# МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА» (БГТУ им. В.Г.Шухова)

Лабораторная работа №1 дисциплина «ЭВМ и переферийные устройства» по теме «Изучение принципов программного управления внешними устройствами на примере вывода информации на цифровой индикатор»

Выполнил: студент группы BT-31 Макаров Д.С. Проверил: Шамраев А.А.

# Лабораторная работа №1

«Изучение принципов программного управления внешними устройствами на примере вывода информации на цифровой индикатор»

**Цель работы:** изучить принципы функционирования и возможности программного управления цифровым индикатором, разработать алгоритм и программу для вывода информации на цифровой индикатор.

#### Вариант 6

**Задание:** Разработать в среде программирования Code Composer программу для микроконтроллера MSP430, которая выполняет вывод информации (фамилию, имя и отчество студента, и группу) на экран цифрового индикатора.

#### Порядок выполнения задания:

- запустить Code Composer IDE.
- создать пустой проект.
- создать файл ресурса для кода программы и подключить его к проекту.
- выполнить компиляцию исходного модуля программы и устранить ошибки, полученные на данном этапе.
- проверить работоспособность программы и показать результаты работы преподавателю.

## Ход работы

Схема стенда

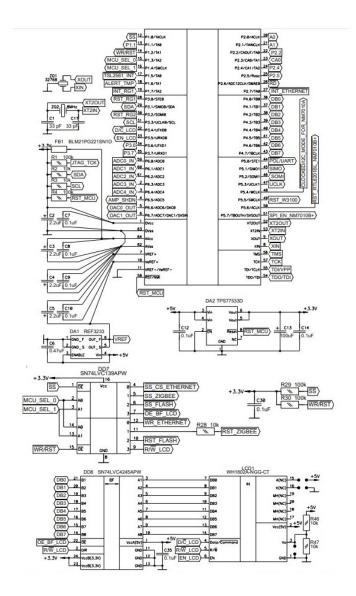


Рис. 1: Схема устройства

## Демонстрация работы программы

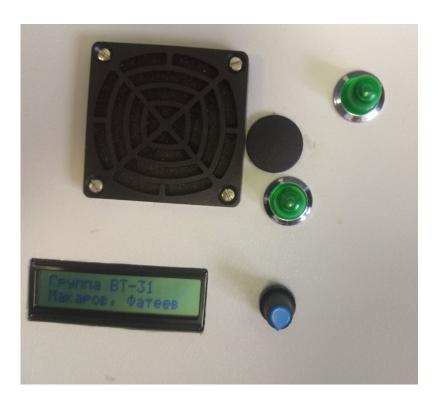


Рис. 2: Задание 1



Рис. 3: Задание 2\3

## Приложение

#### Содержимое файла lcd.c

```
#include "function_prototype.h"
#include "sysfunc.h"
#include "lcd.h"
char LCD_table[64]={
                      0x41,0xA0,0x42,0xA1,
                                              //0xC0...0xC3 \iff A \ B \ B \ \Gamma
                      0xE0,0x45,0xA3,0x33,
                                              //OxC4...OxC7 <=> Д Е Ж З
                                              //0xC8...0xCB <=> И Й К Л
                      0xA5,0xA6,0x4B,0xA7,
                                               //0xCC...0xCF \iff M H O \Pi
                      0x4D,0x48,0x4F,0xA8,
                      0x50,0x43,0x54,0xA9,
                                              //0xD0...0xD4 \iff P C T Y
                                               //OxD5...OxD7 <=> Ф X Ц Ч
                      0xAA,0x58,0xE1,0xAB,
                      OxAC, OxE2, OxAC, OxAE,
                                               //OxD8...ОxDB <=> Ш Щ Ъ Ы
                      0x62,0xAF,0xB0,0xB1,
                                               //OxDC...OxDF <=> Ь Э Ю Я
                      0x61,0xB2,0xB3,0xB4,
                                               //0xE0...0xE4 <=> а б в г
                      0xE3,0x65,0xB6,0xB7,
                                               //0xE5...0xE7 <=> дежз
                      0xB8,0xA6,0xBA,0xBB,
                                               //0xE8...0xEB <=> u ŭ κ л
                                               //0xEC...0xEF <=> м н о п
                      0xBC,0xBD,0x6F,0xBE,
                                               //0xF0...0xE4 \iff p c m y
                      0x70,0x63,0xBF,0x79,
                      0xE4,0xD5,0xE5,0xC0,
                                               //0xF5...0xE7 <=> $ x \ u \ \upsilon$
                                               //OxF8...OxEB <=> ш щ з ы
                      0xC1,0xE6,0xC2,0xC3,
                      0XC4,0xC5,0xC6,0xC7
                                               //0xFC...0xEF <=> ь э ю я
};
byte LCD_row, LCD_col, n;
void LCD_init()
  wait_1ms(20);
 P3DIR |= (D_nC_LCD + EN_LCD);
  Reset_EN_LCD();
  LCD_WriteCommand(0x3C);
   wait_1ms(1);
 LCD_WriteCommand(0x3C);
  wait_1ms(1);
 LCD_WriteCommand(0x0C);
 LCD_clear();
  LCD_WriteCommand(0x06);
void LCD_message(const char * buf){
 n = 0;
  while (buf[n]){
    if ( (LCD_row < LCD_MAXROWS-1) && (LCD_col >= LCD_MAXCOLS) )
     LCD_set_pos(++LCD_row, 0);
    if (LCD_col >= LCD_MAXCOLS )
     LCD_set_pos(0,0);
    LCD_WriteData( LCD_recode(buf[n]) );
    LCD_col++;
    n++;
```

```
}
void LCD_clear(){
 LCD_WriteCommand(0x01);
 LCD_row=0;
 LCD_col=0;
}
void LCD_set_pos(byte row, byte col){
  if (row > LCD_MAXROWS-1)
   row = LCD_MAXROWS-1;
 if (col > LCD_MAXCOLS-1)
   col = LCD_MAXCOLS-1;
 LCD_row = row;
 LCD_col = col;
 LCD_WriteCommand( BIT7 | ((0x40 * LCD_row) + LCD_col) );
}
byte LCD_get_row(){
  return LCD_row;
}
byte LCD_get_col(){
 return LCD_col;
void LCD_set_cursor(byte cursor){
  if (cursor > 3)
    cursor = 2;
 LCD_WriteCommand(cursor | BIT2 | BIT3);
}
void LCD_WriteCommand(char byte){
   LCD_WriteByte(byte, 0);
}
void LCD_WriteData(char byte){
    LCD_WriteByte(byte, 1);
}
void LCD_WriteByte(char byte, char D_nC){
  //шина данных на вход
 DB_DIR = 0x00;
  //установка входов дешифратора
  Set_MCU_SEL_0();
  Set_MCU_SEL_1();
 Reset_D_nC_LCD();
 Set_nWR_nRST();
 Reset_nSS();
  //строб
  Set_EN_LCD();
  Set_EN_LCD();
  Set_EN_LCD();
  //ожидание busy flag
  while (DB_IN & BIT7);
```

```
Reset_EN_LCD();
  Set_nSS();
  if (D_nC) Set_D_nC_LCD();
  else Reset_D_nC_LCD();
  Reset_nWR_nRST();
 Reset_nSS();
  //шина данных на выход
 DB_DIR = OxFF;
 DB_OUT = byte;
  //cmpo6
 Set_EN_LCD();
  Set_EN_LCD();
  Set_EN_LCD();
 Reset_EN_LCD();
 Set_nSS();
 DB_DIR = 0x00;
 Set_nWR_nRST();
}
char LCD_recode(char b){
  if (b<192) return b;
    else return LCD_table[b-192];
}
```

#### Содержимое файла lcd.h

```
#ifndef __LCD_H__
#define __LCD_H__

#define LCD_MAXCOLS 16
#define LCD_MAXROWS 2
#endif
```

## Содержимое файла main.c

```
#include <msp430.h>
#include "stdio.h"
#include "system_define.h"
#include "system_variable.h"
#include "function_prototype.h"
#include "main.h"
/*
 * main.c
void main(void) {
    //Выключение сторожевого таймера
    WDTCTL = WDTPW + WDTHOLD;
    //Инициализация таймеров тактирования
    Init_System_Clock();
    Init_System();
    char i = 0;
    char send_str0[] = "Γρуππа BT-31";
    char send_str1[] = "Макаров, Фатеев";
    for(i=0;i<sizeof(send_str0);i++){</pre>
        send_str0[i] = LCD_recode(send_str0[i]);
    };
    for(i=0;i<sizeof(send_str1);i++){</pre>
```

```
send_str1[i] = LCD_recode(send_str1[i]);
   };
   LCD_init();
   LCD_set_pos(0,0);
   LCD_message(send_str0);
   LCD_set_pos(1,0);
   LCD_message(send_str1);
    /*LCD_init();
    byte character[8] =
\rightarrow \quad \{0b00011111,0b00011011,0b00011011,0b00011011,0b00010111,0b00010111,0b00011101,0b00010111,0b00011101\};
    char i;
    for (i = 0; i < 8; i++) {
       LCD_WriteCommand(0x40 + i);
       LCD_WriteData(character[i]);
   LCD_set_pos(0,0);
   LCD_WriteData(0);
   byte character1[8] =
for (i = 0; i < 8; i++) {
       LCD_WriteCommand(0x48 + i);
       LCD_WriteData(character1[i]);
    7
   LCD_set_cursor(2);
   LCD_set_pos(1,0);
   LCD_WriteData(1);*/
   while (1)\{\};
}
    Содержимое файла sysfunc.c
#include <msp430.h>
#include "sysfunc.h"
void Init_System(){
       P1DIR |= (nSS + nWR_nRST + MCU_SEL_0 + MCU_SEL_1);
       DB_DIR = 0x00;
}
void Init_System_Clock(){
       volatile byte i;
       BCSCTL1 &= ~XT20FF;
       do{
               IFG1 &= ~OFIFG;
               for (i = 0xFF; i > 0; i--);
       }
       while ((IFG1 & OFIFG));
       BCSCTL2 |= SELM_2 | SELS;
}
void wait_1ms(word cnt){
 for (wait_i = 0; wait_i < cnt; wait_i++)</pre>
   for (wait_j = 0; wait_j < 1000; wait_j++);
}
void wait_1mks(word cnt){
  for (wait_i = 0; wait_i < cnt; wait_i++);</pre>
}
```

## Содержимое файла sysfunc.h

```
#ifndef __SYSFUNC_H__
#define __SYSFUNC_H__
#include "system_define.h"

extern word wait_i, wait_j;
#endif
```