Форум по AVR

- не работает программа из примера про пролистывания меню
- sinaprog не работает
- Пароль к архивам на сайте
- Пароль
- HDD и прерывания - доработка программы из статьи /node/220

Все записи

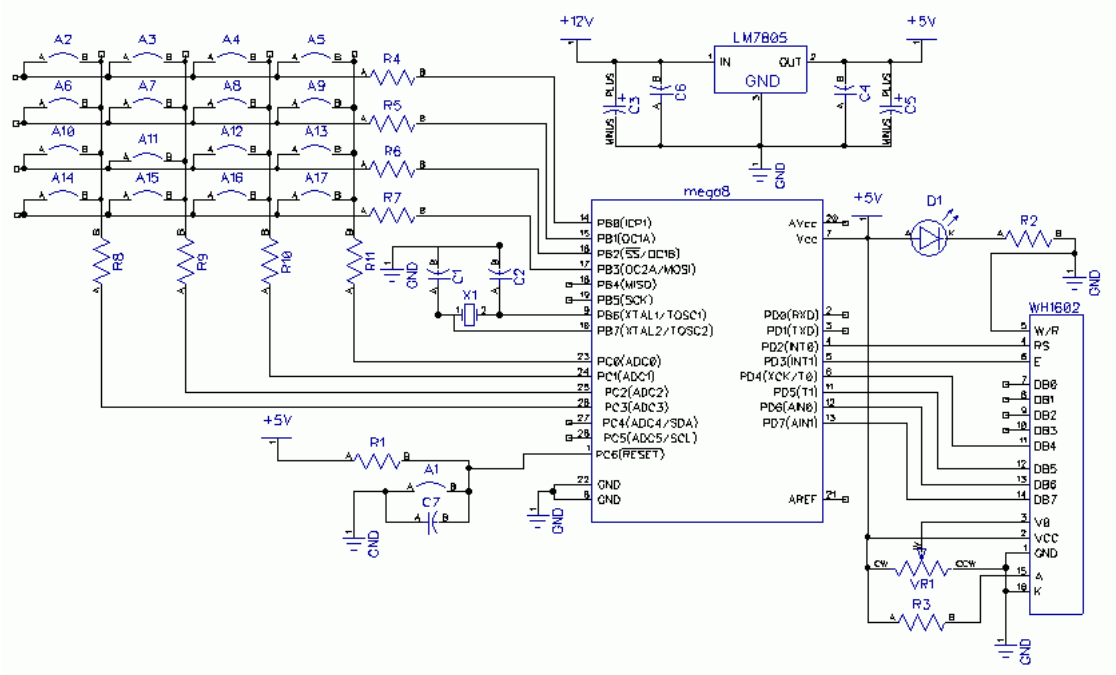
Главная страница » Каталог статей

Матричная клавиатура 4x4 динамическое считывание данных с портов

atmega8 | avr | keyboard | LCD | WH1602A | ЖКИ

В прошлой своей заметке я описывал **считывание одной клавиши** с помощью **микроконтроллера ATmega8** и вывод на **ЖКИ**. Написанную выше программу можно легко модифицировать на большее количество кнопок. Следует подсоединить клавиши к свободным выводам и периодически их всех опрашивать.

Но при большом количестве клавиш не хватает ножек. Можно перейти на более «многоногие» контроллеры, но они, как правило, стоят больших денег и в некоторых случаях приходиться заново переписывать проект под них. Рассмотрим решение проблемы нехватки ножек при считывании с клавиатуры, так называемое динамическое считывание. Например, мы имеем 16 клавиш, и хотим считать с них информацию. Хитро соединим их в клавиатурную матрицу, как показано на рисунке ниже:



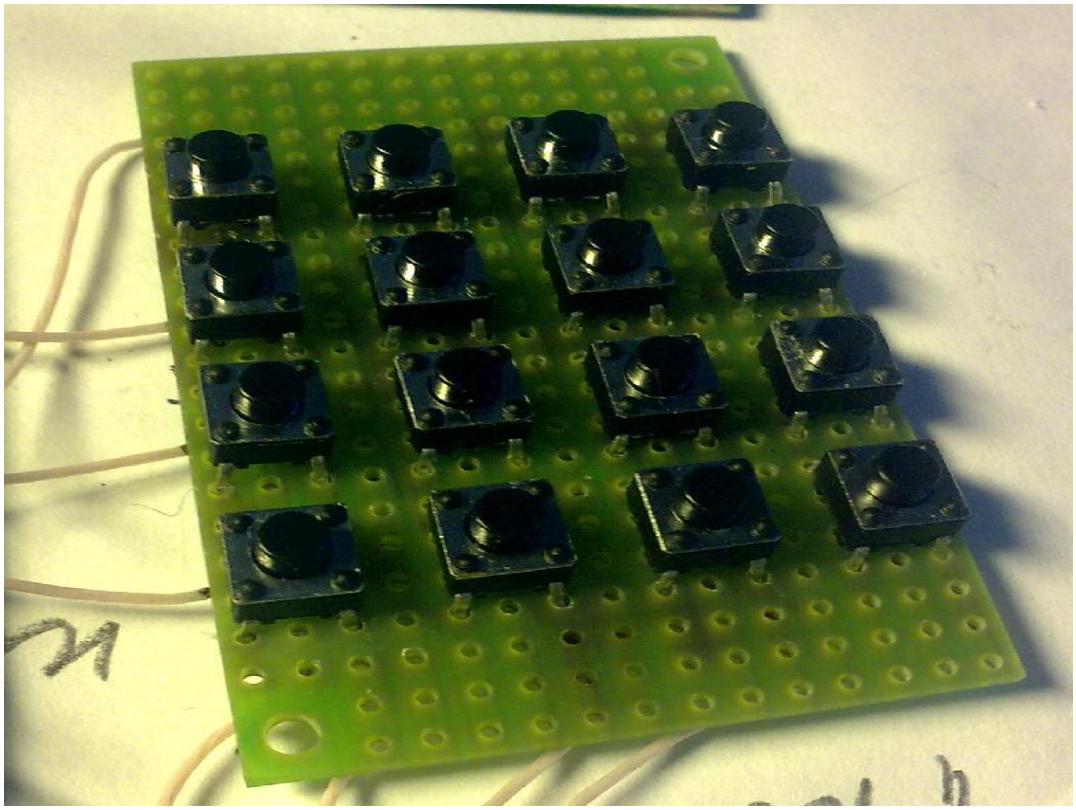
Со схемы видно, что для считывания 16-ти клавиш нам потребуется 8 выводов. По сравнению с обычным подключением (один пин – одна кнопка) мы выиграем в 16/8=2 раза. Алгоритм считывания будет таков: конфигурируем выводы PB0-PB3 как выхода и подадим на них нулевой уровень, а выводы PC0-PC3, выставим как входа и будем смотреть в какой колонке находиться нажатая клавиша. Далее поменяем все местами. Порты PC0-PC3 будут выходами с нулевым уровнем, а на портах PB0-PB3 будет считывать строку, в которой находиться нажатая клавиша. Зная строку и колонку можно однозначно вычислить клавишу, которая была нажата. Каждой клавише сопоставим код, который будет храниться в 2х мерном массиве 4x4 (строка нажатой клавиши – первый индекс массива, колонка - второй):

```
1.      unsigned char key_code[4][4]=     {{'C','D','E','F'},
2.                                         {'B','3','6','9'},
3.                                         {'A','2','5','8'},
4.                                         {'0','1','4','7'}};
```

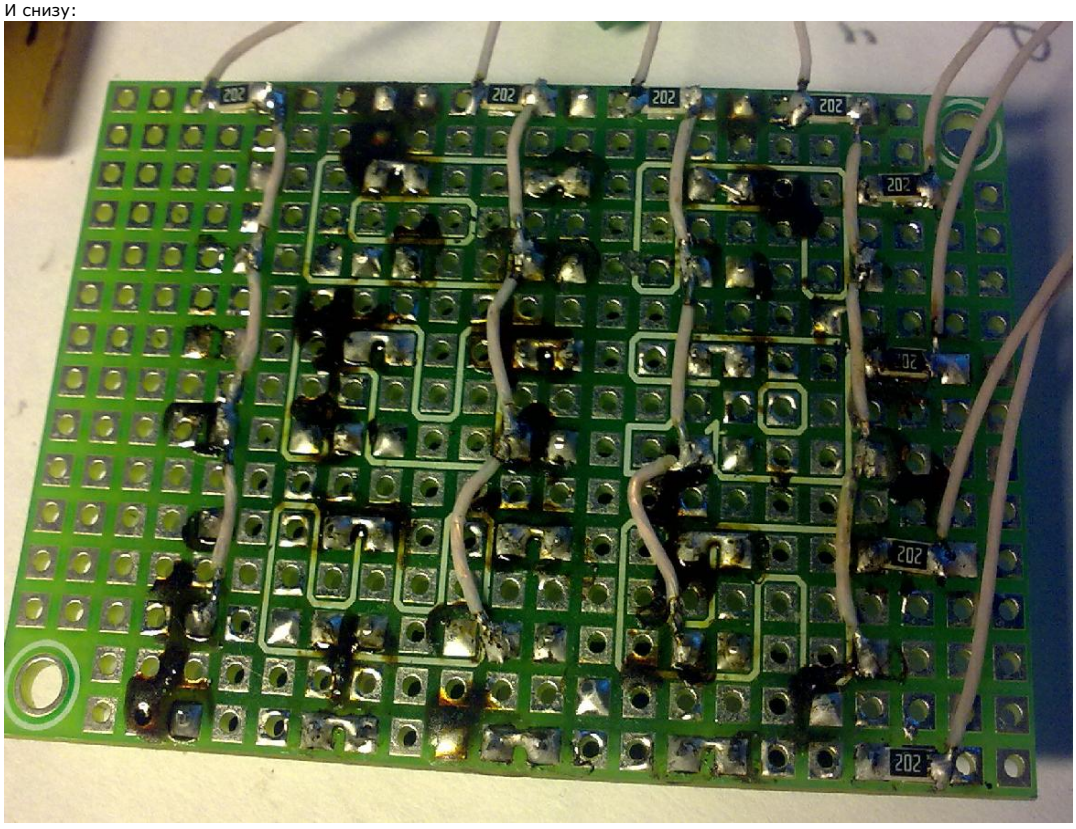
То есть, нажав клавишу во 2-й строке и 4-й колонке, микроконтроллер поймет, что была нажата клавиша «9» (массив будет отличаться, в зависимости от того, как подсоединить клавиатуру к микроконтроллеру). Далее все это будем **выводить на экран ЖКИ**. В схеме так же присутствуют ограничивающие резисторы R4-R11 номиналом 2 кОм, которые на «всякий пожарный случай» не дают пройти большому току через порт микроконтроллера. Клавиатуру проще всего изготовить из макетной платы и отдельных кнопок, соединив их короткими перемычками, она верой и правдой послужит в дальнейших опытах. Вид сверху:

Навига

- Авто
- Журн
  - Ж
  - Ж
  - Ра
- Ката
  - AV на ми AT
  - AV се AT
  - FT ви по ин
  - Fu ми AT
  - Fu ми At ко пр
  - LP Пр ми
  - SD па ми AV Ба
  - SD па ми AV Си Fa
  - US At пр ус ис пр
  - AL ми AT во
  - Ar ми AT
  - Бе AT пр ми
  - Вн EE 24 ми AV
  - Ге со US
  - Гр W ми AT
  - За тр дв
  - ИК ми
  - Ко RS дл ми с к
  - Ма от дл
  - Ма кл ди сч с к



И снизу:



Теперь перейдем к написанию программы:

```
#include <avr/io.h>//стандартная библиотека ввода/вывода
#include <avr/interrupt.h>//библиотека с прерываниями

#define RS 2            //RS=PD2 управляющий сигнал ЖКИ
#define E  3            //E=PD3  управляющий сигнал ЖКИ

#define TIME 10         //временная константа для ЖКИ
                        //Тактовая частота 4Mhz

unsigned char key_code[4][4]={{'C','D','E','F'},
                              {'B','3','6','9'},
                              {'A','2','5','8'},
                              {'0','1','4','7'}}; //матрица соответствия кодов клавиш

//Подпрограмма формирвоания задержки(паузы)
void pause (unsigned int a)
{ unsigned int i;

  for (i=a;i>0;i--);
}
```

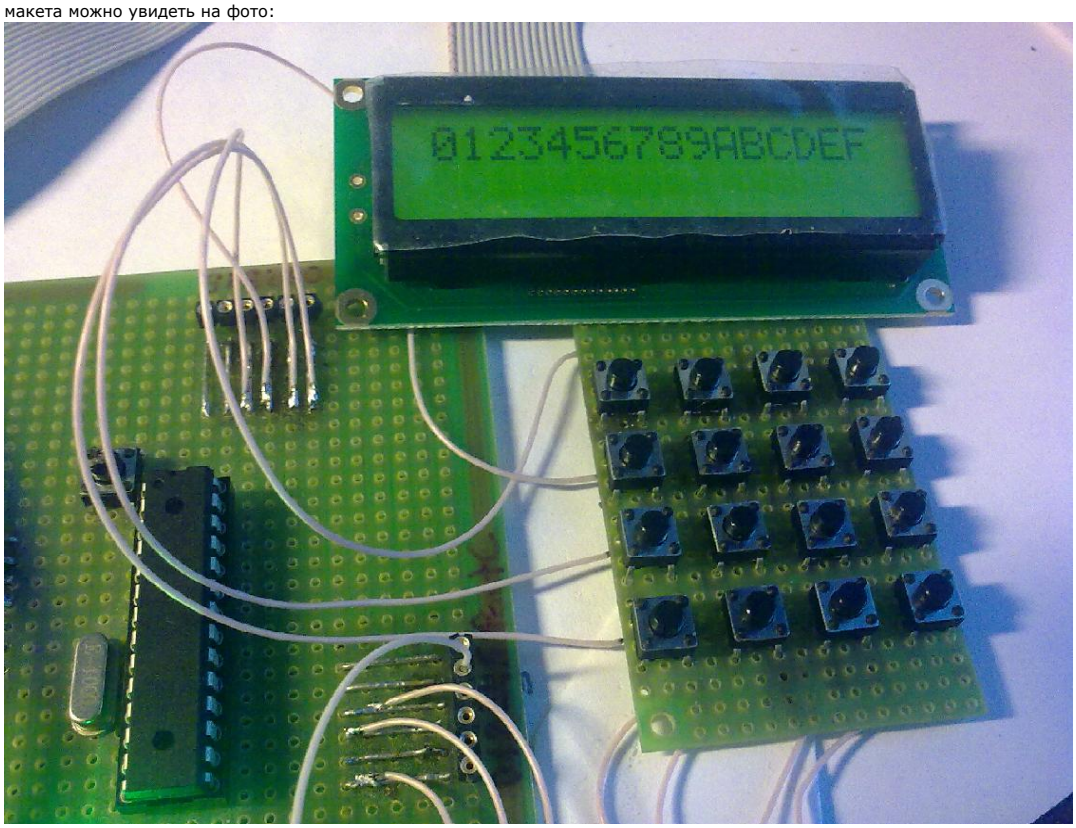
```
21.
22. //Подпрограмма передача команды в ЖКИ
23. void lcd_com (unsigned char lcd)
24. { unsigned char temp;
25.
26.   temp=(lcd&~(1<<RS))|(1<<E);   //RS=0 – это команда
27.   PORTD=temp;                   //Выводим на portD старшую тетраду команды, сигналы RS, E
28.    asm("nop");                  //Небольшая задержка в 1 такт МК, для стабилизации
29.   PORTD=temp&~(1<<E);           //Сигнал записи команды
30.
31.   temp=((lcd*16)&~(1<<RS))|(1<<E); //RS=0 – это команда
32.   PORTD=temp;                      //Выводим на portD младшую тетраду команды, сигналы RS, E
33.    asm("nop");                     //Небольшая задержка в 1 такт МК, для стабилизации
34.   PORTD=temp&~(1<<E);              //Сигнал записи команды
35.
36.   pause (10*TIME);              //Пауза для выполнения команды
37. }
38.
39. //Подпрограмма запись данных в ЖКИ
40. void lcd_dat (unsigned char lcd)
41. { unsigned char temp;
42.
43.   temp=(lcd|(1<<RS))|(1<<E);  //RS=1 – это данные
44.   PORTD=temp;                 //Выводим на portD старшую тетраду данных, сигналы RS, E
45.    asm("nop");          //Небольшая задержка в 1 такт МК, для стабилизации
46.   PORTD=temp&~(1<<E);   //Сигнал записи данных
47.
48.   temp=((lcd*16)|(1<<RS))|(1<<E);       //RS=1 – это данные
49.   PORTD=temp;           //Выводим на portD младшую тетраду данных, сигналы RS, E
50.    asm("nop");          //Небольшая задержка в 1 такт МК, для стабилизации
51.   PORTD=temp&~(1<<E);   //Сигнал записи данных
52.
53.   pause(TIME);                  //Пауза для вывода данных
54. }
55.
56. //Подпрограмма инициализация ЖКИ
57. void lcd_init (void)
58. {
59.  lcd_com(0x2c);         //4-проводный интерфейс, 5x8 размер символа
60.  pause(100*TIME);
61.  lcd_com(0x0c);         //Показать изображение, курсор не показывать
62.  pause(100*TIME);
63.  lcd_com(0x01);         //Очистить DDRAM и установить курсор на 0x00
64.  pause (100*TIME);
65. }
66.
67. //Подпрограмма инициализации таймера
68. void init_timer (void)
69. {
70.   TIMSK=(1<<TOIE0);                    //Разрешить прерывания по переполнению таймера0
71.   TCCR0=(1<<CS00)|(1<<CS01)|(0<<CS02);  //Делитель =/64
72. }
73.
74. //Подпрограмма-обработчик прерывания по переполнению таймера
75. ISR (TIMER0_OVF_vect)
76. { unsigned char i,j;
77.
78.   DDRC=0x00;    //конфигурируем порт C как вход
79.   PORTC=0x0F;   //выводим на 4 младших бита порта C лог. 1
80.   DDRB=0x0F;    //конфигурируем порт B как выход
81.   PORTB=0x00;   //обнуляем порт B
82.
83.   pause(10);    //Задержка для устранения всяких переходных процессов, важно ее не забыть!
84.   i=4;
85.   if ((PINC&0x01)==0x00) i=0;           //Если нажата клавиша в 0й колонке, i=0
86.   if ((PINC&0x02)==0x00) i=1;           //Если нажата клавиша в 1й колонке, i=1
87.   if ((PINC&0x04)==0x00) i=2;           //...
88.   if ((PINC&0x08)==0x00) i=3;           //Если нажата клавиша в 3й колонке, i=3
89.
90.   DDRC =0x0F;   //конфигурируем порт C как выход
91.   PORTC=0x00;   //обнуляем порт C
92.   DDRB =0x00;   //конфигурируем порт B как вход
93.   PORTB=0x0F;   //выводим на 4 младших бита порта B лог. 1
94.
95.   pause(10);    //Задержка для устранения всяких переходных процессов, важно ее не забыть!
96.   j=4;
97.   if ((PINB&0x01)==0x00) j=0;           //Если нажата клавиша в 0й строке, j=0
98.   if ((PINB&0x02)==0x00) j=1;           //...
99.   if ((PINB&0x04)==0x00) j=2;
100.  if ((PINB&0x08)==0x00) j=3;
101.
102.  if ((i!=4)&&(j!=4)) {                 //Если была нажата клавиша
103.                        while ((PINB&_BV(j))==0x00);    //Ждем пока кнопку отпустят
104.
105.                        lcd_dat(key_code[i][j]);        //Пишем ее код на ЖКИ
106.                      }
107.
108. TCNT0=0x00;             //Очищаем счетчик
109. TIFR=0x00;              //Очищаем флаг переполнения
110.
111. return;
112. }
113.
114. //Основная программа
115. int main(void)
116. {
117. DDRD=0xFC;                      //Инициируем PortD
118. PORTD=0x00;
119.
120. pause(1000);                    //Задержка для включения ЖКИ
121. lcd_init();                     //Инициализация ЖКИ
122. init_timer();                   //Инициализация нулевого таймера
123.
124. sei();                          //Глобальное разрешение прерываний
```

```
125.
126. while(1);                        //Вечный цикл
127.
128.
129. return 1;
130. }
```

Программа очень похожа на считывание одной клавиши, за исключением глобально объявленного массива key_code и функции обработки прерываний, которая проста, хорошо прокомментирована и словесно описывалась выше. Единственное, критично делать небольшую задержку, перед считыванием с порта, иначе всякие паразитные емкости и индуктивности могут внести помеху, и на экране будет отображаться ерунда. Если у вас после сборки не работают некоторые клавиши – следует проверить порты микроконтроллера на целостность. Сделать это просто скомпилировав простенькую программу проверки портов:

```
1. DDRB=0xFF;
2. while (1)
3. {
4. PORTB=0xFF;//Выводим лог. 1 на все 8 бит порта B
5. pause(32000);//пауза
6. PORTB=0x00;//Выводим лог. 0 на все 8 бит порта B
7. pause(32000);//пауза
8. }
```

И подсоединить светодиод к каждой ножке порта, если он замигает – все в порядке, если нет – порт дефектный. Внешний вид макета можно увидеть на фото:



Видео для демонстрации приведено ниже:

Atmega8 and keyboard connected LCD HD4478...

Но на этом не стоит ограничиваться. Зачастую введенные данные нужно обрабатывать, также следует предусмотреть возможность сброса введенного или удаление последнего символа. Реализуем макет, где с клавиатуры будет вводиться шестизначное целое число, при нажатии на клавишу «C» на экране **ЖКИ** должен мигать курсор, число его миганий равно числу, введенному с клавиатуры. Также клавиша «F» будет означать очистку экрана, а клавиша «E» - последнего введенного символа. Перейдем к написанию программы:

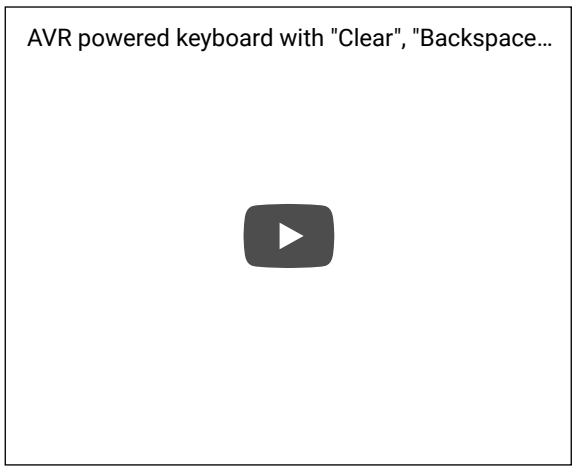
```
1. #include <avr/io.h>
2. #include <avr/interrupt.h>
3.
4. #define NUMBER_SIZE 6   //Максимальная разрядность числа
5.
```

```
#define RS 2            //RS=PD2
#define E  3            //E=PD3

#define TIME 10         //временная константа для ЖКИ
                        //Тактовая частота 4Mhz

unsigned char key_code[4][4]={{'C','D','E','F'},
                              {'B','3','6','9'},
                              {'A','2','5','8'},
                              {'0','1','4','7'}};       //Коды клавиш

unsigned char row[NUMBER_SIZE]={};      //массив, который содержит нажатые клавиши
unsigned char row_counter=0;            //количество нажатых клавиш

unsigned long int number=0;             //Число морганий курсора

void pause (unsigned int a)
{ unsigned int i;

  for (i=a;i>0;i--)
   ;
}

//Передача команды ЖКИ
void lcd_com (unsigned char lcd)
{ unsigned char temp;

  temp=(lcd&~(1<<RS))|(1<<E);   //RS=0 - это команда
  PORTD=temp;                   //Выводим на portD старшую тетраду команды, сигналы RS, E
   asm("nop");                  //Небольшая задержка в 1 такт МК, для стабилизации
  PORTD=temp&~(1<<E);       //Сигнал записи команды

  temp=((lcd*16)&~(1<<RS))|(1<<E);  //RS=0 - это команда
  PORTD=temp;                       //Выводим на portD младшую тетраду команды, сигналы RS, E
   asm("nop");                      //Небольшая задержка в 1 такт МК, для стабилизации
  PORTD=temp&~(1<<E);      //Сигнал записи команды

  pause (10*TIME);         //Пауза для выполнения команды
}

//Запись данных в ЖКИ
void lcd_dat (unsigned char lcd)
{ unsigned char temp;

  temp=(lcd|(1<<RS))|(1<<E);       //RS=1 - это данные
  PORTD=temp;                      //Выводим на portD старшую тетраду данных, сигналы RS, E
   asm("nop");                     //Небольшая задержка в 1 такт МК, для стабилизации
  PORTD=temp&~(1<<E);          //Сигнал записи данных

  temp=((lcd*16)|(1<<RS))|(1<<E);  //RS=1 - это данные
  PORTD=temp;                      //Выводим на portD младшую тетраду данных, сигналы RS, E
   asm("nop");                     //Небольшая задержка в 1 такт МК, для стабилизации
  PORTD=temp&~(1<<E);         //Сигнал записи данных

  pause(TIME);                //Пауза для вывода данных
}

//Иниализация ЖКИ
void lcd_init (void)
{
 lcd_com(0x2c);         //4-проводный интерфейс, 5x8 размер символа
 pause(100*TIME);
 lcd_com(0x0c);         //Показать изображение, курсор не показывать
 pause(100*TIME);
 lcd_com(0x01);         //Очистить DDRAM и установить курсор на 0x00
 pause (100*TIME);
}


void init_timer (void)
{
  TIMSK=(1<<TOIE0);                     //Разрешить прерывания по переполнению таймера0
  TCCR0=(1<<CS00)|(1<<CS01)|(0<<CS02);  //Делитель =/64
}

//Проверка, является ли x цифрой, если да результат =1, иначе результат =0
unsigned char is_digit (unsigned char x)
{
 if ((x>='0')&&(x<='9')) return 1;
                    else return 0;
}

//Конвертируем "row" в int значение number
void get_number(void)
{ unsigned char i;
 number=0;
  for (i=0;i<row_counter;i++)
   {
    number=number*10;
        number=number+row[i]-0x30;
   }
}

//Пишем row на ЖКИ
void write_row (void)
{ unsigned char i;

 lcd_com(0x86);                         //Перейдем в начало
  for (i=0;i<row_counter;i++)           //Пишем row
   lcd_dat(row[i]);
  for(i=row_counter;i<NUMBER_SIZE;i++)  //Очистим незаполненные ячейки
   lcd_dat(' ');
}
```

```
ISR (TIMER0_OVF_vect)
{ unsigned char i,j;

  DDRC=0x00;    //PortC как вход
  PORTC=0x0f;
  DDRB=0x0f;    //PortB как выход
  PORTB=0x00;

  pause(10);    //Задержка для устренения всяких переходных процессов, важно ее не забыть!
  i=4;
  if ((PINC&0x01)==0x00) i=0;           //Если нажата клавиша в 0й колонке, i=0
  if ((PINC&0x02)==0x00) i=1;           //...
  if ((PINC&0x04)==0x00) i=2;
  if ((PINC&0x08)==0x00) i=3;

  DDRC=0x0f;    //PortC как выход
  PORTC=0x00;
  DDRB=0x00;    //PortB как вход
  PORTB=0x0f;

  pause(10);    //Задержка для устренения всяких переходных процессов, важно ее не забыть!
  j=4;
  if ((PINB&0x01)==0x00) j=0;           //Если нажата клавиша в 0й строке, j=0
  if ((PINB&0x02)==0x00) j=1;           //...
  if ((PINB&0x04)==0x00) j=2;
  if ((PINB&0x08)==0x00) j=3;

  if ((i!=4)&&(j!=4))
{                       //Если была нажата клавиша
  while ((PINB&_BV(j))==0x00)   //Ждем отжатия
   ;

  if ((is_digit(key_code[i][j])==1)&&  //Нажата цифра и не достигнут лимит в NUMBER_SIZE
      (row_counter<NUMBER_SIZE)) {     //Добавить в row и увеличить row_counter
                                  row[row_counter]=key_code[i][j];
                                  row_counter++;
                                 }
  if (key_code[i][j]=='F') row_counter=0x00; //Если нажата 'F' очищаем row
  if ((key_code[i][j]=='E')&&
      (row_counter>0x00)) row_counter--;   //Если нажата 'E'
                                           //и есть введенные цифры – удалить последнюю
  if (key_code[i][j]=='C') {    //Если нажата 'C' тогда мигаем number раз
                             get_number();
                             while (number>0)           //Мигаем number раз
                              {
                               lcd_com(0xc0);
                               lcd_dat(0xff);
                               pause(32000);
                               pause(32000);
                               pause(32000);
                               pause(32000);
                               lcd_com(0xc0);
                               lcd_dat(' ');
                               pause(32000);
                               pause(32000);
                               pause(32000);
                               pause(32000);
                               number=number-1;
                              }
                                          }
   write_row();           //Пишем row на ЖКИ
}
TCNT0=0x00;             //Очищаем таймер и флаг переполнения
TIFR=0x00;

return;
}

int main(void)
{
DDRD=0xfc;                      //Инициализация PortD
PORTD=0x00;

pause(1000);                    //Задержка для включения ЖКИ
lcd_init();                     //Инициализация ЖКИ
init_timer();                   //Инициализация нулевого таймера
lcd_dat('D');                   //Напишем на ЖКИ "Data= "
lcd_dat('a');
lcd_dat('t');
lcd_dat('a');
lcd_dat('=');
lcd_dat(' ');

sei();                          //Разрешения прерываний

while(1)                        //Вечный цикл
 ;

return 1;
}
```

Номиналы резисторов:
R4-R11 - 2кОм,
R3 - 17 Ом,
R1, R2 - 1 кОм.

Теперь программа выглядит посерьезней. Добавился глобальный массив row в котором содержаться нажатые цифры и переменная row_count – их количество. Функция void write_row(void) выводит на **ЖКИ** содержимое row, а void get_number(void) – конвертирует массив цифр в целочисленную переменную number. Также появилась функция проверки, является ли x цифрой – is_digit(x). Во всем остальном код хорошо прокомментирован, и разобраться в нем не составит труда. Исходные коды программ, в виде проектов под **AVR Studio**
скачиваем здесь.
Видео работы:



» **Войдите** или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

вход-выход
Опубликовано Anonymous в Вт, 07/12/2011 - 11:11.

>> DDRC=0x00; //конфигурируем порт С как вход
>> PORTC=0x0F; //выводим на 4 младших бита порта С лог. 1

объясните пожалуйста, какая разница между входом и выходом, если на вход мы можем _выводить_ сигналы?

- -

Серега

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

>> DDRC=0x00; //конфигурируем
Опубликовано vakula в Вт, 07/12/2011 - 21:24.

>> DDRC=0x00; //конфигурируем порт С как вход
>> PORTC=0x0F; //выводим на 4 младших бита порта С лог. 1

В этом случае PC0-PC3 - входа с подключенным подтягивающим резистором.
Читать мануал на стр. 52, таблица 12-1

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

количество нажатых кнопок,а именно вывод этого количества на жки
Опубликовано Anonymous в Чт., 06/09/2011 - 18:34.

как вывести на жки количество нажатых(или зажатых)кнопок,например я зажал 7 кнопок,и мне нужно чтобы на жки вывелось 7,я зажал 16 и на жки вывелось 16????помогите пожайлуста,очень нужно!

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

Читай каммент
Опубликовано extremist в Чт., 06/09/2011 - 21:16.

Читай каммент ниже!
http://avrlab.com/node/85#comment-737

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

сайт регулярно глючит -
Опубликовано Anonymous в Суб, 02/26/2011 - 18:25.

сайт регулярно глючит - ошибки пхп, сбитая кодировка - куда прислать скриншот?
по теме: а что будет, если нажать сразу несколько кнопок одновременно?

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

Ага, глюки сам наблюдаю, буду
Опубликовано vakula в Суб, 02/26/2011 - 18:31.

Ага, глюки сам наблюдаю, буду лечить. Скрины можно на vakula_s_s at inbox.ru.
Если нажать одновременно несколько кнопок - заглючит, т.е. считается первая строка и первый столбец, на котором будет 0.

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

А как изменится матрица
Опубликовано foxit в Чт., 01/20/2011 - 15:55.

А как изменится матрица соответствия кнопок при изменении подключения кнопок?

»
Войдите или **зарегистрируйтесь**, чтобы получить возможность отправлять комментарии

Не до конца понял вопрос.
Опубликовано vakula в Чт., 01/20/2011 - 15:58.

Не до конца понял вопрос. Т.е. если перепутать правую с нижней стороной матрицы при подключении?

»

По какой-то причине у меня

Опубликовано foxit в Чт., 01/20/2011 - 16:13.

По какой-то причине у меня работают кнопки 1-9, а 0 и буквы не работают.
Пытаюсь найти причину.
Порты целые.Линии прозваниваются.

»

Взял новый контроллер и

Опубликовано foxit в Пн, 05/23/2011 - 21:10.

Взял новый контроллер и клавиатуру.
Проверил порты МК - целые.
Подключил все по-новому.
Прозвонил - все ок.
Все-равно один столбец не работает.
Уже не знаю что думать.
Прошу помощи.

»

Попробуй увеличить settle

Опубликовано vakula в Вт, 05/24/2011 - 10:38.

Попробуй увеличить settle delay (стр. 83, 95 в статье). Если не поможет - подключи проблемный столбец к земле вручную.
И пробей какие данные на порту МК, когда жмешь кнопки.

Прошивка с сайта, что-то в ней менял? Какая частота МК?

»

отсоединил

Опубликовано foxit в Вт, 05/24/2011 - 19:24.

отсоединил клавиатуру
перемычкой поочередно замыкал пины клавиатуры - на lcd отображаются соотв. цифры
с пина PC0 - нет ничего.
Если PC0 замкнуть на землю - на lcd появляется буква A.
кнопки подкючены вот так
unsigned char key_code[4][4]={{'A','B','C','D'},
{'3','6','9','#'},
{'2','5','8','0'},
{'1','4','7','*'}};

В программе ничего не менял
Частота 1 МГц

»

Если верить результатам

Опубликовано Anonymous в Вт, 05/24/2011 - 21:38.

Если верить результатам эксперимента и исповедовать традиционную логику - виноваты кнопки в неработающем столбце.

»

Та вряд ли, так как видео

Опубликовано extremist в Вт, 05/24/2011 - 21:24.

Та вряд ли, так как видео снято на этом коде.

»

На какой частоте МК работает?

Опубликовано vakula в Чт., 01/20/2011 - 17:05.

На какой частоте МК работает? Там надо не провтыкать с задержкой между сканированием столбцов и строчек. + если работают цифры и 0, то буквы обязаны работать

»

"0" не работает. Каким

Опубликовано foxit в Чт., 01/20/2011 - 19:09.

"0" не работает.
Каким кнопкам какие цифры и буквы назначены?

»

A2='0' A17='F' A14='7' Еще

Опубликовано vakula в Чт., 01/20/2011 - 23:05.

A2='0'
A17='F'
A14='7'

Еще раз проверь цепочки на PC0, PB0 наверняка в них дела

»

Проект под Мега16

Опубликовано Anonymous в Вт, 12/07/2010 - 21:43.

Доброго времени суток! Я начинаю разбираться с микроконтроллерами и хотелось бы освоить работу и с ЖК, и с клавиатурой... Помогите пожалуйста переделать проект под Мегу16 в протеусе. Заранее благодарен!

»

Протеус для новичков не

Опубликовано extremist в Вт, 12/07/2010 - 21:50.

Протеус для новичков не рекомендую, собирай все в железяке. Купи микроконтроллер, там собирать нечего, все просто. Так и наглядно будет и эффектней!
Что касается микроконтроллера, нет никакой разницы какой применять. Код даже не надо будет изменять просто необходимо будет подключить все в соответствии со схемой для ATmega8.

»

поясню ситуацию)

Опубликовано Anonymous в Вт, 12/07/2010 - 22:24.

Кроме вывода на дисплей, мне нужно ещё управлять напряжением примерно от 0 до 5 В(думаю использовать ЦАП mcp4921, но ещё не знаю как), потом принимать данные в виде напряжения до 2 В. И заранее задавать рабочую точку этого выходного напряжения и как-то реализовать обратную связь для корректировки и удержания этой рабочей точки на месте. Думал компаратор использовать, но не аналоговый, а оцифровывать сигнал(тот что 2 В). По-этому выбрал мегу16 у неё 4ре полноразрядных 8ми битных порта. Ещё думал можно ли использовать 2 микроконтроллера отдельно для вывода инфы на дисплей, а другой чтобы занимался сравнением и т.д.... Вот такие пироги)
А так уже и программатор аналог STK 500 раздобыл, и дисплей есть... но боюсь попалить что-нибудь....
Спасибо! Если будут какие-нибудь советы от опытных людей, то с радостью выслушаю!:)

»

Расчет ног для проекта

Опубликовано extremist в Ср, 12/08/2010 - 08:00.

На дисплей: 6 выводов
На АЦП(для обратной связи): 1 вывод
На ШИМ(для управления выходным напряжением): 1 вывод
Матричная клавиатура: 8 выводов
Итого: 16 выводов
При условии, что ноги АЦП и выхода ШИМ не будут пересекаться можно вполне применить ATmega16.
А если постараться то и ATmega8

»

Вопрос по поводу схемы?

Опубликовано Anonymous в Вс, 08/15/2010 - 11:42.

Подскажи пожалуйста где взять значения для гонденсаторов и резисторов C1-C6 и R1-R7

»

кодовый замок

Опубликовано foxit в Суб, 07/31/2010 - 22:55.

Помогите переделать проект под кодовый замок.
Как сделать изменение кода с клавиатуры?
Спасибо

»

Кодовый замок

Опубликовано vakula в Пн, 08/02/2010 - 09:33.

Давай условимся, что код у нас только цифровой, т.е. состоит из цифр от 0 до 9. В примере сделана клавиатура с кнопками "Backspace" и "Сброс", что облегчает задачу. Теперь нужно добавить функционал: "при нажатии клавиши "C" кодовый замок сверяет введенный код с кодом, разрешающим доступ, и если они совпадают - отпирает какое-то там реле и т.д.". Отобразим наши рассуждения в коде:

```
1. #include <avr/io.h>
2. #include <avr/interrupt.h>
3.
4. #define NUMBER_SIZE 6   //Максимальная разрядность числа
5.
6. #define RS 2            //RS=PD2
7. #define E  3            //E=PD3
8.
9. #define TIME 10         //временная константа для ЖКИ
10.                        //Тактовая частота 4Mhz
```

```
11.
12. #define CODE 123456     //Код нашего кодового замка
13.
14. unsigned char key_code[4][4]={{'C','D','E','F'},
15.                              {'B','3','6','9'},
16.                              {'A','2','5','8'},
17.                              {'0','1','4','7'}};       //Коды клавиш
18.
19. unsigned char row[NUMBER_SIZE]={};      //массив, который содержит нажатые клавиши
20. unsigned char row_counter=0;            //количество нажатых клавиш
21.
22. unsigned long int number=0;             //Число морганий курсора
23.
24. void pause (unsigned int a)
25. { unsigned int i;
26.
27.   for (i=a;i>0;i--)
28.    ;
29. }
30.
31. //Передача команды ЖКИ
32. void lcd_com (unsigned char lcd)
33. { unsigned char temp;
34.
35.   temp=(lcd&~(1<<RS))|(1<<E);   //RS=0 - это команда
36.   PORTD=temp;                   //Выводим на portD старшую тетраду команды, сигналы RS, E
37.    asm("nop");                  //Небольшая задержка в 1 такт МК, для стабилизации
38.   PORTD=temp&~(1<<E);       //Сигнал записи команды
39.
40.   temp=((lcd*16)&~(1<<RS))|(1<<E);  //RS=0 - это команда
41.   PORTD=temp;                       //Выводим на portD младшую тетраду команды, сигналы RS,
42.    asm("nop");                      //Небольшая задержка в 1 такт МК, для стабилизации
43.   PORTD=temp&~(1<<E);      //Сигнал записи команды
44.
45.   pause (10*TIME);         //Пауза для выполнения команды
46. }
47.
48. //Запись данных в ЖКИ
49. void lcd_dat (unsigned char lcd)
50. { unsigned char temp;
51.
52.   temp=(lcd|(1<<RS))|(1<<E);       //RS=1 - это данные
53.   PORTD=temp;                      //Выводим на portD старшую тетраду данных, сигналы RS, E
54.    asm("nop");                     //Небольшая задержка в 1 такт МК, для стабилизации
55.   PORTD=temp&~(1<<E);          //Сигнал записи данных
56.
57.   temp=((lcd*16)|(1<<RS))|(1<<E);  //RS=1 - это данные
58.   PORTD=temp;                      //Выводим на portD младшую тетраду данных, сигналы RS, E
59.    asm("nop");                     //Небольшая задержка в 1 такт МК, для стабилизации
60.   PORTD=temp&~(1<<E);         //Сигнал записи данных
61.
62.   pause(TIME);                //Пауза для вывода данных
63. }
64.
65. //Иниализация ЖКИ
66. void lcd_init (void)
67. {
68.  lcd_com(0x2c);         //4-проводный интерфейс, 5x8 размер символа
69.  pause(100*TIME);
70.  lcd_com(0x0c);         //Показать изображение, курсор не показывать
71.  pause(100*TIME);
72.  lcd_com(0x01);         //Очистить DDRAM и установить курсор на 0x00
73.  pause (100*TIME);
74. }
75.
76.
77. void init_timer (void)
78. {
79.   TIMSK=(1<<TOIE0);                    //Разрешить прерывания по переполнению таймера0
80.   TCCR0=(1<<CS00)|(1<<CS01)|(0<<CS02);  //Делитель =/64
81. }
82.
83. //Проверка, является ли x цифрой, если да результат =1, иначе результат =0
84. unsigned char is_digit (unsigned char x)
85. {
86.  if ((x>='0')&&(x<='9')) return 1;
87.                     else return 0;
88. }
89.
90. //Конвертируем "row" в int значение number
91. void get_number(void)
92. { unsigned char i;
93.  number=0;
94.   for (i=0;i<row_counter;i++)
95.    {
96.     number=number*10;
97.         number=number+row[i]-0x30;
98.    }
99. }
100.
101. //Пишем row на ЖКИ
102. void write_row (void)
103. { unsigned char i;
104.
105.  lcd_com(0x86);                         //Перейдем в начало
106.   for (i=0;i<row_counter;i++)           //Пишем row
107.    lcd_dat(row[i]);
108.   for(i=row_counter;i<NUMBER_SIZE;i++)  //Очистим незаполненные ячейки
109.    lcd_dat(' ');
110. }
111.
112. ISR (TIMER0_OVF_vect)
113. { unsigned char i,j;
114.
```

```
115.   DDRC=0x00;    //PortC как вход
116.   PORTC=0x0f;
117.   DDRB=0x0f;    //PortB как выход
118.   PORTB=0x00;
119.
120.   pause(10);    //Задержка для устренения всяких переходных процессов, важно ее не забыть!
121.   i=4;
122.   if ((PINC&0x01)==0x00) i=0;           //Если нажата клавиша в 0й колонке, i=0
123.   if ((PINC&0x02)==0x00) i=1;           //...
124.   if ((PINC&0x04)==0x00) i=2;
125.   if ((PINC&0x08)==0x00) i=3;
126.
127.   DDRC=0x0f;    //PortC как выход
128.   PORTC=0x00;
129.   DDRB=0x00;    //PortB как вход
130.   PORTB=0x0f;
131.
132.   pause(10);    //Задержка для устренения всяких переходных процессов, важно ее не забыть!
133.   j=4;
134.   if ((PINB&0x01)==0x00) j=0;           //Если нажата клавиша в 0й строке, j=0
135.   if ((PINB&0x02)==0x00) j=1;           //...
136.   if ((PINB&0x04)==0x00) j=2;
137.   if ((PINB&0x08)==0x00) j=3;
138.
139.   if ((i!=4)&&(j!=4))
140. {                       //Если была нажата клавиша
141.   while ((PINB&_BV(j))==0x00)   //Ждем отжатия
142.    ;
143.
144.   if ((is_digit(key_code[i][j]))==1)&&  //Нажата цифра и не достигнут лимит в NUMBER_SIZE
145.       (row_counter<NUMBER_SIZE)) {     //Добавить в row и увеличить row_counter
146.                                   row[row_counter]=key_code[i][j];
147.                                   row_counter++;
148.                                  }
149.   if (key_code[i][j]=='F') row_counter=0x00; //Если нажата 'F' очищаем row
150.   if ((key_code[i][j]=='E')&&
151.       (row_counter>0x00)) row_counter--;   //Если нажата 'E'
152.                                           //и есть введенные цифры – удалить последнюю
153.   if (key_code[i][j]=='C') {    //Если нажата 'C' тогда сверяем number и код доступа замка
154.                               get_number();
155.                               if (number==CODE) {
156.                                                  //отпираем замок
157.                                                 }
158.                                }
159.                                                    }
160.  write_row();           //Пишем row на ЖКИ
161. }
162. TCNT0=0x00;             //Очищаем таймер и флаг переполнения
163. TIFR=0x00;
164.
165. return;
166. }
167.
168. int main(void)
169. {
170. DDRD=0xfc;                     //Инициализация PortD
171. PORTD=0x00;
172.
173. pause(1000);                   //Задержка для включения ЖКИ
174. lcd_init();                    //Инициализация ЖКИ
175. init_timer();                  //Инициализация нулевого таймера
176. lcd_dat('D');                  //Напишем на ЖКИ "Data= "
177. lcd_dat('a');
178. lcd_dat('t');
179. lcd_dat('a');
180. lcd_dat('=');
181. lcd_dat(' ');
182.
183. sei();                        //Разрешения прерываний
184.
185. while(1)                      //Вечный цикл
186.  ;
187.
188. return 1;
189. }
```

Как видно, от последнего макета прошивка отличается строками 152-168. Также был добавлен #define в строку 12.

ИМХО: изменения кода доступа с клавиатуры как-то неправильно. Код доступа должен зашиваться, или выставляться какими-то джамперами, миниперекючателями, чтобы у злоумышленника не было возможности открыть замок, кроме как ввести правильный код.

»

Перемычки?

Опубликовано Anonymous в Вс, 10/17/2010 - 16:58.

А не проще ли по очереди сравнивать введенные символы со статическими значениями, вшитыми в EEPROM? ИМХО так проще... Да и наклепать функцию считывания/записи данных в EEPROM проще, чем разводить лишние дороги под джампера. Аффтору человеческое спасибо (на мысль натолкнул)...

»

Конечно проще :) я к этому и

Опубликовано vakula в Вс, 10/17/2010 - 22:31.

Конечно проще :) я к этому и клонил в предыдущем посте. Программные реализации в большинстве случаев получаются проще и дешевле аппаратных.

»

Кто бы еще подсказал....
Опубликовано Anonymous в Суб, 12/25/2010 - 03:53.

Кто бы еще подсказал, как в кодовом замке в строке 156 кратковременно выдать логическую единицу на PD0. Что-то не соображу никак.

»

с зависаниями разобрался -
Опубликовано Anonymous в Вс, 12/26/2010 - 14:50.

с зависаниями разобрался - проблема в микроконтроллере - у него один вывод походу накрылся (PB0) - если на него подать сигнал - проц виснет. Заменил МК - все работает.

»

DDRD|=0x01; PORTD|=0x01; paus
Опубликовано vakula в Суб, 12/25/2010 - 13:57.

```
1. DDRD|=0x01;
2.
3. PORTD|=0x01;
4. pause(100);
5. PORTD&=~0x01;
```

Так?

»

Сделал немного по-другому -
Опубликовано Anonymous в Вс, 12/26/2010 - 14:29.

Сделал немного по-другому - но смысл тот же. Единственное пришлось перейти на порт C (там 2 выхода свободные) - на D не получилось - подвисал ЖКИ

1. DDRC=0xff;
2. PORTC=0b00010000;
3. pause(1000);
4. PORTC=0b00000000;

Аналогично на другой выход подается сигнал при неправильном коде.

1. DDRC=0xff;
2. PORTC=0b00100000;
3. pause(10000);
4. PORTC=0b00000000;

Теперь иногда другая проблема возникает - при вводе кода вся система "подвисает". Помогает только сброс. Чую, надо копать в сторону watchgod...

»

Странно, подвисать ничего не
Опубликовано vakula в Вс, 12/26/2010 - 21:18.

Странно, подвисать ничего не должно. У тебя оптимизация включена? Используеш кварц как источник тактирования? На каком моменте подвисает?

»