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

Лабораторная работа № 3  
Дисциплина: ЭВМ и периферийные устройства  
Тема: «Изучение принципов организации обмена данными по последовательному интерфейсу I2C на примере управления блоком светодиодов и программного опроса клавиатуры»

Выполнил: ст. группы ВТ-31

Проверил: Шамраев А. А.

Белгород 2020

# **Цель работы:** изучить принципы программного управления двунаправленным обменом данных по последовательному интерфейсу I2C.

# **Задание:** Разработать в среде программирования IAR Embedded Workbench программу на языке С, которая выполняет опрос клавиатуры лабораторного стенда и выводит информацию о нажатых клавишах с помощью блока светодиодов. Варианты индивидуальных заданий представлены в таблице 4.3.8.

# **Порядок выполнения работы**

– включить лабораторный макет.

– запустить компилятор IAR Embedded Workbench.

– создать пустой проект.

– создать файл ресурса для кода программы и подключить его к проекту.

– ввести код исходного модуля программы обмена данными между микроконтроллером MSP430F1611 с регистрами PCA9538 по интерфейсу I2C

соответствие с индивидуальным заданием, приведенным в таблице 4.3.8.

– выполнить компиляцию исходного модуля программы и устранить

ошибки, полученные на данном этапе.

– настроить параметры программатора.

– создать загрузочный модуль программы и выполнить программирование микроконтроллера.

Проверить работоспособность загруженной в микроконтроллер программы и показать результаты работы преподавателю.

В случае некорректной работы разработанной программы, выполнить

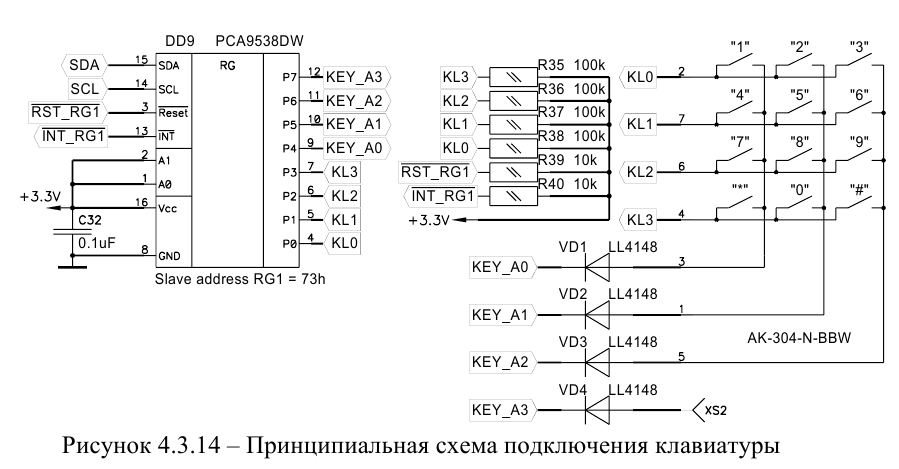
аппаратный сброс микроконтроллера, провести отладку исходного модуля программы и заново проверить функционирование программы.

**Вариант 2**

**Задание по варианту:**

Разработать программу, фиксирующую нажатия клавиш 5, 7 и 9 матричной клавиатуры включением светодиодов 5, 6 и 7 соответственно. Выход из цикла опроса осуществляется при нажатии клавиши \*. Частота тактовых импульсов на линии SCL – 20 кГц.

**Схема подключения используемых устройств:**





**Текст программы:**

**main.c:**

#include <msp430.h>

#include "system\_define.h"

#include "system\_variable.h"

#include "function\_prototype.h"

#include "I2C.h"

#include "main.h"

/\*

\* main.c

\*/

void main(void) {

\_enable\_interrupt();

WDTCTL = WDTPW+WDTHOLD; // Отключить сторожевой таймер

Init\_System\_Clock(); // Запустить тактирование

Init\_System(); // Настройка портов, итд

Init\_I2C(); // Настройка I2C

LED\_clear(); // Очистка дисплея

char key;

int f = 0;

// Цикл продолжается, пока не введется символ \*

while (f != 1) {

key = KEYS\_scannow();

switch (key) {

case '5':

LED\_clear();

LED\_set(5);

break;

case '7':

LED\_clear();

LED\_set(6);

break;

case '9':

LED\_clear();

LED\_set(7);

break;

case '\*':

f = 1;

}

}

// \* - выход из цикла опроса

while(1){

}

}

**I2C.c**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// I2C function

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "function\_prototype.h"

#include "system\_define.h"

#include "I2C.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//============================================================================

// Инициализация модуля UART0 для работы в режиме I2C

void Init\_I2C()

{

P3SEL |= 0x0A; // Выбор альтернативной функции для линий порта P3

// в режиме I2C SDA->P3.1, SCL->P3.3

U0CTL |= I2C + SYNC; // Выбрать режим I2C для USART0

U0CTL &= ~I2CEN; // Выключить модуль I2C

// Конфигурация модуля I2C

I2CTCTL=I2CSSEL\_1; // SMCLK = 32768 Гц

// Необходимо получить частоту 20 кГц. Выставим внутреннюю частоту 32768/256 = 128 гц

I2CPSC = 0xFF; // Делитель частоты

I2CSCLH = 0x9B; // Формула для вычисления частоты High Period of SCL: (I2CSCLH + 2) \* I2CPSC = 157 \* 128 = 20096 ~ 20 кГц

I2CSCLL = 0x9B; // Low period of SCL = High Period of SCL (выставляем одинаковые периоды высокого и низкого уровня)

U0CTL |= I2CEN; // Включить модуль I2C

// формирование строба сброса I2C-регистров PCA9538 - RST\_RG1->P3.1 и RST\_RG2->P3.2

P3DIR |= 0x05; // переключаем эти ножки порта на вывод,

P3SEL &= ~0x05; // выбираем функцию ввода-вывода для них

P3OUT &= ~0x05; // и формируем строб сброса на 1 мс

wait\_1ms(1);

P3OUT |= 0x05;

}

//============================================================================

//============================================================================

// отправка данных по протоколу I2C

void Send\_I2C(unsigned char\* buffer,unsigned int num, unsigned char address)

{

while (I2CBUSY & I2CDCTL); // проверка готовности модуля I2C

BufTptr=buffer;

I2CSA = address; // установка адреса приемнмка

I2CNDAT =num; // количество передаваемых байт

I2CIE = TXRDYIE+ALIE; // разрешение прерываний по окончанию передачи байта и по потере арбитража

U0CTL |= MST; // режим Master

I2CTCTL |= I2CSTT + I2CSTP + I2CTRX; // инициализировать передачу

while ((I2CTCTL & I2CSTP) == 0x02); // ожидание условия СТОП

}

//============================================================================

//============================================================================

// прием данных по протоколу I2C

void Receive\_I2C(unsigned char\* buffer,unsigned int num, unsigned char address)

{

while (I2CBUSY & I2CDCTL); // проверка готовности модуля I2C

BufRptr=buffer;

I2CSA=address;

I2CTCTL&=~I2CTRX; // режим приема

I2CNDAT=num;

I2CIE=RXRDYIE; // резрешение прерывания по окончанию приема байта

U0CTL |= MST;

I2CTCTL |= I2CSTT + I2CSTP; // инициализировать прием

while ((I2CTCTL & I2CSTP) == 0x02); // ожидание условия СТОП

}

//============================================================================

//============================================================================

// отправка байта устройству на шине I2C

void I2C\_SendByte(char data, char i2c\_addr)

{

Tx\_Data[0] = data; // отправляемый байт

Send\_I2C(&Tx\_Data[0], 1, i2c\_addr); // вывод по I2C на устройство

}

//============================================================================

//============================================================================

// запись байта в регистр устройства на шине I2C

void I2C\_WriteByte(char reg, char data, char i2c\_addr)

{

Tx\_Data[0] = reg; // выбираем регистр

Tx\_Data[1] = data; // записываемые данные

Send\_I2C(&Tx\_Data[0], 2, i2c\_addr); // вывод по I2C на устройство

}

//============================================================================

//============================================================================

// чтение байта из регистра устройства на шине I2C

byte I2C\_ReadByte(char reg, char i2c\_addr)

{

Tx\_Data[0] = reg; // выбираем регистр

Send\_I2C(&Tx\_Data[0], 1, i2c\_addr);

Receive\_I2C(&Rx\_Data[0], 1, i2c\_addr); // получаем значение из регистра

return Rx\_Data[0];

}

//============================================================================

//============================================================================

// чтение слова (2 байта) из регистра устройства на шине I2C

int I2C\_ReadWord(char reg, char i2c\_addr)

{

Tx\_Data[0] = reg; // выбираем регистр

Send\_I2C(&Tx\_Data[0], 1, i2c\_addr);

Receive\_I2C(&Rx\_Data[0], 2, i2c\_addr); // получаем 2 байта значение из регистра

return Rx\_Data[0] + (Rx\_Data[1] \* 256);

}

//============================================================================

//============================================================================

//Обработка прерывания от модуля USART0, работающего в режиме I2C

// вектор прерываний для модуля I2C

#pragma vector=USART0TX\_VECTOR

\_\_interrupt void I2C\_ISR()

{

switch(I2CIV)

{

case 0: break; // нет прерывания

case 2: break; // потеря арбитража

case 4: break; // нет подтверждения

case 6: break; // прерывание собственного адреса

case 8: break; // регистр доступен для чтения

case 10: // окончание приема байта

\*BufRptr++=I2CDRB;

break;

case 12: // окончание передачи байта

I2CDRB=\*BufTptr++;

break;

case 14: break; // общий вызов

case 16: break; // обнаружено условие СТАРТ

default : break;

}

}//============================================================================

**Полученные результаты:** при нажатии клавиш 5, 7 и 9 на лабораторном стенде включаются светодиоды 5, 6 и 7 соответственно. Выход из цикла опроса осуществляется при нажатии клавиши \*.

**Выводы:** изучил принципы программного управления двунаправленным обменом данных по последовательному интерфейсу I2C.