационного моделирования с использованием пакетов прикладных программ ProView и Proteus, а также реальный лабораторный стенд EV8031/AVR.

Лабораторные работы начинаются с короткого теоретического материала по основам работы исследуемого устройства. Далее излагаются методика выполнения работы, справочные данные, требования к содержанию отчета и перечень контрольных вопросов для самоконтроля усвоения материала.

Подготовка к лабораторной работе должна выполнятся по лекционному материалу и дополнительной литературе.

По каждой выполненной работе составляется отчет. На титульном листе указывается дисциплина, тема лабораторной работы и автор отчета. Отчет должен содержать цель работы, вариант индивидуального задания, принципиальную схему исследуемого устройства, результаты исследований. В конце следует анализ полученных результатов и вывод по работе.

Лабораторная работа №1.

ЦИФРОВАЯ СТАТИЧЕСКАЯ ИНДИКАЦИЯ.

Цель работы: изучение принципа работы цифровой статической индикации.

Теоретические сведения.

Многие МК-устройства, требуют наличия только простейшей индикации типа да/нет, вкл./выкл. Такая индикация реализуется на основе отдельных светодиодов.

Семисегментные индикаторы (ССИ) широко используются для отображения цифровой и буквенной информации. Семь отображающих элементов позволяют высвечивать 10-тичные и 16-ричные цифры, некоторые буквы русского и латинского алфавитов, а также некоторые специальные знаки. Структура ССИ и способы его подключения к МК показаны на рисунке 1. Для засветки одного сегмента большинства типов ССИ необходимо обеспечить протекание через сегмент тока 10-15 мА, при напряжении 2,0-2,5 В. Низкая нагрузочная способность МК не допускает прямого соединения с ССИ. В качестве промежуточных усилителей тока могут использоваться логические элементы серий 155, 555, 1555 или интегральные схемы преобразователей кодов для управления ССИ.

Преобразование двоичных кодов в коды для ССИ может осуществляться либо программно, либо аппаратно с использованием преобразователей К514ИД1, К514ИД2, 133ПП4, 564ИД5.

Для отображения многосимвольной информации используются линейные (однострочные) дисплеи. Такие дисплеи представляют собой «линейку», смонтированную из отдельных ССИ. Число знакомест дисплея определяется в соответствии требованиями к МК-системе.

Существует два способа организации интерфейса МК с линейным дисплеем: статический и динамический.

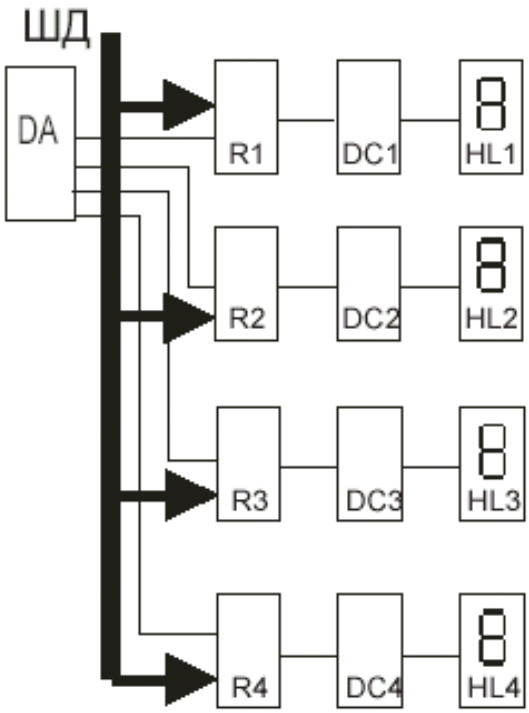
Первый способ (статический) требует наличия на входах каждого индикатора специальных буферных регистров для хранения кодов выводимых символов. С увеличением разрядности дисплея возрастает число дополнительных микросхем, а следовательно, и стоимость МК-системы.

Второй способ (динамический) основан на том, что любой световой индикатор является инерционным прибором, а человеческому глазу, отобра-

жаемая на дисплее информация, если её обновлять с частотой примерно 20 раз в секунду, представляется неизменной. Динамический способ вывода информации на дисплей требует значительно меньших аппаратных затрат, но более сложного программного обеспечения. Именно этот способ организации вывода информации получил преимущественное распространение в МК-системах.

На рисунке 1 приведена структурная схема статической индикации. В такой системе каждый индикатор подключен через собственный дешифратор ДС

к шине данных. Выборка регистров RG производится при помощи дешифратора адреса DA. Аппаратные затраты при такой организации составляют *n* пар регистров + дешифратор при n разрядов индикатора.



DA – дешифратор адреса, необходим для выборки соответствующего регистра
R1-R4- регистры, в которых временно хранится значение кода числа для отображения (соответствующий регистр выбирается DA).
DC1-DC4 – семи сегментные дешифраторы, преобразующие двоичный код в семи сегментный код.
HL1-HL4 – семи сегментные индикаторы.
ШД – шина данных, по ней осуществляется передача данных на индикацию.

Рис. 1 Структурная схема статической индикации.

На рисунке 2 приведена принципиальная схема всех видов индикации стенда EV8031/AVR(V3.2).

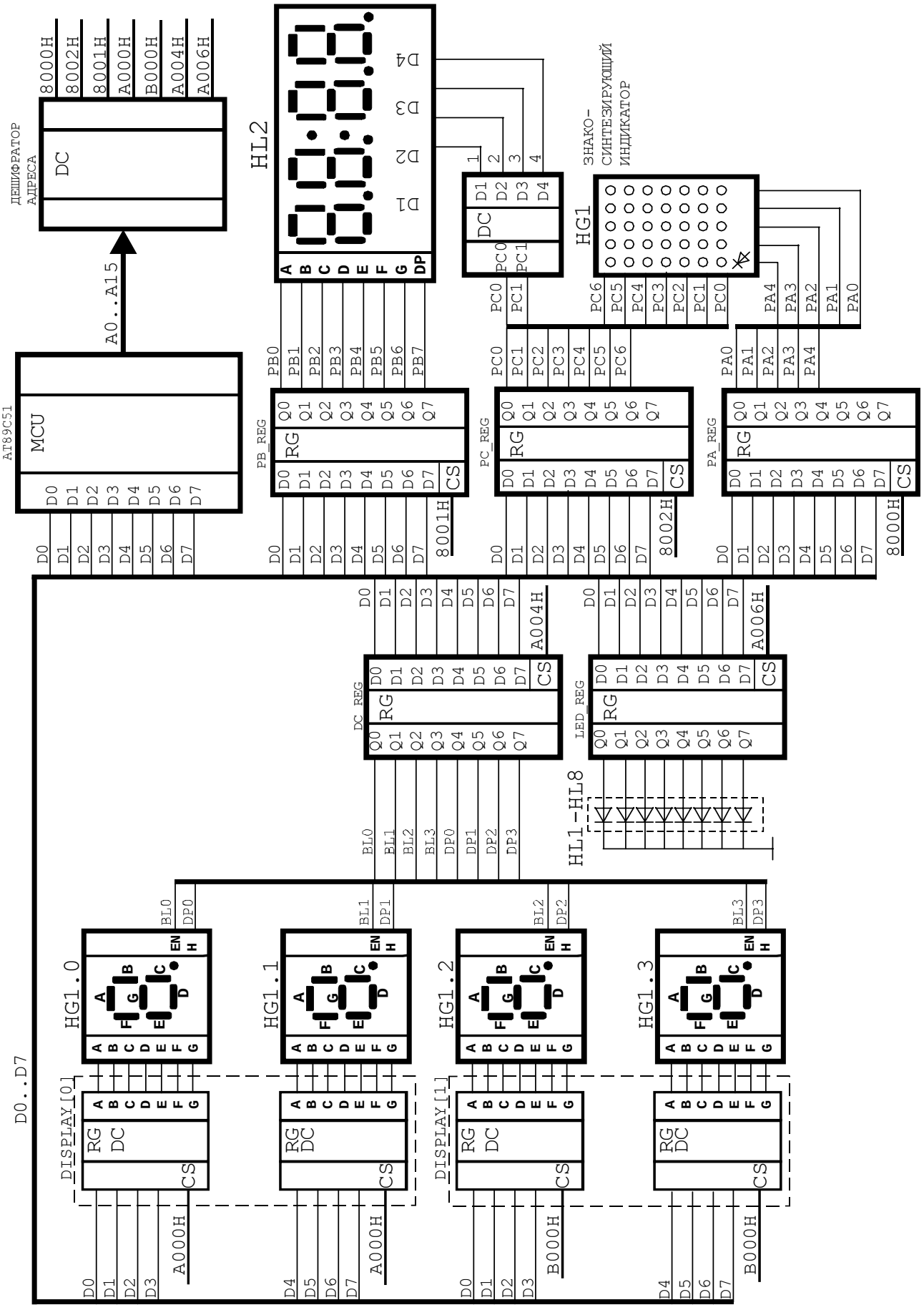


Рис 2. Принципиальная схема индикации стенда EV8031.

Табл.1 Варианты индивидуальных заданий

№	*Текст индивидуального задания*
1	Занести в регистр R4 двоично-десятичное число 0X, в регистр R6 двоично-десятичное число X0, сумму чисел отобразить на первом и втором знакоместе статической индикации .
2	Занести в регистр R3 двоично-десятичное число XX отобразить его на первом и четвертом знакоместе статической индикации.
3	Занести в регистр B двоично-десятичное число, с частотой 2 Гц выводить это число на первом и втором знакоместе статической индикации .
4	Занести в Аккумулятор двоично-десятичное число XX, в регистр R5 X0, число из Аккумулятора отобразить на первом и втором знакоместе статической индикации, число из R5 отобразить на третьем знакоместе статической индикации.
5	Занести в регистр R2 двоично-десятичное число 0X, в регистр R5 X0, сумму чисел отобразить на втором и третьем знакоместе статической индикации.
6	Занести в ячейку с адресом B0h внутренней памяти ОЭВМ двоично-десятичное число 0X, в регистр R3 число X0, сумму чисел отображать на втором и третьем знакоместе статической индикации с частотой 0,5Гц.
7	Занести в регистр R0 двоично-десятичное число XX, попеременно отображать младшую и старшую тетраду на первом и четвертом знакоместе статической индикации с частотой 1 Гц.
8	Занести в B двоично-десятичное число X0, в регистр R1 XX, число из B отображать на первом знакоместе статической индикации с частотой 1 Гц, число из R1 отображать на третьем и четвертом знакоместе статической индикации с частотой 0,5 Гц.
9	Считать значение регистра TCON и отобразить его на третьем и четвертом знакоместе статической индикации.
10	Занести в регистр R4 двоично-десятичное число 0X, в регистр R3 X0, сумму чисел отобразить на втором и третьем знакоместе статической индикации с медленным (в течение 5 сек.) затуханием этого числа.
11	Занести в Аккумулятор двоично-десятичное число X0, в регистр B 0X, сумму чисел отобразить на первом и четвертом знакоместе статической индикации.
12	Занести в регистр B двоично-десятичное число 0X, в регистр R5 X0, два разряда суммы (десятки и единицы) поочередно отображать на первом и втором знакоместе статической индикации
13	Занести в регистр R1 двоично-десятичное число 0X, отнимая от числа единицу отображать на третьем знакоместе статической индикации полученное значение до нуля с частотой 1 Гц.
14	Занести в регистр R3 двоично-десятичное число XX, в регистр R5 XX, попеременно отображать эти числа на первом и втором знакоместе статической индикации (R3) и на третьем и четвертом знакоместе статической индикации (R5).
15	Занести в регистр A двоично-десятичное число 0X, в регистр R2 X0, число из A отобразить на четвертом знакоместе статической индикации, число из регистра R2 отображать на втором знакоместе статической индикации с частотой в 0.5 Гц.
16	Занести в регистр R0 число XX, вывести на левой паре знакомест статической индикации. Через 2 секунды вывести число XX, занесенное в регистр R1. Через 2 секунды на правой паре знакомест статической индикации вывести разницу чисел занесенных в регистр R0 и R1.
17	Занести число 0X в регистр R0 и число XX в регистр R1. Вывести значение R1 на правой паре знакомест статической индикации. Затем прибавлять значение R1 к значению R0, до тех пор, пока результат не достигнет определенного, заранее заданного порога. Осуществить переход на начало программы. (Время задержки 0,5 секунд).

Пример выполнения задания.

Составить программу индикации набора цифр 16-ричной системы счисления от 0 до F. Для индикации использовать статические индикаторы по адресу 0xB000. В программе предусмотреть функцию *time* для временной задержки каждого индицируемого символа.

```
#include <reg51.h>
#define const1 0x7fff
unsigned char sim,z,i;
at 0xb000 unsigned char xdata ind;
short l;
void time()
{
        for(l=const1;l>=0;l--);
}
void main()
{
        sim=0;
        for(i=0;i<=15;i++)
        {
                z=sim;
                z<<=4;
                z|=sim;
                sim++;
                ind=z;
                time();
        }
        while(1);
}
```

Модель схемы статической индикации, составленная в пакете Proteus, представлена на рисунке 3.

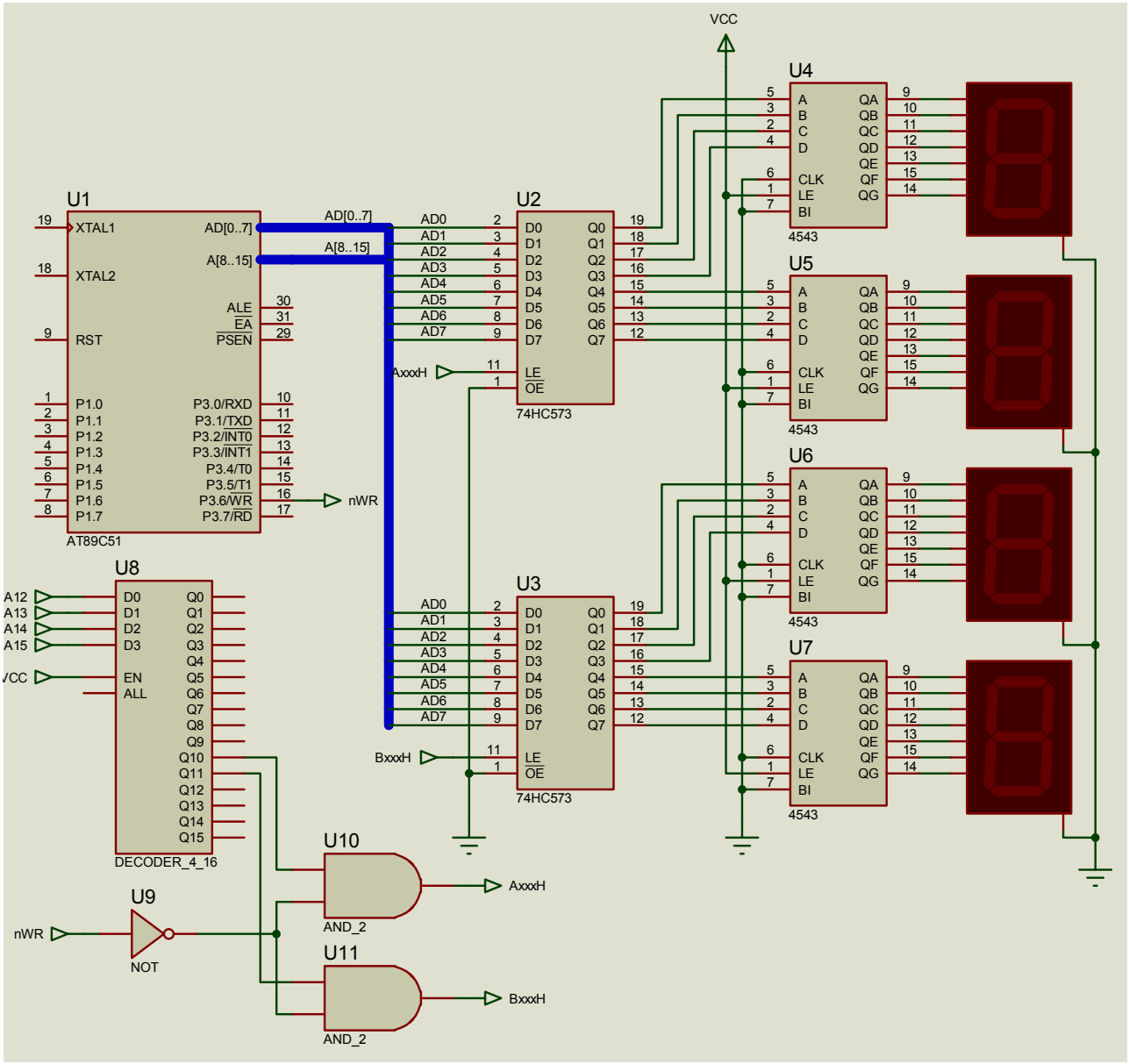


Рис. 3 Модель схемы статической индикации

Порядок выполнения работы.

1. Написать программу на языке Си, реализующую поставленную задачу.
2. Выполнить компиляцию программы и получить файл с расширением *.hex.
3. Проверить работоспособность программы в симуляторе Proteus.
4. Запрограммировать лабораторный стенд EV8031/AVR и проверить работоспособность программы на стенде.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.

2. Цель и задачи работы.
3. Из общей схемы индикации выбрать статическую и привести в отчете.
4. Тексты задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1.Привести структурную схему одного разряда статической цифровой индикации.

2. Сколько разрядов статической индикации на стенде EV8031 ?

3. Схемотехнические отличия статической индикации от динамической.

4. Запишите коды цифрового индикатора для цифр 0, 1, 8, F.

6. Запишите программу на языке программирования СИ для выдачи кода 0x0F на статическую цифровую индикацию.
7. Запишите программу на языке программирования СИ для выдачи кода 0x0A0C на статическую цифровую индикацию.

Лабораторная работа №2.

ЦИФРОВАЯ ДВОИЧНАЯ ИНДИКАЦИЯ.

Цель работы: изучение принципов построения схем индикации на светодиодах и программ управления ими.

Краткие теоретические сведения.

Простейшими приборами отображения двоичной информации в цифровых устройствах являются светодиоды. В полупроводниковых светодиодах используется свойство p-n переходов излучать свет в видимой части спектра при протекании через него прямого тока ($I_{пр}$=5-20мА, $U_{пр}$=2-3В). Варианты включения индикаторов приведены на рисунке 1.

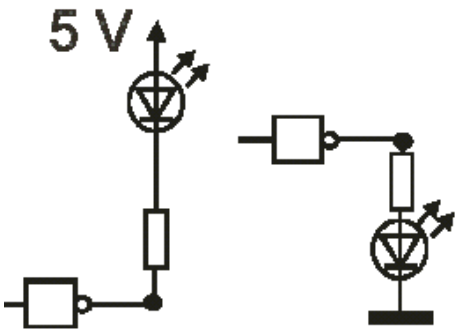


Рис.1 Варианты включения светодиодов

На рисунке 2 приведена схема двоичной индикации стенда на светодиодах HL1-HL8.

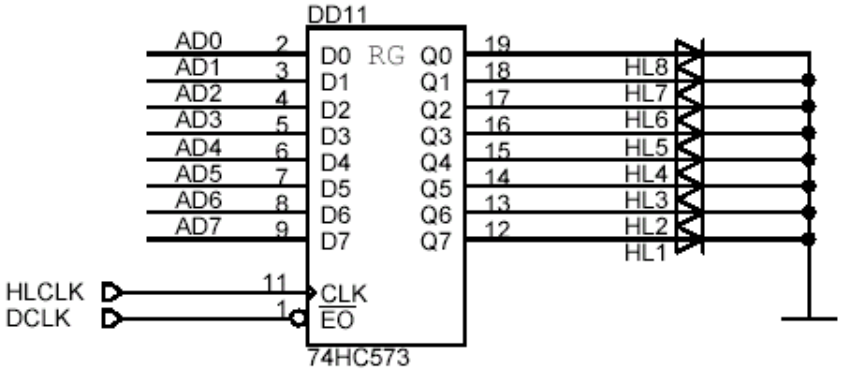


Рис.2 Принципиальная схема двоичной индикации HL1-HL8.

Адрес регистра, к которому подключены светодиоды – 0xA006.

Задание для выполнения лабораторной работы.

Запрограммировать заданную последовательность переключения светодиодов A1,A2,A3,A4,A5,A1,итд. Длительность пребывания в состоянии горения *T*.

Табл.1 Обозначение светодиодов

Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0

Табл.2 Варианты заданий

Вприант№	A1	A2	A3	A4	A5	T, сек
1	Y1	Y2	Y4	Y6	Y7	1
2	Y7	Y6	Y4	Y3	Y1	2
3	Y2	Y3	Y5	Y6	Y7	3
4	Y3	Y4	Y6	Y6	Y7	4
5	Y1	Y3	Y5	Y7	Y0	5
6	Y7	Y6	Y5	Y3	`Y1	1
7	Y0	Y7	Y6	Y6	Y4	2
8	Y0	Y7	Y2	Y6	Y4	3
9	Y7	Y6	Y0	Y1	Y2	4
10	Y6	Y0	Y7	Y1	Y3	5
11	Y4	Y2	Y6	Y0	Y7	1
12	Y3	Y1	Y0	Y7	Y5	2
13	Y2	Y4	Y7	Y0	Y1	3
14	Y0	Y2	Y6	Y3	Y4	4
15	Y6	Y7	Y0	Y1	Y2	5

Пример выполнения задания.

Ниже приведен текст программы на СИ бегущей единицы на светодиодах HL1-HL8.

```
#include <reg51.h>
#define const2 0x7fff
unsigned char cod,i;
at 0xA006 unsigned char xdata ind;
short l;
void main()
{
m2:
cod=0x00;
i=0;
cod=0x01;
m1:          if(i<=7)
             {
                    i++;
                    ind=cod;
                    cod<<=1;
                     for(l=const2;l>=0;l--);
             goto m1;
             }
  goto m2;
  }
```

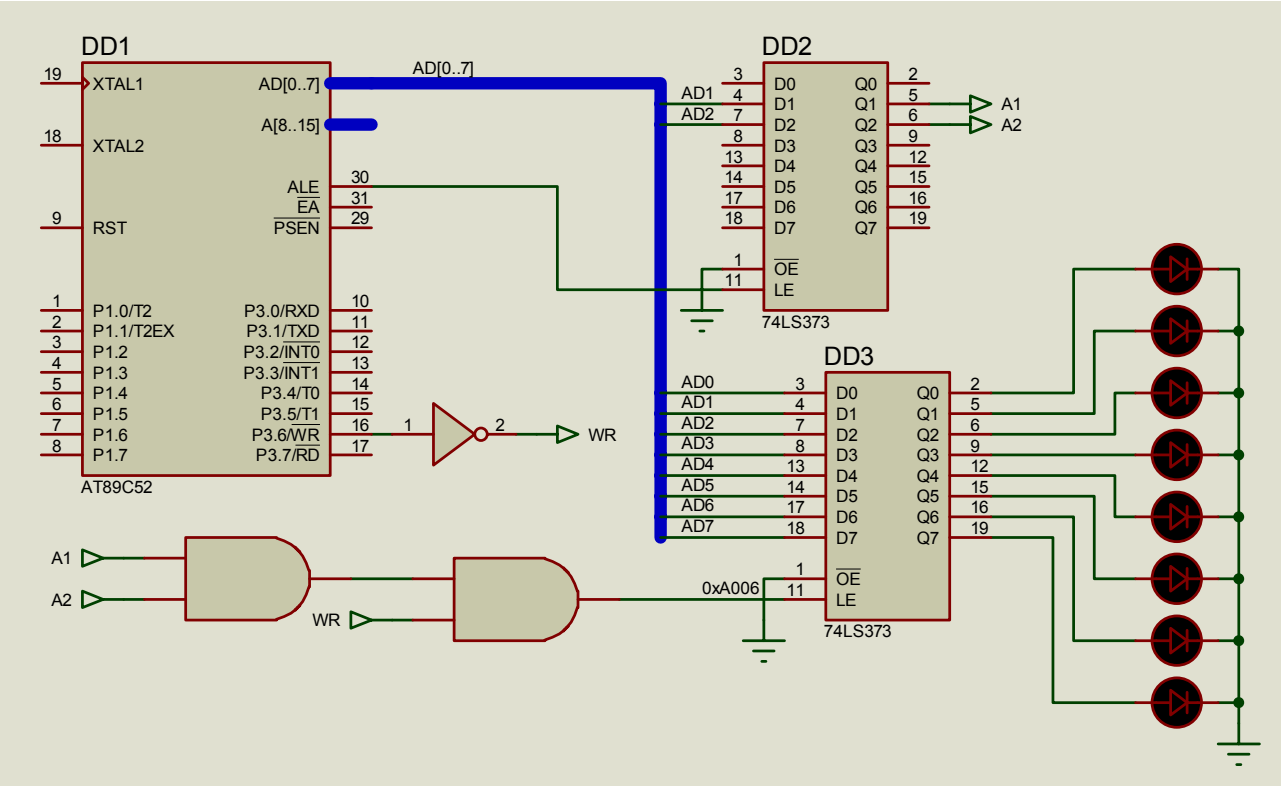


Рис.3 Модель схемы цифровой двоичной индикации в Proteuse

Порядок выполнения работы.

5. Написать программу на языке Си, реализующую поставленную задачу.
6. Выполнить компиляцию программы и получить файл с расширением *.hex.
7. Проверить работоспособность программы в симуляторе Proteus.
8. Запрограммировать лабораторный стенд EV8031/AVR и проверить работоспособность программы на стенде.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему двоичной индикации EV8031.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1. Привести схему включения единичного индикатора для индикации 0 на выходе логического элемента.
2. Привести схему включения единичного индикатора для индикации 1 на выходе логического элемента.
3. Зачем нужны сопротивления в цепи включения светодиодов ? Как рассчитать их значение?
4. Ток и напряжение необходимые для нормальной работы светодиода.
5. Сема включения светодиодов на стенде EV8031 для индикации двоичной информации.
6. Написать программу на языке СИ для выдачи кода 0x0F на двоичную индикацию.

Лабораторная работа №3.

ЦИФРОВАЯ ДИНАМИЧЕСКАЯ ИНДИКАЦИЯ.

Цель работы: изучение принципа динамической индикации и схем построения линейных дисплеев семисегментных светодиодных индикаторов.

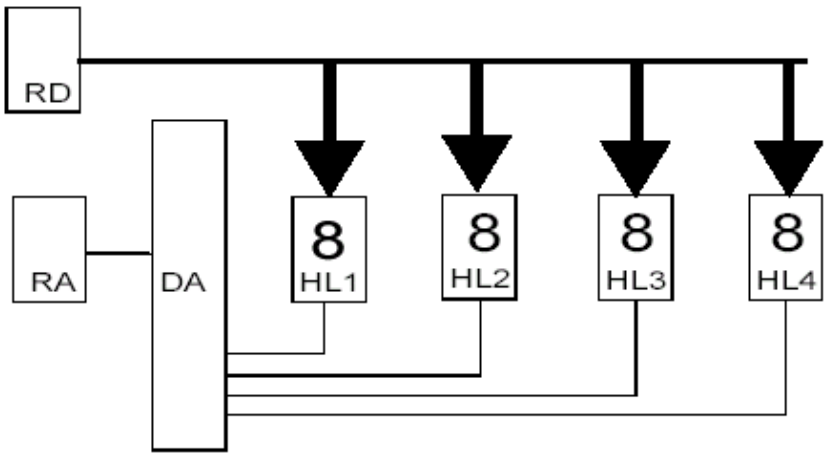
Краткие теоретические сведения.

Существует два способа организации интерфейса МК с линейным дисплеем: статический и динамический.

Первый требует наличия на входах каждого индикатора специальных буферных регистров для хранения кодов выводимых символов. Естественно, что с увеличением разрядности дисплея возрастает число дополнительных микросхем, а следовательно, и стоимость МК-системы.

Второй способ (динамический) основан на том, что любой световой индикатор является инерционным прибором, а человеческому глазу отображаемая на дисплее информация, если ее обновлять с частотой примерно 20 раз в секунду, представляется неизменяемой. Динамический способ вывода информации на дисплей требует значительно меньших аппаратурных затрат, но более сложного программного обеспечения. Именно этот способ организации вывода информации получил преимущественное распространение в МК-системах.

При динамической индикации байт индикации поступает одновременно на входы всех семисегментных светодиодных индикаторов (ССИ), образующих линейный дисплей, а выбор знакоместа осуществляется байтом выборки, представляющим собой код "бегущий нуль" (рис.1). При бездешифраторном способе формирования байта выборки максимальное число знакомест линейного дисплея ограничено разрядностью порта. Использование для формирования кода "бегущий нуль" внешнего дешифратора позволяет значительно увеличить число знакомест линейного дисплея.



RD- регистр данных для временного хранения отображаемого числа либо символа.
RA- регистр адреса для временного хранения двоичного кода адреса выбираемого индикатора.
DA-для преобразования адреса задаваемого двоичным кодом а позиционный код.
HL1-HL4- семи сегментные индикаторы.

Рис.1 Линейный дисплей на семисегментных светодиодных индикаторах.

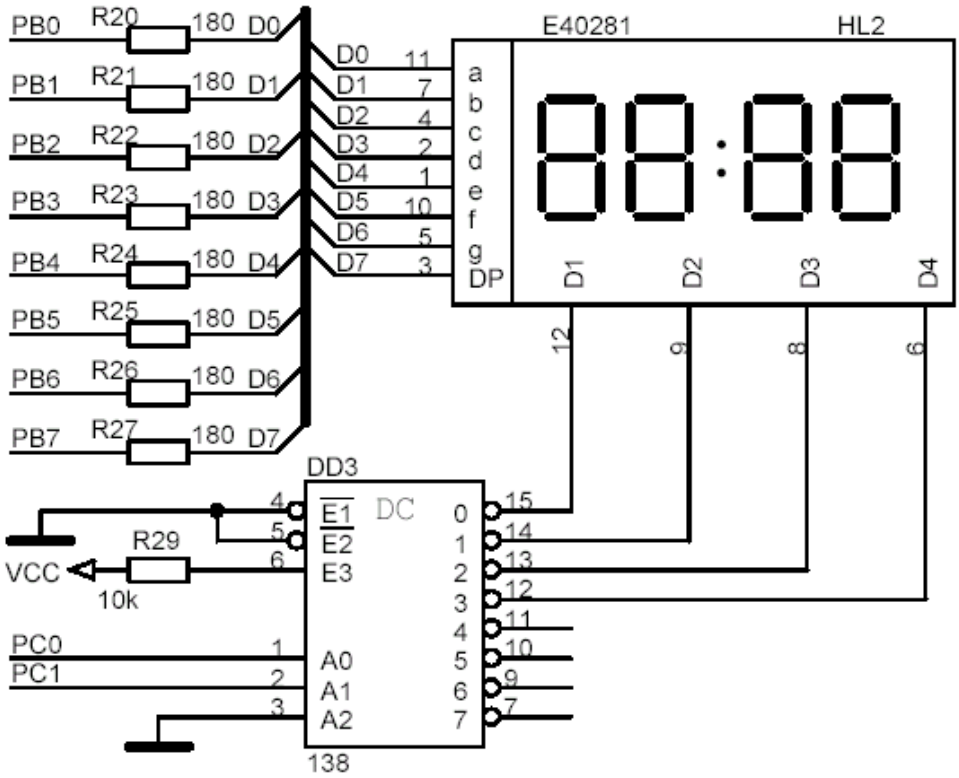


Рис.2 Схема динамической индикации лабораторного стенда EV8031/AVR

Задание для выполнения лабораторной работы.

Составить программу индикации заданного текста на динамическую индикацию (HL2).

Таблица 1. Варианты заданий для лабораторной работы

Вар№	1	2	3	4	5	6	8
Текст	A024	BC36	BCDE	12EF	3678	EF57	AF69
Вар№	9	10	11	12	13	14	15
Текст	123F	357D	579A	EF56	901A	34AD	67AE

Адреса портов для управления индикатором:

– порт B (b55) – 0x8001;

– порт C (c55) – 0x8002;

– регистр управляющего слова (rus55) – 0x8003.

Пример выполнения задания.

```
/*Программа для стенда EV8031/AVR(V3.2)
Программа динамической индикации на HL2.
Программа записывает в буфер BUF коды 4 цифровых символов индикатора
 и динамически выдает их на циф. индикатор HL2. */
/*************************************************************************
**/
#include<reg51.h>
at 0x8001 unsigned char xdata b55;
at 0x8002 unsigned char xdata c55;
at 0x8003 unsigned char rus;
unsigned char buf[4]={0x77,0x7c,0xfb,0x5e};//a,b,c,d.
unsigned char n;
//Обработчик преріваний от T0
*************************************************
void timer_0() interrupt 1
{
        b55=0;//Потушили
        c55=n;//Выдали сигнал сканирования
        b55=buf[n];//Выдаем на информационную шину очередной символ
        n++;
        if (n==4) n=0;
}
//************************************************************************
***
void main()
{
        TMOD=0x00;//13р. 8,192 ms
        ET0=1;//Разрешили прерівания от T0
```

```
    EA=1;//Сняли блокировку преріваний
    TR0=1;//Пуск T0
    n=0;//Номер знакоместа индикатора
    rus55=0x80;//Настроили порты ППА на вывод
    while(1);
}
```

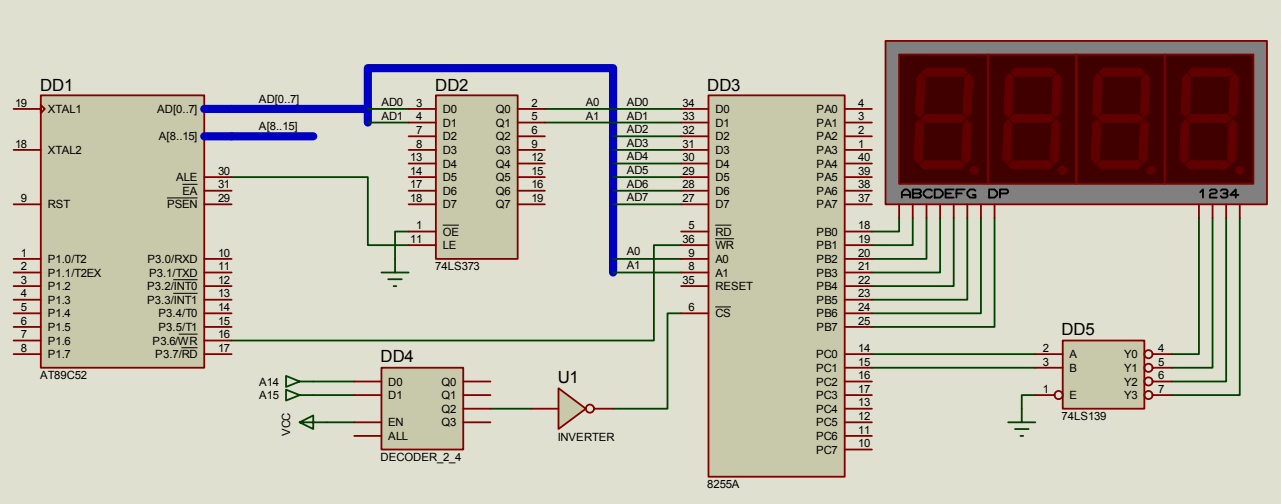


Рис.3 Модель схемы динамической индикации в Proteus.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему динамической индикации EV8031.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1. Приведите структурную схему динамической индикации.
2. Схемотехнические отличия динамической индикации от статической.
3. Расчет времени регенерации для динамического метода отображения информации.
4. Как выполняется программно перекодировка код символа в код цифрового индикатора.
5. Зачем нужен таймер в программе управления динамической индикацией стенда EV8031.
6. Зачем используется система прерываний контроллера для программы динамической индикации стенда EV8031.

Лабораторная работа №4.

ОТОБРАЖЕНИЕ ИНФОРМАЦИИ НА МАТРИЧНЫЙ ИНДИКАТОР.

Цель работы: изучение принципа формирования символов на матричном светодиодном индикаторе.

Краткие теоретические сведения.

Для отображения символа на матричный индикатор обычно используется принцип динамической индикации по колонкам. Байт индикации представляет собой код засветки светодиодов колонками, и графический образ символа ″набирается″ из последовательности байтов индикации путем перебора колонок (рис.1).

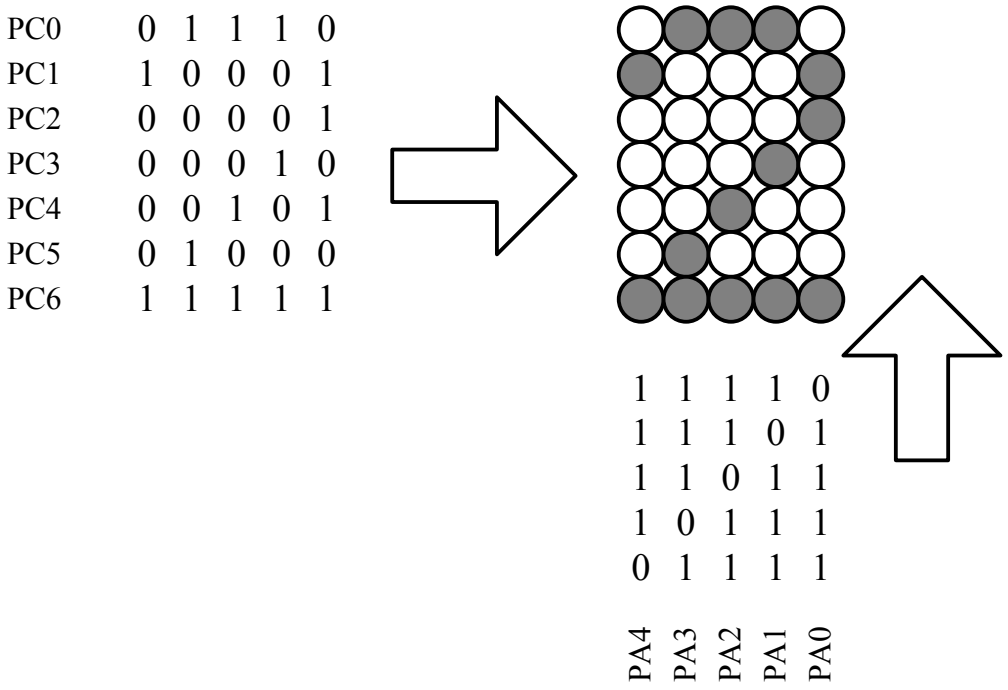


Рис. 1 Формирования символа на матричном индикаторе.

Принципиальная схема матричной индикации лабораторного стенда EV8031/AVR, представлена на рисунке 2.

Адреса портов для управления индикатором:

– порт A (a55) – 0x8000;

– порт C (c55) – 0x8002;

– регистр управляющего слова (rus55) – 0x8003.

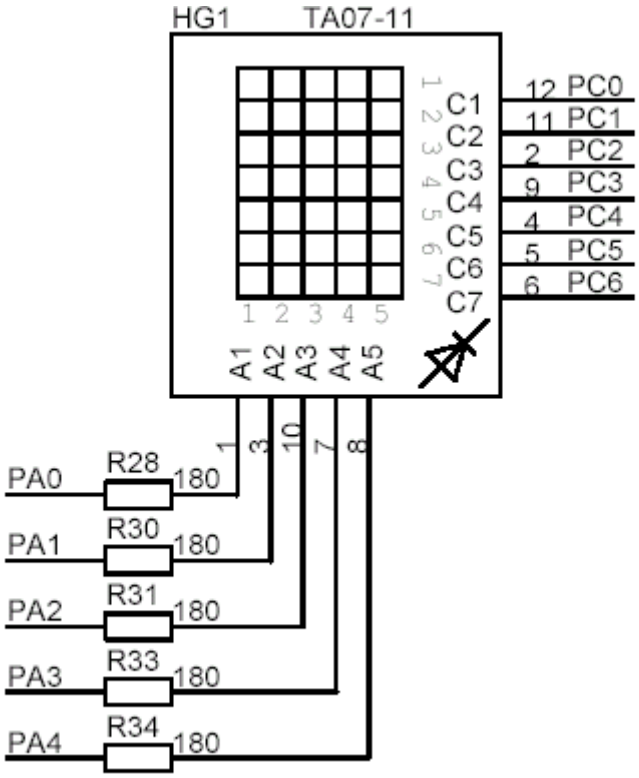


Рис.2 Схема матричной индикации лабораторного стенда EV8031/AVR.

Задание для выполнения лабораторной работы.

Составить программу на СИ индикации заданного цифрового символа 16-ричной системы счисления на матричный индикатор HG1.

Табл.1 Варианты индивидуальных заданий

№	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Сим.	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Пример выполнения задания.

```
/* Для стенда EW8031/AVR(V3.2)*/
/*Программа динамической индикации на матричный индикатор HG1.
Перед запуском программы необходимо в ячейку SIM записатть код высвеч. символа.
Программа записывает в буфер BUF коды 5 колонок соответствующие высв. символу и
динамически выдает их на циф. индикатор HG1. */
#include<reg51.h>
#define uw 0x80
at 0x8000 unsigned char xdata a55;
at 0x8001 unsigned char xdata b55;
at 0x8002 unsigned char xdata c55;
at 0x8003 unsigned char xdata rus55;
unsigned char data buf[5]={0x00,0x00,0x00,0x00,0x00};
```

```
/*Таблица кодов колонок 16 цифровых символов (0,1,2,3,....9,A,B,C,D,E,F) ДЛЯ HTG1*/
        unsigned char code codtbl[]=
        {0x3e,0x41,0x41,0x41,0x3e,
        0x00,0x00,0x10,0x20,0x7f,
        0x31,0x43,0x45,0x49,0x31,
        0x22,0x41,0x49,0x49,0x36,
        0x04,0x0c,0x14,0x24,0x7f,
        0x71,0x51,0x51,0x51,0x4e,
        0x3e,0x49,0x49,0x49,0x46,
        0x47,0x40,0x50,0x60,0x40,
        0x36,0x49,0x49,0x49,0x36,
        0x39,0x45,0x45,0x45,0x3e,
        0x01,0x04,0x14,0x44,0x7f,
        0x7e,0x49,0x49,0x49,0x3e,
        0x3e,0x41,0x41,0x41,0x41,
        0x7f,0x41,0x41,0x41,0x3e,
        0x7f,0x49,0x49,0x49,0x49,
        0x7f,0x48,0x48,0x48,0x48};
        char k;
        char n;
        unsigned char cw=0xfc;
        unsigned char codw=0x20;
        unsigned char sim=0x05;
//функция инициализации таймера и системы
        void inicdi()
        {
                TL0=0x00;
                TH0=0xB0;
                TMOD=0x00;
                IE=0x82;
                TCON=0x10;
                rus55=uw;
                k=0;
                return;
        }
        //обработчик прерывания от таймера c/t0
        void dind()interrupt 1
        {
        a55=0x00;
        b55=0x00;
        c55=~(buf[k]);
        codw=codw>>1;
```

```
        a55=codw;
        k++;
        if(k==5){k=0;codw=0x20;}
        }
        /* функция заполнения буфера кодами колонок символа, выводимого на индика-
цию*/
        void indik(char sim)
        {
        unsigned char p;
        p=sim*0x05;
        for(n=0;n<=4;n++)buf[n]=codtbl[p+n];
        return;
        }
        void main ()
        {
        inicdi();
        indik(sim);
        while(1);
        }
```

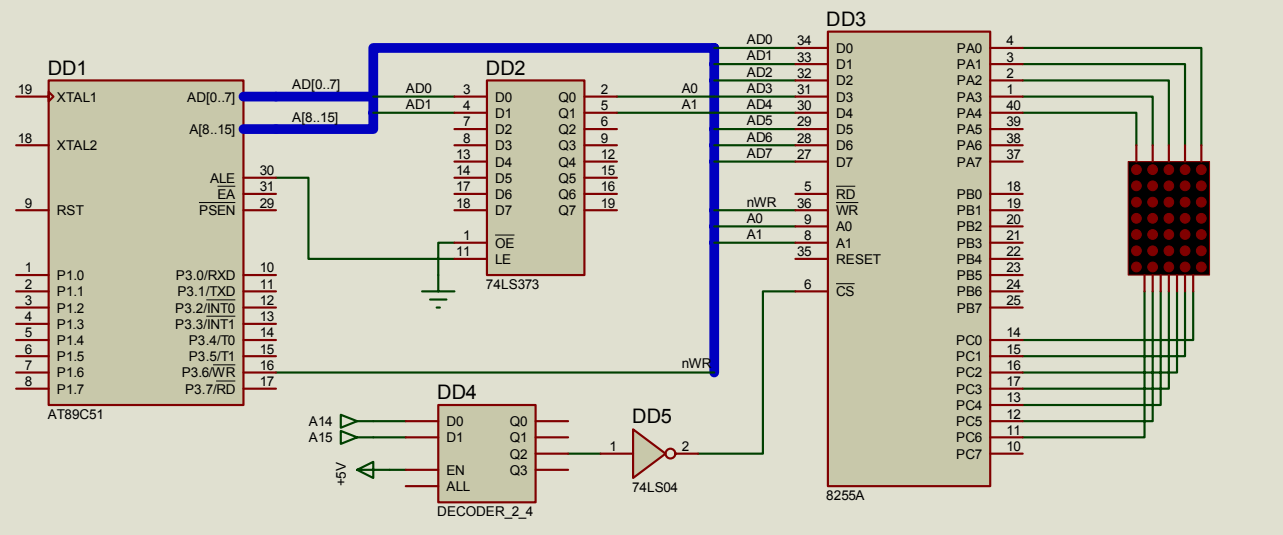


Рис.3 Модель схемы матричной индикации.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему матричной индикации EV8031.

4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1. Приведите схему подключения матричного индикатора стенда EV8031.
2. Запишите коды колонок матричного индикатора для следующих символов 0, 1, 5, F.
3. Зачем нужен таймер в схеме матричной индикации? Как выполняется расчет констант таймера для обслуживания матричного индикатора?
4. Как используется система прерываний в матричной индикации?
5. Действия, которые должна выполнить функция обработчика прерывания.
6. Как вычисляется начальный адрес кодов колонок символа, выдаваемого на матричную индикацию.

Домашнее задание №1.

1. Составить программу выдачи цифрового текста, состоящего из трех цифр 16-ричной системы счисления.
2. Собрать схему выдачи указанного текста на трех разрядный матричный индикатор в PROTEUS.

Домашнее задание №2.

1. Составить программу выдачи цифрового текста, состоящего из ‘n’ цифр 16-ричной системы счисления бегущей колонкой на трех разрядный матричный индикатор.
2. Собрать схему выдачи цифрового текста на трехразрядный матричный индикатор в PROTEUS.

Таблица вариантов задания №2

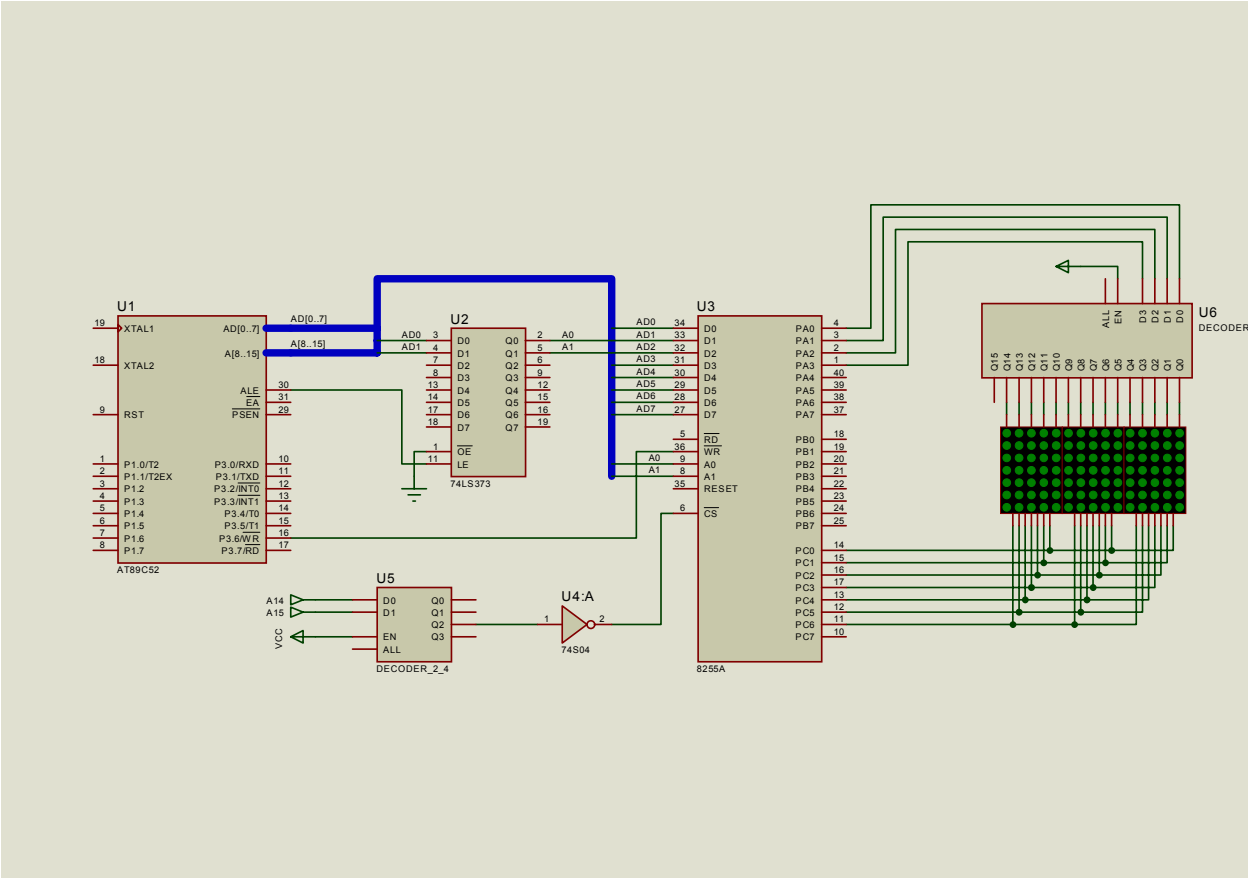
Номер варианта	1	2	3	4	5	6	7	8	9	10
Число символов N	5	8	11	13	16	18	20	24	26	28
T,сек (время идикации колонки)	1	1,5	2	2,5	3	3,5	4	2	2,5	3

Пример схемы и программы приведен ниже.

```
/*Программа выполняет функцию выдачи цифрового текста на 3-х раз-
    рядный матричный индикатор. Текст должен находится в памяти
    ХДАТА и состоит из 16 цифровых символов из набора 16-ричной
    сис.счисления и код пробела 0x20  */
#include <reg51.h>
/*Буфер индикации для 15 колонок 3-х индицируемых цифр */
unsigned char data buf[]=
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
    0x00,0x00};
unsigned char data tbl[]=
{0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x20,
    0x20,0x20};
/*Таблица кодов колонок 16 цифровых символов (0,1,2,3,....9,A,B,C,D,E,F)
    ДЛЯ HTG1*/
unsigned char code codtbl[]=
{0x3e,0x41,0x41,0x41,0x3e,
0x00,0x00,0x10,0x20,0x7f,
0x31,0x43,0x45,0x49,0x31,
0x22,0x41,0x49,0x49,0x36,
0x04,0x0c,0x14,0x24,0x7f,
0x71,0x51,0x51,0x51,0x4e,
```

```
0x3e,0x49,0x49,0x49,0x46,
0x47,0x40,0x50,0x60,0x40,
0x36,0x49,0x49,0x49,0x36,
0x39,0x45,0x45,0x45,0x3e,
0x01,0x04,0x14,0x44,0x7f,
0x7e,0x49,0x49,0x49,0x3e,
0x3e,0x41,0x41,0x41,0x41,
0x7f,0x41,0x41,0x41,0x3e,
0x7f,0x49,0x49,0x49,0x49,
0x7f,0x48,0x48,0x48,0x48,
0x00,0x00,0x00,0x00,0x00};
short countw;
char l,codw,sim,p,n,k,j;
at 0x8000 unsigned char xdata a55;
at 0x8001 unsigned char xdata b55;
at 0x8002 unsigned char xdata c55;
at 0x8003 unsigned char xdata rus55;
/*Подпрограмма инициализации..... */
void inicdi()
{
TL0=0xfa;
TH0=0xff;
TMOD=0x01;//c/t0 в режиме1
IE=0x82;//разрешить прерывания
rus55=0x80;
/*начальные значения индексов*/
countw=0x0258;
k=0;
sim=0;p=0;n=0;l=0;codw=0x0e;//разрешить счет таймеру
TCON=0x10;
}/*Функция обработчик прерывания от таймера T0 */
void dind()interrupt 1
{
m2:
TL0=0x8f;
TH0=0xff;//константы T0 для 2.6мс
```

```
countw--;
if(countw<=0)goto m1;
a55=0x00;
b55=0x00;
c55=~(buf[k]);
codw--;
a55=codw;
k++;
if(k==14){k=0;codw=0x0e;}
goto z2;
m1:
countw=0x0258;
sim=tbl[l];
if(sim==0x20){p=80;goto z1;}else p=sim*0x05;
z1:
buf[15]=codtbl[p+n];
for(j=0;j<=14;j++)buf[j]=buf[j+1];
if(n<=3){n++;goto m2;}
else if(l<=14){l++;n=0;buf[15]=0x00;for(j=0;j<=14;j++)buf[j]=buf[j+1];
 goto m2;}
else {l=0;n=0;k=0;countw=0x0258;goto m2;}
z2:
b55=0x00;
}
//-------------------------------------------------------------
void main ()
{
inicdi();
while(1);
}
```



Лабораторная работа №5.

ВВОД С КЛАВИАТУРЫ И ВЫВОД НА ЦИФРОВУЮ ДИНАМИЧЕСКУЮ ИНДИКАЦИЮ.

Цель работы: изучение принципов разработки программ ввода с клавиатуры и вывода на индикацию.

Краткие теоретические сведения.

Опрос дискретных сигналов. Для ввода информации широко применяются кнопочные переключатели и контактные клавиатуры. Сигнал таких переключателей формируется путем замыкания (размыкания) электрической цепи. Сигнал, формируемый контактной парой, сопровождается дребезгом, длительность которого составляет ~8-12мс (рис.1).

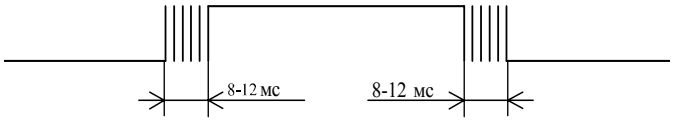


Рис.1 Сигнал контактной пары.

Для устранения дребезга в получаемом сигнале на выходе контактной пары устанавливают специальные формирователи. Пример такого формирователя основанного на принципе непосредственной установки RS-триггера приведён на рисунке 2.

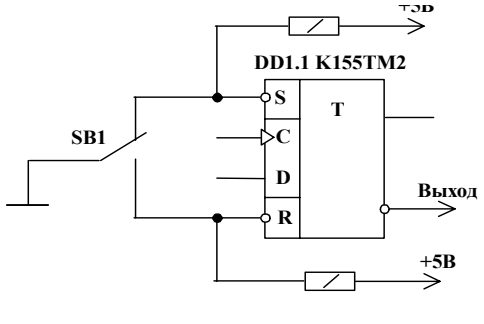


Рис.2 Схема устранения дребезга с помощью RS-триггера

Для уменьшения аппаратных затрат применяют программное подавление дребезга. Оно заключается в повторном опросе контактной пары с задержкой в 12мс, при совпадении результатов опроса кнопка была нажата, иначе в результате первого опроса был зафиксирован дребезг.

В составе учебно-отладочного стенда имеется матричная 3x4 клавиатура SW3-SW14. Клавиатура подключена к шине данных ОЭВМ при помощи микросхемы буфера DD1 74245(АП6).

Опрос всей клавиатуры производится за три раза (за один раз считывается состояние только одного столбца клавиатуры). Чтобы произвести опрос столбца клавиатуры (SW3,SW6,SW9,SW12; SW4,SW7,SW10,S13; или SW5,SW8,SW11,SW14) необходимо выставить на соответствующей линии адреса (A0,A1,A2 для первого, второго и третьего столбца соответственно) уровень логического нуля, а на других линиях уровень логической единицы и прочесть состояние буфера клавиатуры, подключенного к шине данных ОЭВМ как доступная для чтения ячейка памяти с адресом 9000h. Если кнопка нажата то соответствующий бит в считанном байте будет равен нулю, если же не нажата то единице.

Табл.1 Распределение адресов столбцов матричной клавиатуры.

Столбец (кнопки)	Адрес
1 (SW3,SW6,SW9,SW12)	9006h
2 (SW4,SW7,SW10,SW13)	9005h
3 (SW5,SW8,SW11,SW14)	9003h

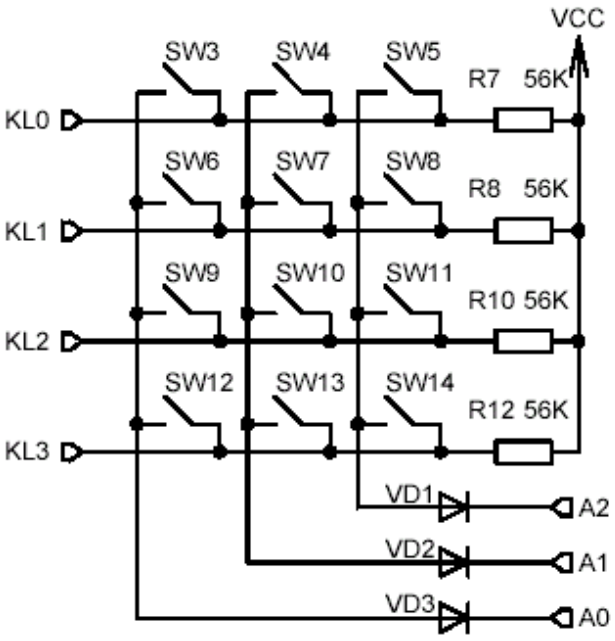


Рис.3 Принципиальная схема матричной клавиатуры стенда EV8031/AVR

Задание для выполнения лабораторной работы.

Составить программу сканирования клавишей первого или второго или третьего столбца клавиатуры. Первый столбец – клавиши (1,4,7), второй столбец(2,5,8,0) и третий столбец(3,6,9). Сформировать коды указанных клавиш. Код нажатой клавиши заданного столбца необходимо выдать на заданный тип индикации. Задания приведены в таблице 2.

Табл.2 Варианты заданий к лабораторной работе

Тип индикации	1 колонка (1,4,7)	2 колонка(2,5,8,0)	3 колнка(6,6,9)
Статическая индикация	1вар. Адрес 0xBxx0	2вар. Адрес 0xBxx0	3вар. Адрес 0xAxx0
Динамическая индикация	4вар. 1-ое знакоместо	5вар. 1-ое знакоместо	6вар. 1-ое знакоместо
Динамическая индикация	7вар. 2-ое знакоместо	8вар. 2-ое знакоместо	9вар. 2-ое знакоместо
Динамическая индикация	10вар. 3-ое знакоместо	11вар. 3-ое знакоместо	12вар. 3-ое знакоместо
Динамическая индикация	4-ое знакоместо	4-ое знакоместо	4-ое знакоместо

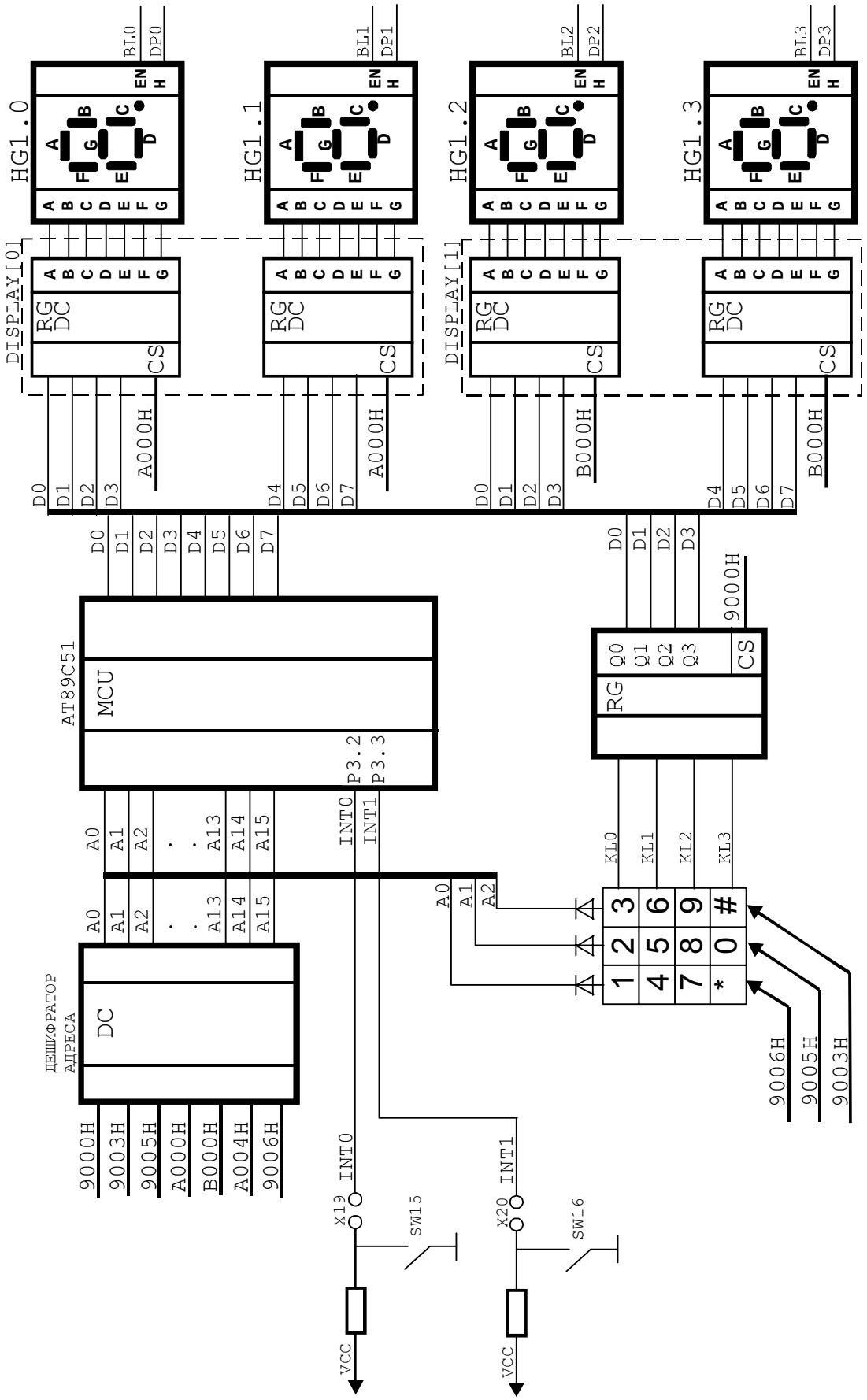


Рис.4 Общая схема включения клавиатуры

Пример выполнения задания.

```
/*Программа для стенда EV8031/AVR(V3.2)*/
/*Программа обслуживания клавиатуры
и динамической индикации на HL2.
С клавиатуры получаем коды клавиш от 0 до 9
и коды 16сс A,B,C,D,E,F.
Код A получаем двумя нажатиями кнопок вначале нажать и отпустить
 кнопку # а затем например 1(нажать и отпустить) получим символ A
 итого # 1 -A;#4 -B;#7 -C;#3 -D;#8 -E;#9 -F.
При нажатии кнопки, например 3, программа выдает на индикацию
 инверсию кода 3 цифрового индикатора (0xb0)- 0x4f.
Программа записывает в буфер BUF коды 4 цифровых символов индикатора набираемых клавиатурой и динамически выдает их на циф. индикатор HL2. */
/*************************************************************************
**/
#include<reg51.h>
#include<stdio.h>
#define uw 0x80
at 0x8000 unsigned char xdata a55;
at 0x8001 unsigned char xdata b55;
at 0x8002 unsigned char xdata c55;
at 0x8003 unsigned char xdata rus55;
at 0x9003 unsigned char xdata s3s12;
at 0x9005 unsigned char xdata s2s11;
at 0x9006 unsigned char xdata s1s10;
unsigned char data buf[4]={0x71,0x71,0x71,0x71};
char k;
unsigned char n;
unsigned char cw=0xfc;
unsigned char klav;
unsigned char klav1;
unsigned char sim;
unsigned char sim1;
//функция инициализации таймера и системы
void inicdi()
{
TL0=0x1C;
TH0=0xB1;
TMOD=0x00;
IE=0x82;
TCON=0x10;
rus55=uw;
```

```
n=0;
k=0;
return;
}
//обработчик прерывания от таймера c/t0
void dind()interrupt 1
{
b55=buf[n];
c55=cw;
n++;
cw++;
if(n==4){n=0;cw=0xfc;}
}
void delay()
{
char i,j;
for(i=0x7f;i>=0;i--){
for(j=0x7f;j>=0;j--);}
return;
}
void indik(char sim)
{
buf[k]=sim;
k++;
if(k==4)k=0;
return;
}
void main ()
{
inicdi();
m1: klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xff)goto m2; else delay();
klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xf7)goto k0;
if(klav1==0xfe){sim=0x4f;sim1=0x33; goto z1;}//Символ "3"
else if(klav1==0xfd){sim=0x7d;sim1=0x36;goto z1;}//Символ"6"
else if(klav1==0xfb){sim=0x6f;sim1=0x39;goto z1;}//Символ "9"
else goto m1;
z1:indik(sim);
// Определение факта отпускания клавиши
```

```
        while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
        delay();
        goto m1;
        m2:
        klav=s2s11;
        klav1=klav|0xf0;
        if(klav1==0xff)goto m3; else delay();
        if(klav1==0xf7){sim=0x3f;sim1=0x30;goto z2;}//Символ "0"
        if(klav1==0xfe){sim=0x5b;sim1=0x32;goto z2;}//Символ"2"
        if(klav1==0xfd){sim=0x6d;sim1=0x35;goto z2;}//символ"5"
        if(klav1==0xfb){sim=0x7f;sim1=0x38;goto  z2;}//Символ"8"
        else goto m1;
        z2:
        indik(sim);
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s2s11;klav1=klav|0xf0;}
        delay();
        goto m1;
        m3:
        klav=s1s10;
        klav1=klav|0xf0;
        if(klav1==0xff)goto m1; else delay();
        if(klav1==0xfe){sim=0x06;sim1=0x31;goto z3;}//Символ"1"
        if(klav1==0xfd){sim=0x66;sim1=0x34;goto z3;}//Символ"4"
        if(klav1==0xfb){sim=0x07;sim1=0x37;goto z3;}//Символ"7"
        else goto m1;
        z3:
        indik(sim);
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s1s10;klav1=klav|0x0f;}
        delay();
        goto m1;
//*************************************************************************
***
        //Была нажата клавиша реш.'#'
        k0:
        while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
        delay();
        k1:
        klav=s3s12;
        klav1=klav|0xf0;
        if(klav1==0xff)goto am1; else delay();
```

```
klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xfe){sim=0x5e;sim1=0x44;goto aut1;}//Символ"Д"
if(klav1==0xfd){sim=0x79;sim1=0x45;goto aut1;}//Символ"E"
if(klav1==0xfb){sim=0x71;sim1=0x46;goto aut1;}//Символ "F"
else goto k1;
aut1:
indik(sim);
//Определение факта отпускания клавиши
while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
delay();
goto m1;
am1:
klav=s1s10;
klav1=klav|0xf0;
if(klav1==0xff)goto k1; else delay();
klav=s1s10;
klav1=klav|0xf0;
if(klav1==0xfe){sim=0x77;sim1=0x41;goto aut2;}//символ"A"
if(klav1==0xfd){sim=0x7c;sim1=0x42;goto aut2;}//Символ"B"
if(klav1==0xfb){sim=0x39;sim1=0x43;goto aut2;}//Символ"C"
else goto k1;
aut2:
indik(sim);
//Определение факта отпускания клавиши
while(klav1!=0xff){klav=s1s10;klav1=klav|0xf0;}
//Устранение дребезга контактов
delay();
goto m1;
}
```

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему лабораторной работы стенда EV8031/AVR.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

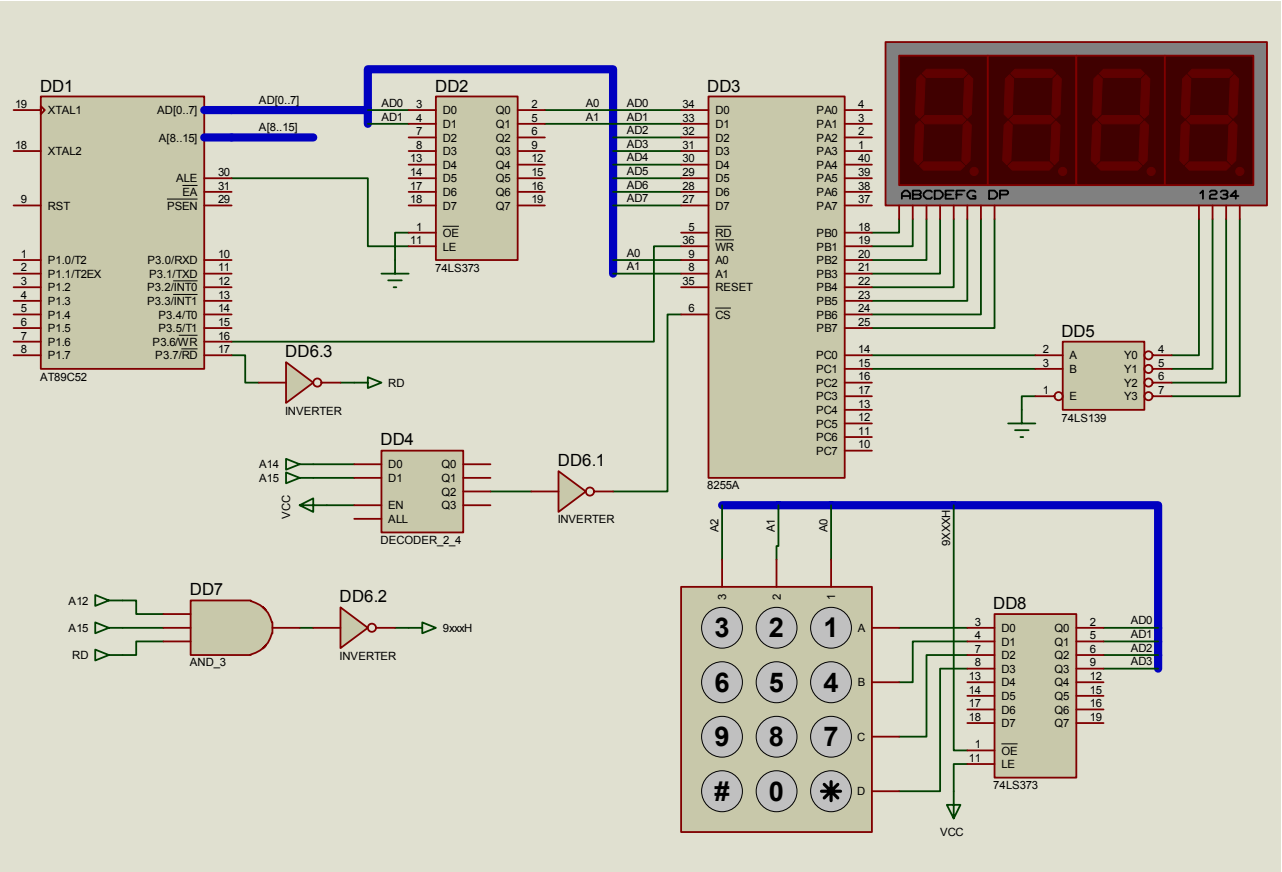


Рис.5 Модель схемы лабораторной работы.

Контрольные вопросы.

1. Аппаратное устранение дребезга контактов для схем с TTL и KMOP.
2. Программное устранение дребезга контактов.
3. Принципиальная схема матричной клавиатуры.
4. Схема подключения матричной клавиатуры стенда EV8031.
5. Приведите текст программы сканирования и идентификации нажатой клавиши колонки с цифрами 3, 6, 9, #.
6. Приведите текст программы сканирования и идентификации нажатой клавиши колонки с цифрами 1, 4, 7,*.
7. Приведите текст программы сканирования и идентификации нажатой клавиши колонки с цифрами D, E, F.

Лабораторная работа №6.

ВВОД С КЛАВИАТУРЫ И ВЫВОД НА МАТРИЧНЫЙ ИНДИКАТОР HG1 СТЕНДА EV8031.

Цель работы: изучить принцип программного управления клавиатурой и матричной индикацией.

Задание для выполнения лабораторной работы.

Закрепить за каждой клавишей символ 16-ричной системы счисления (закрепление выполнить как в приведенном ниже примере). Записать программу ввода заданного символа с клавиатуры и вывода его на матричный индикатор. Варианты заданий приведены в таблице 1.

Табл.1 Варианты заданий к лабораторной работе

Вар.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Сим.	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Пример выполнения задания.

Пример программы ввода с клавиатуры символов 16-ричной системы счисления и вывода их на матричный индикатор HG1 стенда EV8031/AVR

```
/* Для стенда EW8031/AVR(V3.2)*/
/*Программа обслуживания клавиатуры
и динамической индикации на матричный индикатор HG1.
С клавиатуры получаем коды клавиш от 0 до 9
и коды 16сс A,B,C,D,E,F.
Код А получаем двумя нажатиями кнопок вначале нажать и отпустить
 кнопку # а затем например 1(нажать и отпустить) получим символ А
 итого # 1 -A;#4 -B;#7 -C;#3 -D;#8 -E;#9 -F.
При нажатии кнопки, например 3, программа выдает на индикацию
 код  16сс 0x03.
Программа записывает в буфер BUF коды 5 колонок соответствующих
цифре нажатой клавиши  и динамически выдает их на циф. индикатор HG1. */
#include<reg51.h>
#include<stdio.h>
#define uw 0x80
at 0x8000 unsigned char xdata a55;
at 0x8001 unsigned char xdata b55;
at 0x8002 unsigned char xdata c55;
at 0x8003 unsigned char xdata rus55;
at 0x9003 unsigned char xdata s3s12;
```

```
        at 0x9005 unsigned char xdata s2s11;
        at 0x9006 unsigned char xdata s1s10;
        unsigned char data buf[5]={0x00,0x00,0x00,0x00,0x00};
/*Таблица кодов колонок 16 цифровых символов (0,1,2,3,....9,A,B,C,D,E,F) ДЛЯ HTG1*/
        unsigned char code codtbl[]=
        {0x3e,0x41,0x41,0x41,0x3e,
        0x00,0x00,0x10,0x20,0x7f,
        0x31,0x43,0x45,0x49,0x31,
        0x22,0x41,0x49,0x49,0x36,
        0x04,0x0c,0x14,0x24,0x7f,
        0x71,0x51,0x51,0x51,0x4e,
        0x3e,0x49,0x49,0x49,0x46,
        0x47,0x40,0x50,0x60,0x40,
        0x36,0x49,0x49,0x49,0x36,
        0x39,0x45,0x45,0x45,0x3e,
        0x01,0x04,0x14,0x44,0x7f,
        0x7e,0x49,0x49,0x49,0x3e,
        0x3e,0x41,0x41,0x41,0x41,
        0x7f,0x41,0x41,0x41,0x3e,
        0x7f,0x49,0x49,0x49,0x49,
        0x7f,0x48,0x48,0x48,0x48};
        char k;
        char n;
        unsigned char cw=0xfc;
        unsigned char codw=0x20;
        unsigned char klav;
        unsigned char klav1;
        unsigned char sim;
        unsigned char sim1;
        //функция инициализации таймера и системы
        void inicdi()
        {
        TL0=0x00;
        TH0=0xB0;
        TMOD=0x00;
        IE=0x82;
        TCON=0x10;
        rus55=uw;
        k=0;
        return;
        }
        //обработчик прерывания от таймера c/t0
        void dind()interrupt 1
        {
        a55=0x00;
        b55=0x00;
        c55=~(buf[k]);
        codw=codw>>1;
        a55=codw;
        k++;
        if(k==5){k=0;codw=0x20;}
        }
```

```
/*Функция задержки для устранения дребезга контактов при нажатии
и отпускания клавиш */
void delay()
{
char i,j;
for(i=0x7f;i>=0;i--){
for(j=0x7f;j>=0;j--);}
return;
}
/* функция заполнения буфера кодами колонок символа, выводимого на индика-
цию*/
void indik(char sim)
{
unsigned char p;
p=sim*0x05;
for(n=0;n<=4;n++)buf[n]=codtbl[p+n];
return;
}
void main ()
{
inicdi();
m1: klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xff)goto m2; else delay();
klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xf7)goto k0;
if(klav1==0xfe){sim=0x03;sim1=0x33; goto z1;}//Символ "3"
else if(klav1==0xfd){sim=0x06;sim1=0x36;goto z1;}//Символ"6"
else if(klav1==0xfb){sim=0x09;sim1=0x39;goto z1;}//Символ "9"
else goto m1;
z1:indik(sim);
// Определение факта отпускания клавиши
while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
delay();
goto m1;
m2:
klav=s2s11;
klav1=klav|0xf0;
if(klav1==0xff)goto m3; else delay();
if(klav1==0xf7){sim=0x00;sim1=0x30;goto z2;}//Символ "0"
if(klav1==0xfe){sim=0x02;sim1=0x32;goto z2;}//Символ"2"
if(klav1==0xfd){sim=0x05;sim1=0x35;goto z2;}//символ"5"
if(klav1==0xfb){sim=0x08;sim1=0x38;goto  z2;}//Символ"8"
else goto m1;
z2:
indik(sim);
//Определение факта отпускания клавиши
while(klav1!=0xff){klav=s2s11;klav1=klav|0xf0;}
delay();
goto m1;
m3:
```

```
        klav=s1s10;
        klav1=klav|0xf0;
        if(klav1==0xff)goto m1; else delay();
        if(klav1==0xfe){sim=0x01;sim1=0x31;goto z3;}//Символ"1"
        if(klav1==0xfd){sim=0x04;sim1=0x34;goto z3;}//Символ"4"
        if(klav1==0xfb){sim=0x07;sim1=0x37;goto z3;}//Символ"7"
        z3:
        indik(sim);
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s1s10;klav1=klav|0x0f;}
        delay();
        goto m1;
//*************************************************************************
***
        //Была нажата клавиша реш.'#'
        k0:
        while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
        delay();
        k1:
        klav=s3s12;
        klav1=klav|0xf0;
        if(klav1==0xff)goto am1; else delay();
        klav=s3s12;
        klav1=klav|0xf0;
        if(klav1==0xfe){sim=0x0d;sim1=0x44;goto aut1;}//Символ"Д"
        if(klav1==0xfd){sim=0x0e;sim1=0x45;goto aut1;}//Символ"E"
        if(klav1==0xfb){sim=0x0f;sim1=0x46;goto aut1;}//Символ "F"
        aut1:
        indik(sim);
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
        delay();
        goto m1;
        am1:
        klav=s1s10;
        klav1=klav|0xf0;
        if(klav1==0xff)goto k1; else delay();
        klav=s1s10;
        klav1=klav|0xf0;
        if(klav1==0xfe){sim=0x0a;sim1=0x41;goto aut2;}//символ"A"
        if(klav1==0xfd){sim=0x0b;sim1=0x42;goto aut2;}//Символ"B"
        if(klav1==0xfb){sim=0x0c;sim1=0x43;goto aut2;}//Символ"C"
        else goto k0;
        aut2:
        indik(sim);
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s1s10;klav1=klav|0xf0;}
        //Устранение дребезга контактов
        delay();
        goto m1;
        }
```

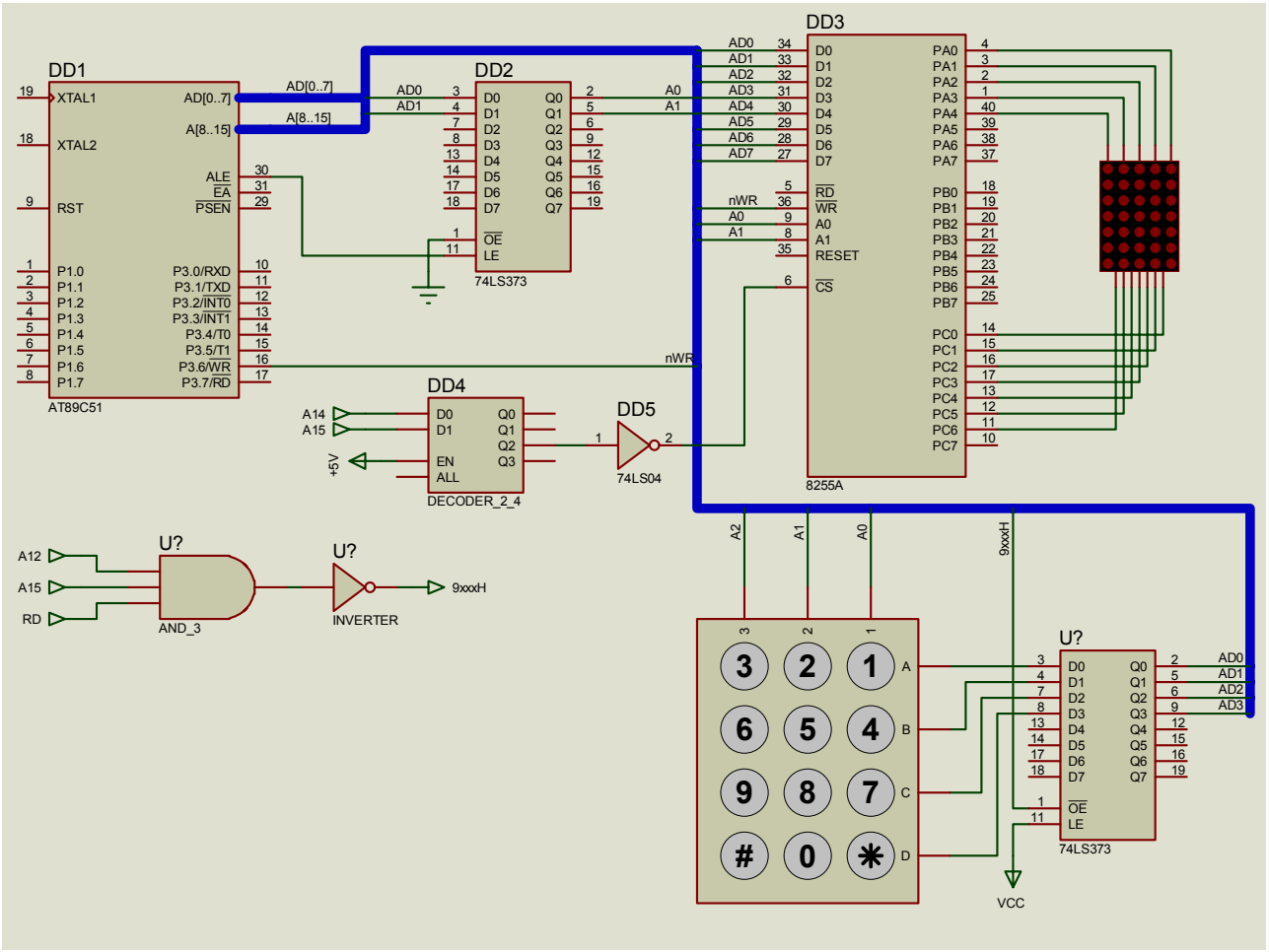


Рис.1 Модель схемы лабораторной работы.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему лабораторной работы стенда EV8031/AVR.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1. Состав функций программы обслуживания клавиатуры и матричной индикации.
2. Действия, выполняемые функцией обслуживания прерываний в программе управления клавиатурой и матричной индикацией.
3. Напишите программу функции задержки для устранения дребезга контактов при нажатии и отпускания клавиши.

4. Напишите программу функции заполнения буфера кодами колонок нажатой клавиши.
5. Напишите программу функции инициализации программы обслуживания клавиатуры и матричной индикации.
6. Действия, выполняемые функцией *main* программы обслуживания клавиатуры и матрицы.

Лабораторная работа №7.

ВЫВОД ТЕКСТОВОЙ ИНФОРМАЦИИ НА ЖИДКО-КРИСТАЛЛИЧЕСКИЙ ИНДИКАТОР СТЕНДА EV8031

Цель работы: Изучение системы команд ЖКИ и принципов инициализации и управления панелями.

Краткие теоретические сведения.

Светодиодные индикаторы способны отображать простую информацию, но они не обладают тем диапазоном возможностей, которые реализуют жидкокристаллические индикаторы (ЖКИ). Эти индикаторы позволяют выводить очень специфичные сообщения, делая интерфейс с пользователем более дружественным. ЖКИ также весьма полезны для вывода сообщений о состоянии устройства и другой необходимой информации в процессе отладки приложения.

Большинство алфавитно-цифровых ЖКИ используют для управления контроллер HITACHI 44780 и реализуют общий интерфейс подключения. Благодаря этим обстоятельствам ЖКИ, обеспечивающие вывод от 8 до 80 символов (организованных в виде двух строк по 40 символов или 4 строк по 20 символов), являются полностью взаимозаменяемыми, так как их применение не требует какого-либо изменения программного обеспечения или аппаратных средств.

Чаще всего ЖКИ, использующие контроллер HITACHI 44780, имеют 14-выводные разъемы с шагом 2,54мм. Выводы ЖКИ имеют следующее назначение:

Вывод 1 - «Земля».

Вывод 2 – Напряжение питания Vcc.

Вывод 3 – Вход регулировки контрастности изображения.

Вывод 4 – Сигнал выбора регистра данных или команд (R/S).

Вывод 5 – Сигнал выбора режима «чтение/запись» (R/W).

Вывод 6 – Синхросигнал E.

Выводы 7-14 – Линии передачи данных.

Из данного описания видно, что интерфейс микроконтроллера с ЖКИ представляет собой параллельную шину, которая позволяет просто и быстро

осуществлять чтение и запись данных в ЖКИ. Временные диаграммы сигналов на рис.1 иллюстрируют процесс выдачи байта, содержащего ASCII-код символа, на экран ЖКИ. ASCII-код символа содержит 8 бит, которые посылаются в ЖКИ по 4 или по 8 бит за один цикл обмена. Если используется 4-битный режим обмена, то полный 8-битный код символа передается в виде двух 4-битных (полубайтов): сначала 4 старших бита, затем 4 младших. Каждая посылка сопровождается синхросигналом E, который инициирует прием данных в ЖКИ.

Передача 4 или 8 бит данных – это два основных режима параллельного обмена. Рассмотрим некоторые соображения по поводу выбора того или иного режима. Восьми битный режим передачи целесообразно использовать, когда требуется высокая скорость обмена и есть не менее 10 доступных линий для ввода-вывода данных. Четырехбитный режим передачи требует, как минимум, 6 линий ввода-вывода. Чтобы подсоединить микроконтроллер к ЖКИ при четырехбитном режиме используются только 4 старших разряда линии данных ДВ7-4 (рисунок 2).

Дальнейшее сокращение числа требуемых линий ввода-вывода может быть обеспечено путем использования сдвигового регистра: в этом случае потребуется всего три линии (рисунок3).

Приведенная ниже таблица содержит набор команд, реализуемых ЖКИ.

R/S	R/w	D7	D6	D5	D4	D3	D2	D1	D0	Команда/Описание
0	0	0	0	0	0	0	0	0	1	Очистить индикатор
0	0	0	0	0	0	0	0	1	*	Вернуть курсор в начальную позицию
0	0	0	0	0	0	0	1	ID	S	Установить направление движения курсора
0	0	0	0	0	1	SC	RL	*	*	Переместить курсор/Сдвинуть экран
0	0	0	0	1	DL	N	F	*	*	Установить размерность интерфейса
0	0	0	1	A	A	A	A	A	A	Переместить курсор на область CGRAM
0	0	1	A	A	A	A	A	A	A	Переместить курсор на экран
0	0	BF	*	*	*	*	*	*	*	Прочесть флаг ЗАНЯТО

1	1	D	D	D	D	D	D	D	D	Вывести ASCII-символ на экран
1	1	D	D	D	D	D	D	D	D	Прочесть ASCII-символ с экрана

Назначение отдельных битов команд:

Указание направления движения курсора:

ID – Перемещение курсора после записи каждого байта, если бит установлен в 1,

S – Сдвиг изображения на экране после записи байта

Включение экрана/курсора

D – Экран Включить (1), (0) Выключить

C – Курсор Включить (1), Выключить (0)

B – Мигание курсора Включить (1), Выключить (0)

Перемещение курсора / Сдвиг экрана

SC – Сдвиг екрана Включить (1) / Выключить (0)

RL – Направление сдвига Вправо(1) / Влево(0)

Установка размерности интерфейса

DL – Разрядность данных 8 (1) / 4 (0)

N – Число строк на экране 1(0)/ 2 (1)

F – Размер шрифта 5x10 (1) / 5x7 (0)

Установка курсора на CGRAM

A – Адрес

Чтение запись ASCII- символов

D – Данные

Обратите внимание, что тип команды определяется числом старших нулей.

Флаг «занято» («busy») устанавливается на время выполнения команды.

Для написания приложений, которые работают с максимально возможной скоростью, необходимо опрашивать этот флаг, чтобы исключить необходимость реализации задержки, рассчитанной на наихудший случай выполнения команды ЖКИ. Обычно скорость обмена с ЖКИ не очень важна, и целесообразно использовать программную задержку, которую не сложно организовать. Выполнение всех команд занимает не более 160мкс, кроме команд

«Очистить индикатор» и «Вернуть курсор в начальную позицию», которые требуют максимум 4,1мс. Для наихудшего случая можно установить задержку в 5мс.

Если не используется опрос флага ЗАНЯТО, то рекомендуется всегда использовать максимальные задержки.

В большинстве применений линию R/W подсоединяют к земле, так как чтение состояния ЖКИ не требуется. Это значительно упрощает приложение, поскольку для считывания данных необходимо менять режим работы выводов – с записи на чтение. В некоторых случаях возможность чтения состояния ЖКИ бывает полезна, например, при прокручивании данных на экране. Подключение линии R/W к земле также освобождает один вывод микроконтроллера.

Перед тем, как вводить в ЖКИ команды или данные, его необходимо инициализировать. Это делается при помощи следующей последовательности действий:

Для 8-битного режима:

1. Подождать более 15мс после подачи питания.
2. Записать 0x30 в ЖКИ и ждать 5мс до завершения выполнения команды.
3. Записать 0x30 в ЖКИ и ждать 160мкс до завершения выполнения команды.
4. Снова записать 0x30 в ЖКИ и ждать 160мкс до завершения выполнения команды или опрашивать флаг ЗАНЯТО.
5. Установить рабочие характеристики ЖКИ.
   Ввести «Установка размерности интерфейса»
   Ввести 0x10, чтобы выключить экран.
   Ввести 0x01. чтобы очистить экран.
   Ввести «Установка направления движения курсора», чтобы установить поведение курсора.
   Ввести «Включение экрана / курсора», чтобы включить экран и, если требуется курсор.

Для инициализации индикатора в 4-битном режиме используется пересылка двух отдельных полубайтов, а не полных байтов, составляющих команду. При посылке байта сначала посылается старший полубайт, затем младший. При этом каждая посылка 4 бит сопровождается переключением линии E.

1. Подождать более 15мс после подачи питания.
2. Записать 0x30 в ЖКИ и ждать 5мс до завершения выполнения команды.
3. Записать 0x3 в ЖКИ и ждать 160мкс до завершения выполнения команды.
4. Снова записать 0x3 в ЖКИ и ждать 160мкс до завершения выполнения команды или опрашивать флаг ЗАНЯТО.
5. Установить рабочие характеристики ЖКИ.

Ввести 0x02 в ЖКИ, чтобы разрешить 4-битный режим

Все следующие команды/данные требуют пересылки двух полубайт.

Ввести «Установка размерности интерфейса».

Ввести0x1, 0x0, чтобы выключить экран.

Ввести 0x0, 0x1, чтобы очистить экран.

Ввести «Установка направления движения курсора», чтобы установить поведение курсора.

Ввести «Включение экрана/курсора»,чтобы включить и, если требуется, курсор.

После того как инициализация завершена, ЖКИ готов к приему команд и данных.

Память модуля HD44780

В модуле HD 44780 используется две разные памяти:

DD-RAM – для хранения отображаемых данных;

CG-RAM – для хранения битовых комбинаций, которые соответствуют матрице размером 5x8 или 5x10(определяет форму символа).

Доступ как к одной, так и к другой памяти осуществляется по текущему адресу, хранимому в счетчике адреса.

Емкость памяти DD составляет 80 знаков, представленных в 8-разрядной ASCII-кодировке.

В двухстрочном табло по 16 символов в строке можно отображать в 1 строке 16 символов из 40, а во второй строке следующие 16символов из 40.

Распределение адресов памяти соответствует рисунку 1.

№ Знакоместа		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
АДРЕС	1-я строка	0h	1h	2h	3h	4h	5h	6h	7h	8h	9h	0Ah	0Bh	0Ch	0Dh	0Eh	0Fh
	2-я строка	40h	41h	42h	43h	44h	45h	46h	47h	48h	49h	4Ah	4Bh	4Ch	4Dh	4Eh	4Fh

Рис.1 Распределение адресов двустрочного ЖКИ

После **сдвига** влево первая строка начинается с адреса $01, а вторая строка – с адреса $41, а после сдвига вправо первая строка начинается с адреса $27, а вторая строка – с адреса $67.

На стенде EV8031/AVR использован 4 строчный LCD HD44780 по 10 символов в строке. Первая строка начинается с адреса 0x00, вторая строка начинается с адреса 0x40, третья строка идет за первой адрес 0x0a, четвертая идет за второй и имеет начальный адрес 0x4a.

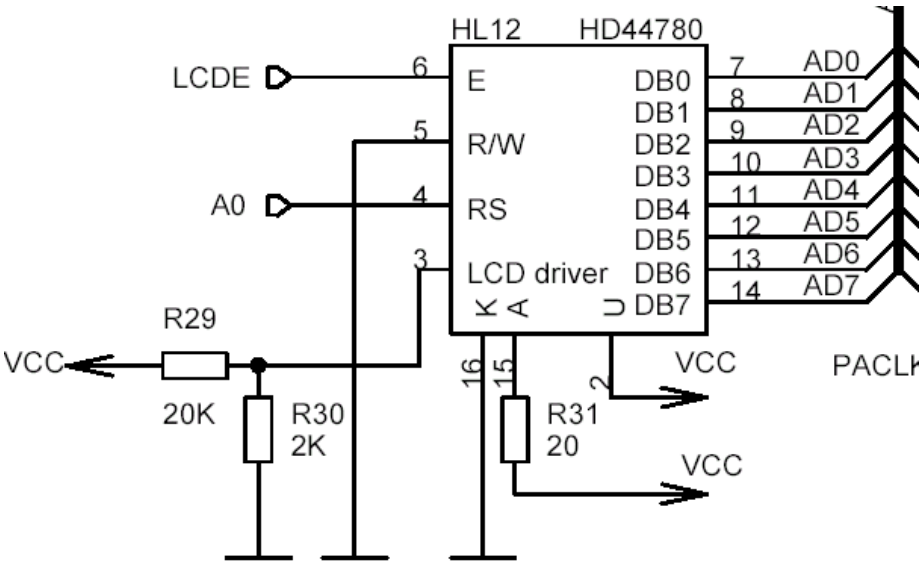


Рис.2 Принципиальная схема ЖКИ стенда EV8031

Запись команд необходимо выполнять по адресу 8xx4h.

Запись данных необходимо выполнять по адресу 8xx5h.

Строб записи E вырабатывается аппаратно.

Задание для выполнения лабораторной работы.

Заготовить текст: фамилия, имя, отчество студента и год рождения. Записать их в CSEG по адресу BUF в коде ASCII. Выдать текст на первую и вторую строку ЖКИ.

Для выполнения задания можно использовать текст программы, приведенной ниже.

```
#include<reg51.h>
#include<stdlib.h>
signed short l,k;
```

```
                at 0x8004 unsigned char xdata DC;
                at 0x8005 unsigned char xdata DD;
                unsigned char code
buf[]={0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,
                0x31,0x32,0x33,0x34,0x35};
                unsigned char t;
                        void pausa(short l)
                {
                for(k=l;k>=0;k--);
                }
                //инициализация ЖКИ
                void start_gki()
                {
                l=0x0535;
                pausa(l);//пауза 20мс
                DC=0x30;
                l=0x029a;//пауза 10мс
                pausa(l);
                DC=0x30;
                l=0x0040;
                pausa(l);
                DC=0x30;
                l=0x0a6a;//пауза 40мс
                pausa(l);
                DC=0x38;
                l=0x0a6a;//пауза40мс
                pausa(l);
                DC=0x0e;
                l=0x014e;//пауза 5мс
                pausa(l);
                DC=0x06;
                l=0x014e;
                pausa(l);
                }
                void gki_out()
                {
                DC=0x01;
                l=0x029a;
                pausa(l);//пауза 10мс
                for(t=0;t<=14;t++)
                {DD=buf[t];
                l=0x0040;
```

```
pausa(l);
}
}
void main()
{
start_gki();
gki_out();
while(1);
}
```

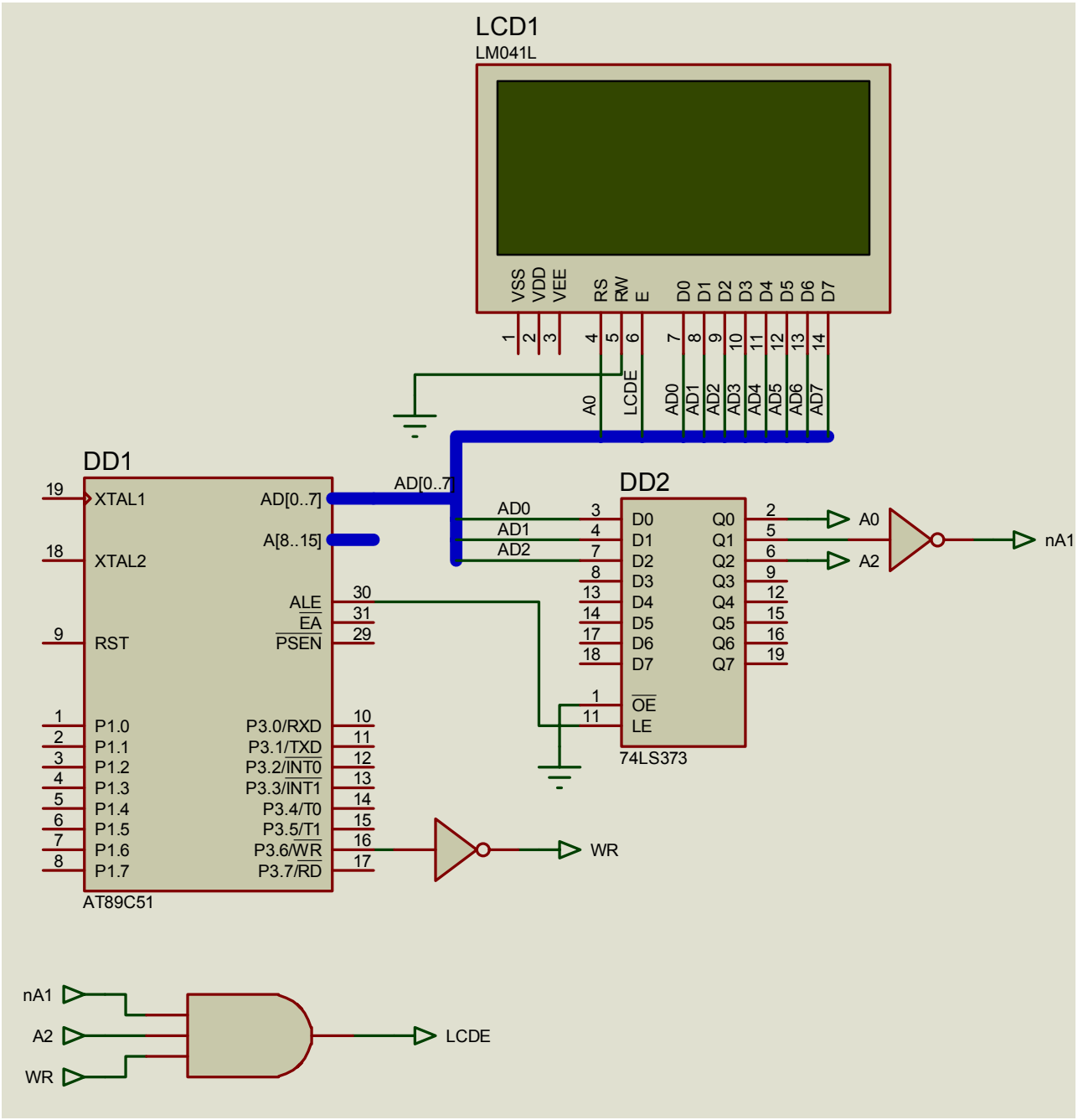


Рис.3 Модель схемы лабораторной работы.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему индикации на ЖКИ стенда EV8031/AVR.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1. Интерфейс LCD, назначение выводов, схема включения LCD на стенде EV8031.
2. Система команд LCD. Назначение каждой команды.
3. Распределение адресов двухстрочного и четырехстрочного LCD.
4. Напишите программу выдачи текста (фамилия имя отчество, год рождения, домашний адрес) на первой, второй, третьей и четвертой строке. Текст начинать с четвертой позиции первой строки.
5. Написать программу выдачи текста (фамилия имя отчество год рождения, домашний адрес) бегущей строкой.
6. Привести программу функции PAUSE. Как настроить программу на отработку различных интервалов паузы.

Лабораторная работа №8.

ВВОД С КЛАВИАТУРЫ И ВЫВОД НА LCD СТЕНДА EV8031.

Цель работы: изучить принципы программного управления клавиатурой и LCD стенда EV8031/AVR

Задание для выполнения лабораторной работы.

Закрепить за каждой клавишей символ 16-ричной системы счисления (закрепление выполнить как в приведенном ниже примере). Записать программу ввода заданного символа с клавиатуры и вывода его на LCD на заданную строку в заданную позицию в строке. Варианты заданий приведены в таблице 1.

Табл.1 Варианты заданий для выполнения лабораторной работы.

Вар.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Символ	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
№строки	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3
№позции в строке	1	2	3	4	5	6	7	8	2	3	4	5	6	7	8

Пример программы ввода с клавиатуры символов 16-ричной системы счисления и вывода их на LCD стенда EV8031/AVR приведен ниже.

```
/* Для стенда EW8031/AVR(V3.2)*/
/*Программа обслуживания клавиатуры
и  индикации на ЖКИ.
С клавиатуры получаем коды клавиш от 0 до 9
и коды 16сс A,B,C,D,E,F.
Код A получаем двумя нажатиями кнопок вначале нажать и отпустить
 кнопку # а затем например 1(нажать и отпустить) получим символ A
 итого # 1 -A;#4 -B;#7 -C;#3 -D;#8 -E;#9 -F. */
#include<reg51.h>
#include<stdio.h>
#define uw 0x80
unsigned char cnt=0x01;
at 0x9003 unsigned char xdata s3s12;
at 0x9005 unsigned char xdata s2s11;
at 0x9006 unsigned char xdata s1s10;
at 0xa000 unsigned char xdata ind;
unsigned char cw=0xfc;
```

```
unsigned char codw=0x20;
unsigned char klav;
unsigned char klav1;
unsigned char sim;
unsigned char sim1;
        signed short l,k;
        at 0x8004 unsigned char xdata DC;
        at 0x8005 unsigned char xdata DD;
        void pausa(short l)
        {
        for(k=l;k>=0;k--);
        }
        //инициализация ЖКИ
        void start_gki()
        {
        l=0x0535;
        pausa(l);//пауза 20мс
        DC=0x30;
        l=0x029a;//пауза 10мс
        pausa(l);
        DC=0x30;
        l=0x0040;
        pausa(l);
        DC=0x30;
        l=0x0a6a;//пауза 40мс
        pausa(l);
        DC=0x38;
        l=0x0a6a;//пауза40мс
        pausa(l);
        DC=0x0e;
        l=0x014e;//пауза 5мс
        pausa(l);
        DC=0x06;
        l=0x014e;
        pausa(l);
        DC=0x01;
        l=0x029a;
        pausa(l);//пауза 10мс
        }
        void gki_out()
        {
        if(cnt==11){DC=0xc0;l=0x014e;pausa(l);};
        if(cnt==21){DC=0x8a;l=0x014e;pausa(l);};
        if(cnt==31){DC=0xca;l=0x014e;pausa(l);};
        DD=sim1;l=0x029;pausa(l);cnt++;
  if(cnt==41){DC=0x01;l=0x014e;pausa(l);cnt=1;};
        }
 void main ()
{
start_gki();
m1: klav=s3s12;
klav1=klav|0xf0;
```

```
        if(klav1==0xff)goto m2; else {l=0x0a6a; pausa(l);};
        klav=s3s12;
        klav1=klav|0xf0;
        if(klav1==0xf7)goto k0;
        if(klav1==0xfe){sim=0x03;sim1=0x33; goto z1;}//Символ "3"
        else if(klav1==0xfd){sim=0x06;sim1=0x36;goto z1;}//Символ"6"
        else if(klav1==0xfb){sim=0x09;sim1=0x39;goto z1;}//Символ "9"
        else goto m1;
        z1:ind=sim;
        gki_out();
        // Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
        l=0x0a6a;
        pausa(l);
        goto m1;
        m2:
        klav=s2s11;
        klav1=klav|0xf0;
        if(klav1==0xff)goto m3; else {l=0x0a6a; pausa(l);};
        if(klav1==0xf7){sim=0x00;sim1=0x30;goto z2;}//Символ "0"
        if(klav1==0xfe){sim=0x02;sim1=0x32;goto z2;}//Символ"2"
        if(klav1==0xfd){sim=0x05;sim1=0x35;goto z2;}//символ"5"
        if(klav1==0xfb){sim=0x08;sim1=0x38;goto  z2;}//Символ"8"
        else goto m1;
        z2:
        ind=sim;
        gki_out();
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s2s11;klav1=klav|0xf0;}
        l=0x0a6a;
        pausa(l);
        goto m1;
        m3:
        klav=s1s10;
        klav1=klav|0xf0;
        if(klav1==0xff)goto m1; else {l=0x0a6a; pausa(l);};
        if(klav1==0xfe){sim=0x01;sim1=0x31;goto z3;}//Символ"1"
        if(klav1==0xfd){sim=0x04;sim1=0x34;goto z3;}//Символ"4"
        if(klav1==0xfb){sim=0x07;sim1=0x37;goto z3;}//Символ"7"
        if(klav1==0xf7){sim=0x20;sim1=0x20;goto z3;}//символ пробел
        z3:
        ind=sim;
        gki_out();
        //Определение факта отпускания клавиши
        while(klav1!=0xff){klav=s1s10;klav1=klav|0xf0;}
        l=0x0a6a;
        pausa(l);
        goto m1;
//*************************************************************************
***
        //Была нажата клавиша реш.'#'
        k0:
```

```
while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
l=0x0a6a;
pausa(l);
k1:
klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xff)goto am1; else {l=0x0a6a; pausa(l);};
klav=s3s12;
klav1=klav|0xf0;
if(klav1==0xfe){sim=0x0d;sim1=0x44;goto aut1;}//Символ"Д"
if(klav1==0xfd){sim=0x0e;sim1=0x45;goto aut1;}//Символ"E"
if(klav1==0xfb){sim=0x0f;sim1=0x46;goto aut1;}//Символ "F"
aut1:
ind=sim;
gki_out();
//Определение факта отпускания клавиши
while(klav1!=0xff){klav=s3s12;klav1=klav|0xf0;}
l=0x0a6a;
pausa(l);
goto m1;
am1:
klav=s1s10;
klav1=klav|0xf0;
if(klav1==0xff)goto k1; else {l=0x0a6a; pausa(l);};
klav=s1s10;
klav1=klav|0xf0;
if(klav1==0xfe){sim=0x0a;sim1=0x41;goto aut2;}//символ"A"
if(klav1==0xfd){sim=0x0b;sim1=0x42;goto aut2;}//Символ"B"
if(klav1==0xfb){sim=0x0c;sim1=0x43;goto aut2;}//Символ"C"
else goto k0;
aut2:
ind=sim;
gki_out();
//Определение факта отпускания клавиши
while(klav1!=0xff){klav=s1s10;klav1=klav|0xf0;}
//Устранение дребезга контактов
l=0x0a6a;
pausa(l);
goto m1;
}
```

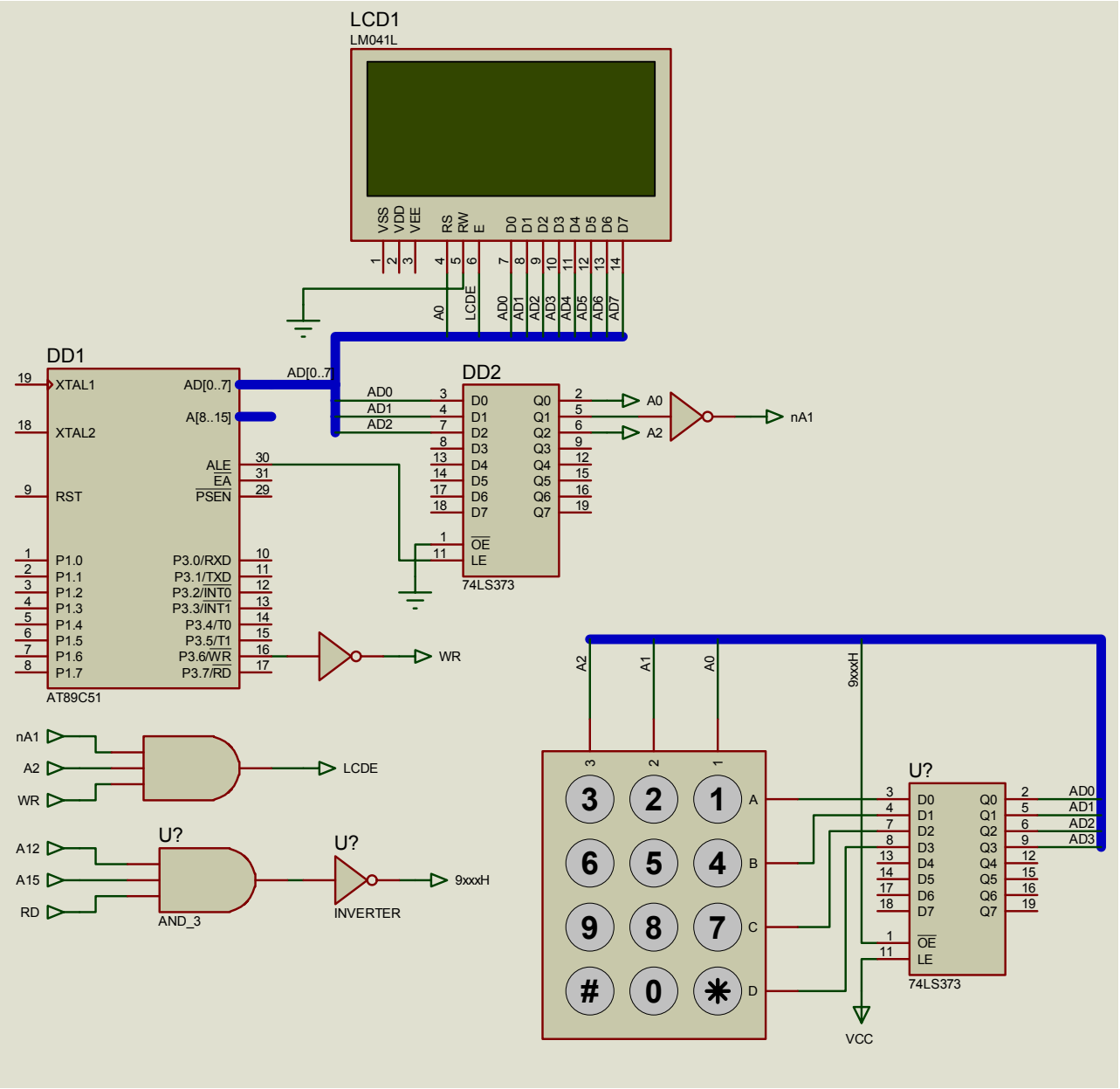


Рис.1 Модель схемы лабораторной работы.

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему клавиатуры и ЖКИ-индикации стенда EV8031/AVR.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

Контрольные вопросы.

1. Перечислить имена всех функций входящих в состав программы обслуживания клавиатуры и LCD лабораторной работы.
2. Кратко описать действия выполняемые каждой функцией перечисленной в пункте 1.
3. Привести схему сопряжения клавиатуры и LCD с системной шиной процессора стенда EV8031.
4. Записать функцию PAUSA для отработки выдержки времени 25мс.
5. Записать функцию вывода символа SIM1 в позицию 8 второй строки LCD.
6. Записать программу сканирования колонки клавиатуры и переменной SIM1 присвоить код АСК-2 нажатой клавиши 3 этой колонки.

1.

Лабораторная работа №9

Тема: Изучение принципа программного управления графическими ЖКИ

Цель работы : Изучить устройство, систему команд и программное управление графическими ЖКИ

Краткие теоретические сведения.

Таблица 1. Назначение выводов, используемого граф. ЖКИ

Вывод	Обозначение	Назначение вывода
1	GND	Общий вывод (0V)
2	VCC	Напряжение питания (5V)
3	NC	Не используется
4	A0	Адресный сигнал - выбор между передачей данных и команд управления
5	E1	Разрешение обращений к модулю (а также строб данных)(1 кристалл)
6	E2	Разрешение обращений к модулю (а также строб данных)(2 кристалл)
7	NC	Не используется
8	NC	Не используется
9	RD/WR	Выбор режима записи или чтения
10	DB0	Шина данных (младший бит)
11	DB1	Шина данных
12	DB2	Шина данных
13	DB3	Шина данных
14	DB4	Шина данных
15	DB5	Шина данных
16	DB6	Шина данных
17	DB7	Шина данных (старший бит)
18	RES	Сброс / Выбор типа интерфейса
19	+LED	+ Питание подсветки
20	-LED	− Питание подсветки

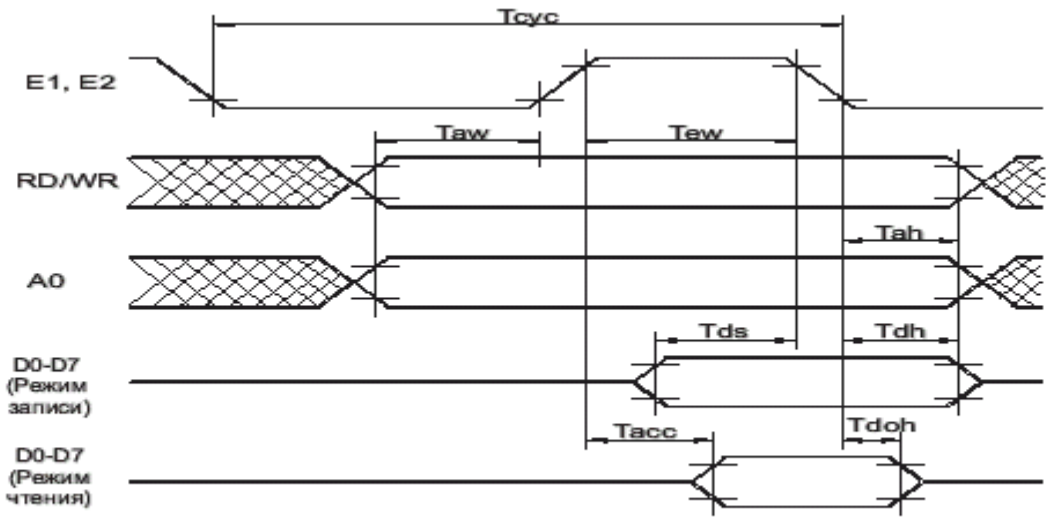


Рисунок 1. Временная диаграмма сигналов обмена через интерфейс ЖКИ

Таблица 2. Перечень команд модулей.

Команда	RD/WR	A0	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Функция
Display ON/OFF	0	0	1	0	1	0	1	1	1	0/1	Включает или выключает ЖКИ, независимо от данных в экранном ОЗУ и внутреннего состояния "1"-включить дисплей "0"-выключить дисплей
Display START Line	0	0	1	1	0	Display START Line (0...31)					Определяет строку ОЗУ , которая будет отображаться в верхней строке ЖКИ (Стартовая строка ЖКИ).
Set Page	0	0	1	0	1	1	1	0	Page (0...3)		Устанавливает страницу ОЗУ в режиме адреса страницы (стр. 0...3)
Set Address	0	0	0	Column address (0...79)							Устанавливает столбец ОЗУ в режиме адреса столбца
Status Read	1	0	BUSY	ADC	ON/OFF	RESET	0	0	0	0	Чтение режима состояния: BUSY: 1 – модуль занят внутренней обработкой; 0 – модуль готов к работе с внешним МП ADC: 1 – вывод прямых данных; 0 – вывод обратных данных ON/OFF: 1 – ЖКИ выключен; 0 – ЖКИ включен RESET: 1 – состояние сброса; 0 – нормальное состояние
Write Display Data	0	1	Write Data								Запись данных в ОЗУ модуля. Эти команды выбирают ОЗУ по ранее заданному адресу, после чего адрес столбца инкрементируется
Read Display Data	1	1	Read Data								Чтение данных из ОЗУ модуля. Эти команды выбирают ОЗУ по ранее заданному адресу, после чего адрес столбца инкрементируется
ADC Select	0	0	1	0	1	0	0	0	0	0/1	Используется для изменения в обратном направлении соответствия между адресом столбца и позиции на индикаторе: 0 – прямое соответствие 1 – обратное соответствие
Static Drive ON/OFF	0	0	1	0	1	0	0	1	0	0/1	Выбор статического или нормального режима управления: 1 – статическое управление (малого потребления) 0 – обычное управление
Duty Select	0	0	1	0	1	0	1	0	0	0/1	Выбор мультиплекса: 1 – Для модуля МТ-12232B
Read Modify Write	0	0	1	1	1	0	0	0	0	0	По этой команде устанавливается флаг RMW, после чего инкрементируется адрес счетчика столбца при записи данных в ОЗУ (и не инкрементируется при чтении)
END	0	0	1	1	1	0	1	1	1	0	Снятие флага RMW
RESET	0	0	1	1	1	0	0	0	1	0	Стартовая строка ЖКИ (Display Start Line) сбрасывается в 0, адрес страницы устанавливается равным 0, содержимое ОЗУ не изменяется

Начальная установка модуля

Для начальной установки модуля необходимо выполнить следующие действия:

1. после подачи напряжения питания удерживать вывод RES в состоянии логического “0” еще не менее 10 мкс;
2. подать перепад на вывод RES с логического “0” в логическую “1”, длительность фронта не более 10 мкс;
3. ожидать сброса бита RESET в байте состояния или выждать не менее 2 мс;
4. подать команду снятия флага RMW (END);
5. подать команду включения обычного режима работы (Static Drive ON/OFF);
6. подать команду выбора мультиплекса (Duty Select);
7. подать команду включения дисплея (Display ON/OFF).

Распределение ОЗУ Режимы отображения

Модуль MT_12232B содержит два кристалла, которые управляют двумя половинами отображаемого поля точек (левая половина и правая). Модуль содержит ОЗУ для хранения данных, выводимых на ЖКИ, размером 80x32 бит (80x32 бит на каждый кристалл). Все ОЗУ разбито на 4 страницы размером по 80x8 бит каждая. Каждая страница ОЗУ имеет организацию 80x8 бит. а ЖКИ отображаются только 61 байт из 80 из каждой страницы. Одновременно отображается четыре страницы: верхние 8 точек по вертикали соответствуют нулевой странице, нижние 8_ третьей (если при начальной установке была выбрана нулевая начальная строка отображения). Это можно изменить командой "Display START Line". Левые 61 точки по горизонтали выводит первый кристалл, правые 61 точки _ второй кристалл.

Модуль имеет два режима отображения информации из внутреннего ОЗУ: прямой и обратный. Он различается местоположением на ЖКИ первого отображаемого байта и направлением увеличения адреса во внутреннем ОЗУ при смещении отображаемой позиции на ЖКИ. В обратном режиме отображения адрес во внутреннем ОЗУ увеличивается при перемещении отображаемой позиции на ЖКИ вправо. В режиме он наоборот уменьшается. Режим работы выбирается командой "ADC Select". Каждой светящейся точке на ЖКИ соответствует логическая "1" в ячейке ОЗУ модуля. Соответствие между ячейками ОЗУ модуля и отображаемыми точками на ЖКИ показано на рис. 5. Н прямом Чтение (запись) информации из (в) модуль осуществляется по страницам (80x8 бит или 80x1 байт). Каждая страница представлена как 80 байт. Страницы не пересекаются. Адреса с 80 по 127 не используются, в них невозможно ничего записать, а при чтении по этим адресам на шине данных может присутствовать любая информация.

Для чтения или записи байта данных по произвольному адресу необходимо предварительно установить страницу ОЗУ и выбрать столбец внутри страницы ОЗУ. Это осуществляется командами "Set Page" и "Set Address" соответственно. После этого можно прочитать или записать байт данных. Одной команды "Set Page" недостаточно, так как она не изменяет адрес столбца. Для упрощения программ модули поддерживают также непрерывную последовательность операций чтения или записи (а также их комбинацию, см. ниже): после чтения (записи) одного байта счетчик столбца автоматически увеличивается на 1 и модули готовы к новой операции чтения (записи) по следующему адресу без предварительной установки страницы ОЗУ и адреса столбца. Счетчик столбца считает только внутри одной страницы! При достижении адреса 79 следующим значением счетчика будет 80 и т.д., то есть не происходит ни перехода на следующую страницу, ни сброса счетчика в 0. Таким образом, после чтения (записи) последнего байта

данных по адресу 79 модули прекратят прием (выдачу) информации. Для модуля MT_12232B также не происходит переход через середину отображаемых точек по горизонтали: левая и правая половина поля точек совершенно независимы и выдаются на ЖК панель из разных кристаллов.
В режиме чтения информации после команд "Set Page" и "Set Address", необходимо однократно выполнить “пустую” операцию чтения, результат которой не использовать.
Модуль поддерживает специальный режим увеличения счетчика адреса столбца только при записи. Это удобно для изменения информации в ОЗУ модулей: можно сначала прочитать данные, изменить их и записать в модули по тому же адресу (без повторной установки адреса столбца для операции записи). После операции записи будет выполнен переход к следующему байту данных. Этот режим включается командой "Read Modify Write" и выключается командой "END”

Чтение и запись данных

Вертикальное смещение отображаемой информации

Модуль поддерживает команду "Display START Line", устанавливающую номер самой верхней отображаемой строки. Это позволяет реализовать плавный сдвиг информации на ЖКИ по вертикали изменением номера первой отображаемой строки. Номер может быть в интервале от 0 до 31, что соответствует интервалу от первой строки нулевой страницы ОЗУ до последней строки третьей страницы ОЗУ. После отображения последней строки (31) будет отображаться снова нулевая строка.

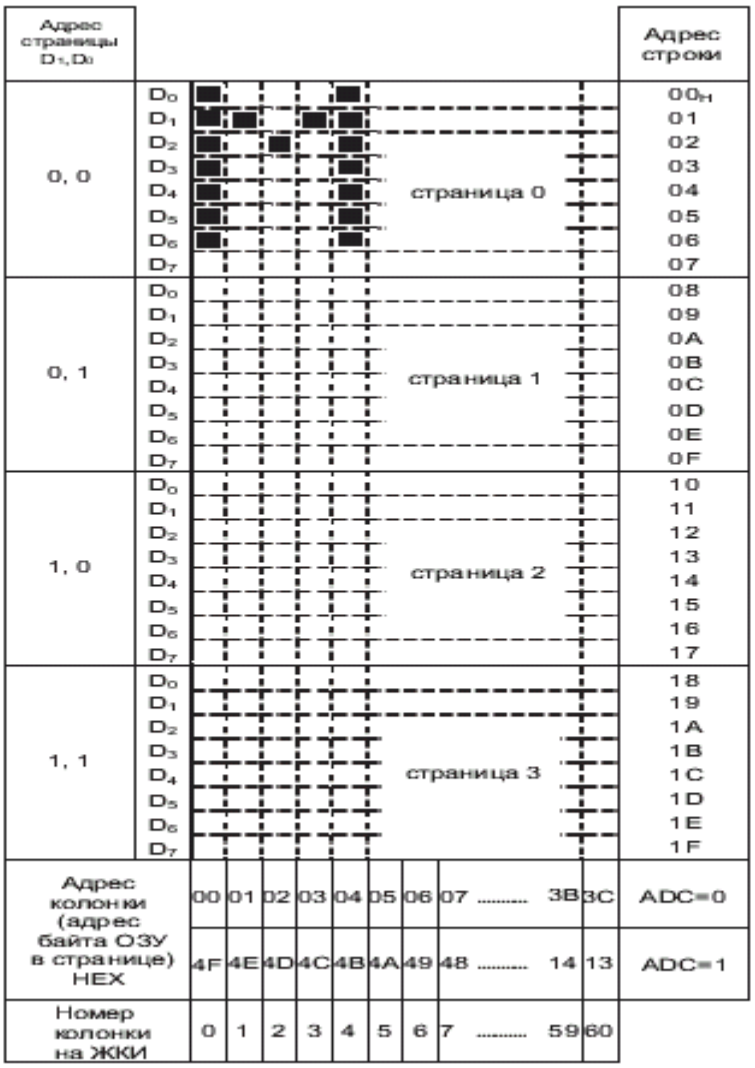


Рис.2. Соответствие между адресами ОЗУ модуля и отображаемыми точками на ЖКИ.

Задание для проведения лабораторной работы.

Составить программу выдачи заданного образа из внешней оперативной памяти согласно варианту задания. Использовать модель схемы в PROTEUS.

Таблица вариантов заданий

№ вар задания	1	2	3	4	5	6	7	8	9	10
Геом. фигура	Квадрат	Ромб	Прямо-угольник	Треу-гольник	Треуголь-ник прям	Квад рат	ромб	Прямо-угольник	Треу-гольник	Треуг ник п
Размер изображ	4 стр	4 стр	4 стр	4 стр	4 стр	2 стр	2 стр	2 стр	2 стр	2 с
Поле лев или правое	левое	левое	левое	левое	левое	правое	правое	правое	правое	право

```
 /*пРОГРАММА ВЫВОДА ГРАФИЧЕСКОГО ОБРАЗА И  ЦИФРОВОГО ТЕКСТА НА ГРАФИЧЕСКИЙ ЖКИ*/
/*Программа выводит на графический ЖКИ графический образ  из внешней оперативной памяти массив tbl1, состоящий из 240 байтов (4 страницы по 60 байтов) наложением на графический образ выводится цифровой текст из массива TBL[], состоящий из m символов 16сс*/
#include <reg51.h>
#define p_d P1
/*Таблица цифровых символов выводимых на ЖКИ */
at 0x0010 unsigned char xdata tbl[]={0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,
0x08,0x09,0x0a,0x0b,0x0d,0x0c,0x0e,0x0f};
/*Таблица кодов колонок 16 цифровых символов (0,1,2,3,....9,A,B,C,D,E,F) ДЛЯ HTG1*/
unsigned char code codtbl[]=
{0x3e,0x51,0x49,0x45,0x3e,
0x00,0x00,0x42,0x7f,0x40,
0x46,0x61,0x51,0x49,0x46,
0x21,0x41,0x45,0x4b,0x31,
0x18,0x14,0x12,0x7f,0x10,
0x2f,0x45,0x45,0x45,0x39,
0x3c,0x4a,0x49,0x49,0x30,
0x03,0x01,0x71,0x09,0x07,
0x36,0x49,0x49,0x49,0x36,
0x06,0x49,0x49,0x29,0x1e,
0x7e,0x11,0x11,0x11,0x7e,
0x7f,0x49,0x49,0x49,0x36,
0x3e,0x41,0x41,0x41,0x22,
0x7f,0x41,0x41,0x22,0x1c,
0x7f,0x49,0x49,0x49,0x41,
0x7f,0x09,0x09,0x09,0x01,
0x00,0x00,0x00,0x00,0x00};
at 0x0100 unsigned char xdata tbl1[]=
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
00,
0x80,0x80,0x40,0x40,0x20,0x20,0x10,0x10,0x08,0x08,0x04,0x04,0x02,0x02,0x0
1,
0x01,0x02,0x02,0x04,0x04,0x08,0x08,0x10,0x10,0x20,0x20,0x40,0x40,0x80,0x8
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,//str0
0x80,0x80,0x40,0x40,0x20,0x20,0x10,0x10,0x08,0x08,0x04,0x04,0x02,0x02,0x0
1,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
```

```
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x01,0x02,0x02,0x04,0x04,0x08,0x08,0x10,0x10,0x20,0x20,0x40,0x40,0x80,0x8
0,//str1
0x01,0x02,0x02,0x04,0x04,0x08,0x08,0x10,0x10,0x20,0x20,0x40,0x40,0x80,0x8
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x80,0x80,0x40,0x40,0x20,0x20,0x10,0x10,0x08,0x08,0x04,0x04,0x02,0x02,0x0
1,//str2
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x01,0x02,0x02,0x04,0x04,0x08,0x08,0x10,0x10,0x20,0x20,0x40,0x40,0x80,0x8
0,
0x80,0x80,0x40,0x40,0x20,0x20,0x10,0x10,0x08,0x08,0x04,0x04,0x02,0x02,0x0
1,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0};//swtr3
/*at 0x0200 unsigned char xdata tabl2[]=
{0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x
00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,//str0
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x80,0x80,0x40,0x40,0x20,0x20,0x10,0x10,0x08,0x08,0x04,0x04,0x02,0x02,0x0
1,
0x01,0x02,0x02,0x04,0x04,0x08,0x08,0x10,0x10,0x20,0x20,0x40,0x40,0x80,0x8
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,//str1
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x01,0x02,0x02,0x04,0x04,0x08,0x08,0x10,0x10,0x20,0x20,0x40,0x40,0x80,0x8
0,
0x80,0x80,0x40,0x40,0x20,0x20,0x10,0x10,0x08,0x08,0x04,0x04,0x02,0x02,0x0
1,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,//str2
```

```
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0
0};//str3 */
 /*Описание переменных */
unsigned char adr_column,adr_page,tncol,ind,pr,pr1;
char i,n,j;
short m,k,x,l,k1,k2;
sbit a0=0xb0;
sbit e1=0xb1;
sbit e2=0xb2;
sbit swr=0xb3;
sbit reset=0xb4;
void pausa()
{
for(x=0x7fff;x>=0;x--){l++;l--;l=0;}
}
void gash()
{
/*установить номер заданной колонки и номер страницы*/
e1=0;
a0=0;
p_d=adr_page;//СТРАНИЦА
e1=1;
e1=0;
a0=1;
e1=0;
for(k1=0;k1<=m;k1++)
{p_d=0x00;
e1=1;
e1=0;}
}
void zasw()
{
swr=0;
e2=1;
e1=0;
a0=0;
p_d=adr_page;//СТРАНИЦА
e1=1;
e1=0;
```

```
a0=1;
for(k=0;k<=m;k++)
{p_d=tbl1[k];
e1=1;
e1=0;
}
}
/*void zasw1()
{
swr=0;
e2=1;
a0=0;
e1=0;
p_d=adr_page;//СТРАНИЦА
e1=1;
e1=0;
a0=1;
for(k2=0;k2<=m;k2++)
{ind=tabl2[k2];
p_d=ind;
e1=1;
e1=0;
}
}*/
/*Функция инициализации
..... */
void inic()
{
unsigned char i,x;//переменная x для удлинения цикла задержки
adr_column=0x3c;
adr_page=0xb8;
/*устанавка интерфеса в исходное состояние */
a0=0;
e1=1;
e2=1;
swr=0;
reset=1;
/*инициализация ЖКИ */
reset=0;
pausa();
reset=1;
pausa();
/*подать команду включения обычного режима static drive on/of*/
a0=0;
swr=0;
```

```
e1=0;
p_d=0xa4;//обычное управление
e1=1;
e1=0;//строб записи
/*подать команду выбор мультиплекса duty select 0xa9*/
p_d=0xa9;
e1=1;
e1=0;
p_d=0xaf; //включить ЖКИ
e1=1;
e1=0;
p_d=0xa1;//ADC SELECT обратное соответствие
e1=1;
e1=0;

/*подать команду включения обычного режима static drive on/of*/
e1=1;
a0=0;
swr=0;
e2=0;
p_d=0xa4;//обычное управление
e2=1;
e2=0;//строб записи
/*подать команду выбор мультиплексaduty select 0xa9*/
p_d=0xa9;
e2=1;
e2=0;
p_d=0xaf;//включить ЖКИ
e2=1;
e2=0;
p_d=0xa1;//ADC SELECT обратное соответствие
e2=1;
e2=0;
return ;
}
/*////////////////////////////////////////////////////////////////// */
void main()
{
z1:
m=255;
inic();
zasw();
pausa();
/*gash();
pausa();
```

```
zasw1();
pausa();*/
/*установить номер заданной колонки и номер страницы*/
z2:
/*установить номер заданной колонки и номер страницы*/
swr=0;
e1=0;
a0=0;
p_d=0xb9;//СТРАНИЦА
e1=1;
e1=0;
p_d=0xe0;//команда чтение-модификация-запись
e1=1;
e1=0;
for(j=0;j<=42;j++)
{
swr=1;//чтение
a0=1;
P1=0xff;//порт p1 на ввод
e1=1;//строб чтения
ind=p_d;//чтение из ОЗУ ЖКИ
e1=0;//строб чтения продолжение
e1=1;
ind=p_d;//повторное чтение из ОЗУ ЖКИ
e1=0;
swr=0;
p_d=ind;//запись в ОЗУ не МОДИФИЦИРОВАННОГО КОДА
e1=1;//строб записи
e1=0;
}
a0=0;
swr=0;
p_d=0xee;//END завершение действия команды ЧТЕНИЕ-МОДИФИКАЦИЯ-
ЗАПИСЬ
e1=1;
e1=0;
/* Выдача текста на экран ЖКИ*/
m3:
pr=tbl[i];
if(pr==0x20)pr=16;
pr1=pr*5;
m1:
ind=codtbl[pr1+n];
n++;
a0=1;
```

```
p_d=ind;
e1=1;
e1=0;
if(n<=4)goto m3;else
{
p_d=00;//пустая колонка между символами
e1=1;
e1=0;
a0=0;
}
i++;
if(i==4)goto m4;
n=0;
goto m3;
m4:
while(1);
}
```

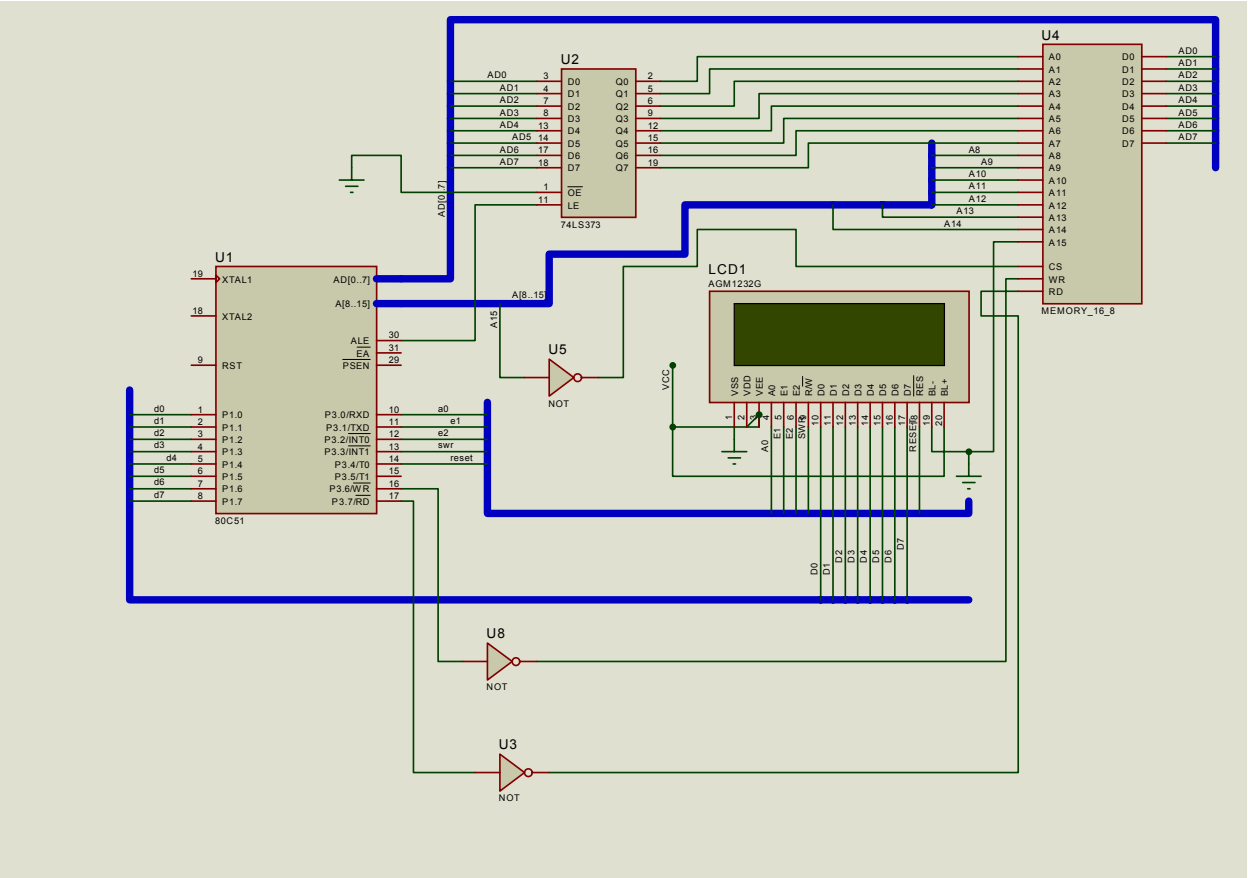


Рисунок 3 Принципиальная схема включения граф. ЖКИ и оперативной памяти к МК

Содержание отчета.

Отчет по лабораторной работе должен содержать:

1. Титульный лист.
2. Цель и задачи работы.
3. Схему матричной индикации EV8031.
4. Тексты индивидуального задания и программы согласно задания на языке СИ.
5. Выводы по работе.

СПИСОК РЕКОМЕНДОВАННОЙ ЛИТЕРАТУРЫ

1. Бродин В.Б., Шагурин И.И. Микроконтроллеры. Архитектура, программирование, интерфейс.–М.: Издательство ЭКОМ, 1999.–400с.
2. Фрунзе А.В. Микроконтроллеры? Это же просто! Т1,Т2,Т3.–М.: ООО «ИД СКИМЕН».
3. Микушин А.В. Занимательно о микроконтроллерах. – СПб.: БХВ-Петербург, 2006. – 432 с.: ил.
4. Сташин В.В. и др. Проектирование цифровых устройств на однокристальных микроконтроллерах/ В.В.Сташин, А.В.Урусов, О.Ф.Мологонцева.–М.: Энергоатомиздат, 1990. – 224с.
5. Бирюков С.А. Применение интегральных микросхем серий ТТЛ. – М.: «Патриот», МП «Символ-Р», «Радио», 1992. – 120 с.