```
================================================================================
main.c
--------------------------------------------------------------------------------------------------------------
#include "main.h"

int main(void)
{
	unsigned int i;
	I2C_Init();//Инициализируем TWI
	LCD_ini();//инициализируем символьный дисплей
	clearlcd();//очистим символьный дисплей
	setpos(0,0);
	str_lcd("TEST TFT ILI9341");
	TFT9341_ini();
	TFT9341_FillScreen(RED);
	_delay_ms(500);
	TFT9341_FillScreen(BLUE);
	_delay_ms(500);
	TFT9341_FillScreen(GREEN);
	_delay_ms(500);
	TFT9341_FillScreen(CYAN);
	_delay_ms(500);
	TFT9341_FillScreen(MAGENTA);
	_delay_ms(500);
	TFT9341_FillScreen(YELLOW);
	_delay_ms(500);
	TFT9341_FillScreen(WHITE);
	_delay_ms(500);
	TFT9341_FillScreen(BLACK);
	_delay_ms(500);
	for(i=0;i<8;i++)
	{
		TFT9341_FillRectangle(TFT9341_RandColor(),0,0,119,159);
		_delay_ms(100);
		TFT9341_FillRectangle(TFT9341_RandColor(),120,0,239,159);
```

```
        _delay_ms(100);
        TFT9341_FillRectangle(TFT9341_RandColor(),0,160,119,319);
        _delay_ms(100);
        TFT9341_FillRectangle(TFT9341_RandColor(),120,160,239,319);
        _delay_ms(100);
}
TFT9341_FillScreen(BLACK);
for(i=0;i<15000;i++)
{
        TFT9341_DrawPixel(rand()%240,rand()%320,TFT9341_RandColor());
}
TFT9341_FillScreen(BLACK);
for(i=0;i<240;i++)
{
        TFT9341_DrawLine(TFT9341_RandColor(),i,0,i,319);
}
_delay_ms(500);
TFT9341_FillScreen(BLACK);
for(i=0;i<1000;i++)
{
        TFT9341_DrawLine(TFT9341_RandColor(),rand()%240,rand()%320,rand()%240,rand()%320);
}
_delay_ms(500);
TFT9341_FillScreen(BLACK);
for(i=0;i<120;i++)
{
        TFT9341_DrawRect(TFT9341_RandColor(),i,i,239-i,319-i);
}
_delay_ms(500);
TFT9341_FillScreen(BLACK);
for(i=0;i<2000;i++)
{
        TFT9341_DrawCircle(rand()%200+20, rand()%280+20, 20, TFT9341_RandColor());
}
_delay_ms(500);
```

```
TFT9341_FillScreen(BLACK);
TFT9341_Draw_Char(10,10,RED,GREEN,0x21,2);
TFT9341_Draw_Char(26,10,RED,GREEN,0x22,2);
TFT9341_Draw_Char(42,10,RED,GREEN,0x23,2);
TFT9341_Draw_Char(58,10,RED,GREEN,0x24,2);
TFT9341_Draw_Char(74,10,RED,GREEN,0x25,2);
TFT9341_Draw_Char(10,26,RED,GREEN,0x21,2);
TFT9341_Draw_Char(26,26,RED,GREEN,0x22,2);
TFT9341_Draw_Char(42,26,RED,GREEN,0x23,2);
TFT9341_Draw_Char(58,26,RED,GREEN,0x24,2);
TFT9341_Draw_Char(74,26,RED,GREEN,0x25,2);
_delay_ms(1000);
TFT9341_FillScreen(BLACK);
TFT9341_String(1,1,RED,GREEN,"12345ABCDE",1);
TFT9341_String(1,9,RED,GREEN,"EDCAB54321",1);
TFT9341_String(10,17,RED,GREEN,"ABCDEabcde",1);
_delay_ms(1000);
TFT9341_FillScreen(BLACK);

        while (1)
        {
                TFT9341_Draw_Char((rand()%15)*16,(rand()%20)*16,GREEN,BLACK,rand()%2+0x21,2);

        }
}
```

==========

main.h

```
#ifndef MAIN_H_
#define MAIN_H_

#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>
```

```
#include <util/delay.h>
#include <stdio.h>
#include <stdlib.h>
#include <avr/pgmspace.h>

#include "twi.h"
#include "lcdtwi.h"
#include "ili9341.h"

#endif /* MAIN_H_ */
```

==========

lcdtwi.c

```
#include "lcdtwi.h"
//-----------------------------------------
unsigned char portlcd = 0; //ячейка для хранения данных порта микросхемы расширения
//-----------------------------------------
void sendhalfbyte(unsigned char c)
{
	c<<=4;
	e1; //включаем линию E
	_delay_us(50);
	I2C_SendByteByADDR(portlcd|c,0b01001110);
	e0; //выключаем линию E
	_delay_us(50);
}
//-------------------------------------
void sendbyte(unsigned char c, unsigned char mode)
{
	if (mode==0) rs0;
	else		rs1;
	unsigned char hc=0;
	hc=c>>4;
	sendhalfbyte(hc); sendhalfbyte(c);
}
```

```
//---------------------------------------
void sendcharlcd(unsigned char c)
{
	sendbyte(c,1);
}
//---------------------------------------
void setpos(unsigned char x, unsigned y)
{
	switch(y)
	{
		case 0:
			sendbyte(x|0x80,0);
			break;
		case 1:
			sendbyte((0x40+x)|0x80,0);
			break;
		case 2:
			sendbyte((0x14+x)|0x80,0);
			break;
		case 3:
			sendbyte((0x54+x)|0x80,0);
			break;
	}
}
//---------------------------------------
void LCD_ini(void)
{
	_delay_ms(15); //Ждем 15 мс (стр 45)
	sendhalfbyte(0b00000011);
	_delay_ms(4);
	sendhalfbyte(0b00000011);
	_delay_us(100);
	sendhalfbyte(0b00000011);
	_delay_ms(1);
	sendhalfbyte(0b00000010);
```

```
        _delay_ms(1);
        sendbyte(0b00101000, 0); //4бит-режим (DL=0) и 2 линии (N=1)
        _delay_ms(1);
        sendbyte(0b00001100, 0); //включаем изображение на дисплее (D=1), курсоры никакие не включаем (C=0, B=0)
        _delay_ms(1);
        sendbyte(0b00000110, 0); //курсор (хоть он у нас и невидимый) будет двигаться влево
        _delay_ms(1);
        setled();//подсветка
        setwrite();//запись
}
//---------------------------------------
void clearlcd(void)
{
        sendbyte(0b00000001, 0);
        _delay_us(1500);
}
//---------------------------------------
void str_lcd (char str1[])
{
        wchar_t n;
        for(n=0;str1[n]!='\0';n++)
        sendcharlcd(str1[n]);
}
//---------------------------------------
```

==========

lcdtwi.h

```
#ifndef LCDTWI_H_
#define LCDTWI_H_

#include "main.h"

//---------------------------------------
void LCD_ini(void);
void setpos(unsigned char x, unsigned y);
```

```
void str_lcd (char str1[]);
void clearlcd(void);
void sendcharlcd(unsigned char c);
//----------------------------------------
#define e1   I2C_SendByteByADDR(portlcd|=0x04,0b01001110) // установка линии E в 1
#define e0   I2C_SendByteByADDR(portlcd&=~0x04,0b01001110)  // установка линии E в 0
#define rs1   I2C_SendByteByADDR(portlcd|=0x01,0b01001110) // установка линии RS в 1
#define rs0   I2C_SendByteByADDR(portlcd&=~0x01,0b01001110)  // установка линии RS в 0
#define setled()     I2C_SendByteByADDR(portlcd|=0x08,0b01001110) // включение подсветки
#define setwrite()   I2C_SendByteByADDR(portlcd&=~0x02,0b01001110)  // установка записи в память дисплея
//----------------------------------------


#endif /* LCDTWI_H_ */
```

==========

twi.c

```
#include "twi.h"

void I2C_Init (void)
{
        TWBR=0x48;//скорость передачи (при 16 мгц получается 100 кгц)
}


void I2C_StartCondition(void)
{
        TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
        while (!(TWCR & (1<<TWINT)));//подождем пока установится TWIN
}

void I2C_StopCondition(void)
{
        TWCR = (1<<TWINT)|(1<<TWSTO)|(1<<TWEN);
}
```

```
void I2C_SendByte(unsigned char c)
{
	TWDR = c;//запишем байт в регистр данных
	TWCR = (1<<TWINT)|(1<<TWEN);//включим передачу байта
	while (!(TWCR & (1<<TWINT)));//подождем пока установится TWIN
}

void I2C_SendByteByADDR(unsigned char c,unsigned char addr)
{
	I2C_StartCondition(); // Отправим условие START
	I2C_SendByte(addr); // Отправим в шину адрес устройства + бит чтения-записи
	I2C_SendByte(c);// Отправим байт данных
	I2C_StopCondition();// Отправим условие STOP
}

unsigned char I2C_ReadByte(void)
{
	TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWEA);
	while (!(TWCR & (1<<TWINT)));//ожидание установки бита TWIN
	return TWDR;//читаем регистр данных
}

unsigned char I2C_ReadLastByte(void)
{
	TWCR = (1<<TWINT)|(1<<TWEN);
	while (!(TWCR & (1<<TWINT)));//ожидание установки бита TWIN
	return TWDR;//читаем регистр данных
}
```

==========

twi.h

```
#ifndef TWI_H_
#define TWI_H_
```

```
#include "main.h"

void I2C_Init (void); //инициализация i2c
void I2C_StartCondition(void); //Отправим условие START
void I2C_StopCondition(void); //Отправим условие STOP
void I2C_SendByte(unsigned char c); //передача байта в шину
void I2C_SendByteByADDR(unsigned char c,unsigned char addr); //передача байта в шину на устройство по адресу
unsigned char I2C_ReadByte(void); //читаем байт
unsigned char I2C_ReadLastByte(void); //читаем последний байт


#endif /* TWI_H_ */
```

==========

ili9341.c

```
#include "ili9341.h"
//-------------------------------------------------------------
unsigned int X_SIZE = 0;
unsigned int Y_SIZE = 0;
unsigned long dtt=0;
//-------------------------------------------------------------
//font 16
const unsigned char chars16[][32] PROGMEM =
{
	//SPACE
	{0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
	0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00},
	//0
	{0x00, 0x00, 0x00, 0x00, 0x07, 0xc0, 0x0f, 0xe0, 0x0c, 0x60, 0x18, 0x30, 0x18, 0x30, 0x18, 0x30,
	0x18, 0x30, 0x18, 0x30, 0x18, 0x30, 0x0c, 0x60, 0x0f, 0xe0, 0x07, 0xc0, 0x00, 0x00, 0x00, 0x00},
	//1
	{0x00, 0x00, 0x00, 0x00, 0x03, 0x80, 0x03, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80,
	0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x01, 0x80, 0x03, 0xc0, 0x03, 0xc0, 0x00, 0x00, 0x00, 0x00},
	//A
	{0x00, 0x00, 0x00, 0x00, 0x07, 0xf0, 0x07, 0xf0, 0x01, 0x40, 0x03, 0x60, 0x03, 0x60, 0x06, 0x30,
```

```
        0x07, 0xf0, 0x0f, 0xf8, 0x0c, 0x18, 0x0c, 0x18, 0x3e, 0x3e, 0x3e, 0x3e, 0x00, 0x00, 0x00, 0x00},
        //B
        {0x00, 0x00, 0x00, 0x00, 0x0f, 0xe0, 0x0f, 0xf0, 0x06, 0x30, 0x06, 0x30, 0x07, 0xe0, 0x07, 0xf0,
        0x06, 0x38, 0x06, 0x18, 0x06, 0x18, 0x06, 0x38, 0x0f, 0xf0, 0x0f, 0xe0, 0x00, 0x00, 0x00, 0x00},
        //C
        {0x00, 0x00, 0x00, 0x00, 0x07, 0xd8, 0x0f, 0xf8, 0x1c, 0x38, 0x38, 0x18, 0x30, 0x00, 0x30, 0x00,
        0x30, 0x00, 0x30, 0x08, 0x38, 0x0c, 0x1c, 0x38, 0x0f, 0xf0, 0x07, 0xe0, 0x00, 0x00, 0x00, 0x00}
};
//-------------------------------------------------------------
//font 8
const unsigned char chars8[][8] PROGMEM ={
        //SPACE
        {0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00},
        // !
        {0x04,0x04,0x04,0x04,0x00,0x04,0x00,0x00},
        // "
        {0x0A,0x0A,0x0A,0x00,0x00,0x00,0x00,0x00},
        // #
        {0x0A,0x0A,0x1F,0x0A,0x1F,0x0A,0x0A,0x00},
        // $
        {0x04,0x0F,0x14,0x0E,0x05,0x1E,0x04,0x00},
        // %
        {0x18,0x19,0x02,0x04,0x08,0x13,0x03,0x00},
        // &
        {0x0C,0x12,0x14,0x08,0x14,0x12,0x0D,0x00},
        // '
        {0x0C,0x04,0x08,0x00,0x00,0x00,0x00,0x00},
        // (
        {0x02,0x04,0x08,0x08,0x08,0x04,0x02,0x00},
        // )
        {0x08,0x04,0x02,0x02,0x02,0x04,0x08,0x00},
        // *
        {0x00,0x04,0x15,0x0E,0x15,0x04,0x00,0x00},
        // +
        {0x00,0x04,0x04,0x1F,0x04,0x04,0x00,0x00},
```

```
// ,
{0x00,0x00,0x00,0x00,0x0C,0x04,0x08,0x00},
// -
{0x00,0x00,0x00,0x1F,0x00,0x00,0x00,0x00},
// .
{0x00,0x00,0x00,0x00,0x00,0x0C,0x0C,0x00},
// /
{0x00,0x01,0x02,0x04,0x08,0x10,0x00,0x00},
// 0
{0x0E,0x11,0x13,0x15,0x19,0x11,0x0E,0x00},
// 1
{0x04,0x0C,0x04,0x04,0x04,0x04,0x0E,0x00},
// 2
{0x0E,0x11,0x01,0x02,0x04,0x08,0x1F,0x00},
// 3
{0x1F,0x02,0x04,0x02,0x01,0x11,0x0E,0x00},
// 4
{0x02,0x06,0x0A,0x12,0x1F,0x02,0x02,0x00},
// 5
{0x1F,0x10,0x1E,0x01,0x01,0x11,0x0E,0x00},
// 6
{0x06,0x08,0x10,0x1E,0x11,0x11,0x0E,0x00},
// 7
{0x1F,0x01,0x02,0x04,0x08,0x08,0x08,0x00},
// 8
{0x0E,0x11,0x11,0x0E,0x11,0x11,0x0E,0x00},
// 9
{0x0E,0x11,0x11,0x0F,0x01,0x02,0x0C,0x00},
// :
{0x00,0x01,0x02,0x04,0x08,0x10,0x00,0x00},
// ;
{0x00,0x0C,0x0C,0x00,0x0C,0x04,0x08,0x00},
// <
{0x02,0x04,0x08,0x10,0x08,0x04,0x02,0x00},
// =
```

```
{0x00,0x00,0x1F,0x00,0x1F,0x00,0x00,0x00},
// >
{0x08,0x04,0x02,0x01,0x02,0x04,0x08,0x00},
// ?
{0x0E,0x11,0x01,0x02,0x04,0x00,0x04,0x00},
// @
{0x0E,0x11,0x01,0x0D,0x15,0x15,0x0E,0x00},
// A
{0x0E,0x11,0x11,0x11,0x1F,0x11,0x11,0x00},
// B
{0x1E,0x11,0x11,0x1E,0x11,0x11,0x1E,0x00},
// C
{0x0E,0x11,0x10,0x10,0x10,0x11,0x0E,0x00},
// D
{0x1C,0x12,0x11,0x11,0x11,0x12,0x1C,0x00},
// E
{0x1F,0x10,0x10,0x1E,0x10,0x10,0x1F,0x00},
// F
{0x1F,0x10,0x10,0x1E,0x10,0x10,0x10,0x00},
// G
{0x0E,0x11,0x10,0x17,0x11,0x11,0x0E,0x00},
// H
{0x11,0x11,0x11,0x1F,0x11,0x11,0x11,0x00},
// I
{0x0E,0x04,0x04,0x04,0x04,0x04,0x0E,0x00},
// J
{0x07,0x02,0x02,0x02,0x02,0x12,0x0C,0x00},
// K
{0x11,0x12,0x14,0x18,0x14,0x12,0x11,0x00},
// L
{0x10,0x10,0x10,0x10,0x10,0x10,0x1F,0x00},
// M
{0x11,0x1B,0x15,0x15,0x11,0x11,0x11,0x00},
// N
{0x11,0x11,0x19,0x15,0x13,0x11,0x11,0x00},
```

```
// O
{0x0E,0x11,0x11,0x11,0x11,0x11,0x0E,0x00},
// P
{0x1E,0x11,0x11,0x1E,0x10,0x10,0x10,0x00},
// Q
{0x0E,0x11,0x11,0x11,0x15,0x12,0x0D,0x00},
// R
{0x1E,0x11,0x11,0x1E,0x14,0x12,0x11,0x00},
// S
{0x0F,0x10,0x10,0x0E,0x01,0x01,0x1E,0x00},
// T
{0x1F,0x04,0x04,0x04,0x04,0x04,0x04,0x00},
// U
{0x11,0x11,0x11,0x11,0x11,0x11,0x0E,0x00},
// V
{0x11,0x11,0x11,0x11,0x11,0x0A,0x04,0x00},
// W
{0x11,0x11,0x11,0x11,0x15,0x15,0x0E,0x00},
// X
{0x11,0x11,0x0A,0x04,0x0A,0x11,0x11,0x00},
// Y
{0x11,0x11,0x11,0x0A,0x04,0x04,0x04,0x00},
// Z
{0x1F,0x01,0x02,0x04,0x08,0x10,0x1F,0x00},
// [
{0x0E,0x08,0x08,0x08,0x08,0x08,0x0E,0x00},
//
{0x11,0x0A,0x1F,0x04,0x1F,0x04,0x04,0x00},
// ]
{0x0E,0x02,0x02,0x02,0x02,0x02,0x0E,0x00},
// ^
{0x04,0x0A,0x11,0x00,0x00,0x00,0x00,0x00},
// _
{0x00,0x00,0x00,0x00,0x00,0x00,0x1F,0x00},
// '
```

```
{0x08,0x04,0x00,0x00,0x00,0x00,0x00,0x00},
// a
{0x00,0x00,0x0E,0x01,0x0F,0x11,0x0F,0x00},
// b
{0x10,0x10,0x1E,0x11,0x11,0x11,0x1E,0x00},
// c
{0x00,0x00,0x0E,0x10,0x10,0x11,0x0E,0x00},
// d
{0x01,0x01,0x0D,0x13,0x11,0x11,0x0F,0x00},
// e
{0x00,0x00,0x0E,0x11,0x1F,0x10,0x0E,0x00},
// f
{0x06,0x09,0x08,0x1C,0x08,0x08,0x08,0x00},
// g
{0x00,0x0F,0x11,0x11,0x0F,0x01,0x0E,0x00},
// h
{0x10,0x10,0x16,0x19,0x11,0x11,0x11,0x00},
// i
{0x04,0x00,0x0C,0x04,0x04,0x04,0x0E,0x00},
// j
{0x02,0x00,0x06,0x02,0x02,0x12,0x0C,0x00},
// k
{0x10,0x10,0x12,0x14,0x18,0x14,0x12,0x00},
// l
{0x18,0x08,0x08,0x08,0x08,0x08,0x1C,0x00},
// m
{0x00,0x00,0x1A,0x15,0x15,0x11,0x11,0x00},
// n
{0x00,0x00,0x16,0x19,0x11,0x11,0x11,0x00},
// o
{0x00,0x00,0x0E,0x11,0x11,0x11,0x0E,0x00},
// p
{0x00,0x00,0x1E,0x11,0x1E,0x10,0x10,0x00},
// q
{0x00,0x00,0x0F,0x11,0x0F,0x01,0x01,0x00},
```

```
// r
 {0x00,0x00,0x16,0x19,0x10,0x10,0x10,0x00},
// s
 {0x00,0x00,0x0E,0x10,0x0E,0x01,0x1E,0x00},
// t
 {0x08,0x08,0x1C,0x08,0x08,0x09,0x06,0x00},
// u
 {0x00,0x00,0x11,0x11,0x11,0x13,0x0D,0x00},
// v
 {0x00,0x00,0x11,0x11,0x11,0x0A,0x04,0x00},
// w
 {0x00,0x00,0x11,0x11,0x11,0x15,0x0A,0x00},
// x
 {0x00,0x00,0x11,0x0A,0x04,0x0A,0x11,0x00},
// y
 {0x00,0x00,0x11,0x11,0x0F,0x01,0x0E,0x00},
// z
 {0x00,0x00,0x1F,0x02,0x04,0x08,0x1F,0x00}
};
//-------------------------------------------------------------
void port_ini(void)
{
	DATA_PORT=0x00;
	DATA_DDR=0xFF;//Шина данных на выход
	COMMAND_DDR=0x1F;//Командные лапки также все на выход
}
//-------------------------------------------------------------
void TFT9341_SendCommand(unsigned char cmd)
{
	CD_COMMAND;//лапка в состоянии посылки команды
	RD_IDLE;//отключим чтение
	CS_ACTIVE;//выбор дисплея
	DATA_PORT=cmd;
	WR_STROBE;
	CS_IDLE;
```

```
}
//---------------------------------------------------------------
void TFT9341_SendData(unsigned char dt)
{
	CD_DATA;//лапка в состоянии посылки данных
	RD_IDLE;//отключим чтение
	CS_ACTIVE;//выбор дисплея
	DATA_PORT=dt;
	WR_STROBE;
	CS_IDLE;
}
//---------------------------------------------------------------
void TFT9341_Write8(unsigned char dt)
{
	DATA_PORT=dt;
	WR_STROBE;
}
//---------------------------------------------------------------
unsigned long TFT9341_ReadReg(unsigned char r)
{
	unsigned long id;
	unsigned char x;
	CS_ACTIVE;//выбор дисплея
	CD_COMMAND;//лапка в состоянии посылки команды
	TFT9341_Write8(r);
	setReadDir();
	CD_DATA;
	_delay_us(50);
	RD_ACTIVE;
	_delay_us(5);
	x=DATA_PIN;
	RD_IDLE;
	id=x;
	id<<=8;
	RD_ACTIVE;
```

```
_delay_us(5);
x=DATA_PIN;
RD_IDLE;
id|=x;
id<<=8;
RD_ACTIVE;
_delay_us(5);
x=DATA_PIN;
RD_IDLE;
id|=x;
id<<=8;
RD_ACTIVE;
_delay_us(5);
x=DATA_PIN;
RD_IDLE;
id|=x;
if(r==0xEF)
{
        id<<=8;
        RD_ACTIVE;
        _delay_us(5);
        x=DATA_PIN;
        RD_IDLE;
        id|=x;
    }
    CS_IDLE;
    setWriteDir();
    _delay_us(150);//stabilization time
    return id;
}
//------------------------------------------------------------
void TFT9341_reset(void)
{
    CS_IDLE;
    WR_IDLE;
```

```
	RD_IDLE;
	RESET_ACTIVE;
	_delay_ms(2);
	RESET_IDLE;
	CS_ACTIVE;
	TFT9341_SendCommand(0x01); //Software Reset
	for (uint8_t i=0;i<3;i++) WR_STROBE;
	CS_IDLE;
}
//------------------------------------------------------------
void TFT9341_SetRotation(unsigned char r)
{
	TFT9341_SendCommand(0x36);
	switch(r)
	{
		case 0:
		TFT9341_SendData(0x48);
		X_SIZE = 240;
		Y_SIZE = 320;
		break;
		case 1:
		TFT9341_SendData(0x28);
		X_SIZE = 320;
		Y_SIZE = 240;
		break;
		case 2:
		TFT9341_SendData(0x88);
		X_SIZE = 240;
		Y_SIZE = 320;
		break;
		case 3:
		TFT9341_SendData(0xE8);
		X_SIZE = 320;
		Y_SIZE = 240;
		break;
```

```
        }
}
//------------------------------------------------------------
void TFT9341_Flood(unsigned short color, unsigned long len)
{
        unsigned short blocks;
        unsigned char i, hi = color>>8, lo=color;
        CS_ACTIVE;
        CD_COMMAND;
        TFT9341_Write8(0x2C);
        CD_DATA;
        TFT9341_Write8(hi);
        TFT9341_Write8(lo);
        len--;
        blocks=(unsigned short)(len/64);//64 pixels/block
        if (hi==lo)
        {
                while(blocks--)
        {
                i=16;
                do
                {
                        WR_STROBE;WR_STROBE;WR_STROBE;WR_STROBE;//2bytes/pixel
                        WR_STROBE;WR_STROBE;WR_STROBE;WR_STROBE;//x4 pixel
                } while (--i);
        }
        //Fill any remaining pixels(1 to 64)
        for (i=(unsigned char)len&63;i--;)
        {
                WR_STROBE;
                WR_STROBE;
        }
}
else
{
```

```
            while(blocks--)
            {
                  i=16;
                  do
                  {
                        TFT9341_Write8(hi);TFT9341_Write8(lo);TFT9341_Write8(hi);TFT9341_Write8(lo);
                        TFT9341_Write8(hi);TFT9341_Write8(lo);TFT9341_Write8(hi);TFT9341_Write8(lo);
                  } while (--i);
            }
            //Fill any remaining pixels(1 to 64)
            for (i=(unsigned char)len&63;i--;)
            {
                  TFT9341_Write8(hi);
                  TFT9341_Write8(lo);
            }
      }
      CS_IDLE;
}
//------------------------------------------------------------
void TFT9341_WriteRegister32(unsigned char r, unsigned long d)
{
      CS_ACTIVE;
      CD_COMMAND;
      TFT9341_Write8(r);
      CD_DATA;
      _delay_us(1);
      TFT9341_Write8(d>>24);
      _delay_us(1);
      TFT9341_Write8(d>>16);
      _delay_us(1);
      TFT9341_Write8(d>>8);
      _delay_us(1);
      TFT9341_Write8(d);
      CS_IDLE;
}
```

```
//------------------------------------------------------------
void TFT9341_SetAddrWindow(unsigned int x1,unsigned int y1,unsigned int x2,unsigned int y2)
{
	unsigned long t;
	CS_ACTIVE;
	t = x1;
	t<<=16;
	t |= x2;
	TFT9341_WriteRegister32(0x2A,t);//Column Addres Set
	t = y1;
	t<<=16;
	t |= y2;
	TFT9341_WriteRegister32(0x2B,t);//Page Addres Set
	CS_IDLE;
}
//------------------------------------------------------------
void TFT9341_FillScreen(unsigned int color)
{
	TFT9341_SetAddrWindow(0,0,X_SIZE-1,Y_SIZE-1);
	TFT9341_Flood(color,(long)X_SIZE*(long)Y_SIZE);
}
//------------------------------------------------------------
void TFT9341_FillRectangle(unsigned int color,unsigned int x1, unsigned int y1,
                                              unsigned int x2, unsigned int y2)
{
	TFT9341_SetAddrWindow(x1, y1, x2, y2);
	TFT9341_Flood(color, (long)(x2-x1+1) * (long)(y2-y1+1));
}
//------------------------------------------------------------
void TFT9341_DrawPixel(int x, int y, unsigned int color)
{
	if((x<0)||(y<0)||(x>=X_SIZE)||(y>=Y_SIZE)) return;
	CS_ACTIVE;
	TFT9341_SetAddrWindow(x,y,X_SIZE-1,Y_SIZE-1);
	CS_ACTIVE;
```

```
        CD_COMMAND;
        TFT9341_Write8(0x2C);
        CD_DATA;
        TFT9341_Write8(color>>8);TFT9341_Write8(color);
        CS_IDLE;
}
//------------------------------------------------------------
void TFT9341_DrawLine(unsigned int color,unsigned int x1, unsigned int y1,
unsigned int x2, unsigned int y2)
{
        int steep = abs(y2-y1)>abs(x2-x1);
        if (steep)
        {
                swap(x1,y1);
                swap(x2,y2);
        }
        if(x1>x2)
        {
                swap(x1,x2);
                swap(y1,y2);
        }
        int dx,dy;
        dx=x2-x1;
        dy=abs(y2-y1);
        int err=dx/2;
        int ystep;
        if(y1<y2)     ystep = 1;
        else          ystep = -1;
        for (;x1<=x2;x1++)
        {
                if (steep)      TFT9341_DrawPixel(y1,x1,color);
                else TFT9341_DrawPixel(x1,y1,color);
                err-=dy;
                if (err<0)
                {
```

```
                        y1 += ystep;
                        err=dx;
                }
        }
}
//------------------------------------------------------------
void TFT9341_DrawRect(unsigned int color,unsigned int x1, unsigned int y1,
unsigned int x2, unsigned int y2)
{
        TFT9341_DrawLine(color,x1,y1,x2,y1);
        TFT9341_DrawLine(color,x2,y1,x2,y2);
        TFT9341_DrawLine(color,x1,y1,x1,y2);
        TFT9341_DrawLine(color,x1,y2,x2,y2);
}
//------------------------------------------------------------
void TFT9341_DrawCircle(unsigned int x0, unsigned int y0, int r, unsigned int color)
{
        int f = 1-r;
        int ddF_x=1;
        int ddF_y=-2*r;
        int x = 0;
        int y = r;
        TFT9341_DrawPixel(x0,y0+r,color);
        TFT9341_DrawPixel(x0,y0-r,color);
        TFT9341_DrawPixel(x0+r,y0,color);
        TFT9341_DrawPixel(x0-r,y0,color);
        while (x<y)
        {
                if (f>=0)
                {
                        y--;
                        ddF_y+=2;
                        f+=ddF_y;
                }
                x++;
```

```
            ddF_x+=2;
            f+=ddF_x;
            TFT9341_DrawPixel(x0+x,y0+y,color);
            TFT9341_DrawPixel(x0-x,y0+y,color);
            TFT9341_DrawPixel(x0+x,y0-y,color);
            TFT9341_DrawPixel(x0-x,y0-y,color);
            TFT9341_DrawPixel(x0+y,y0+x,color);
            TFT9341_DrawPixel(x0-y,y0+x,color);
            TFT9341_DrawPixel(x0+y,y0-x,color);
            TFT9341_DrawPixel(x0-y,y0-x,color);
        }
}
//------------------------------------------------------------
unsigned int TFT9341_RandColor(void)
{
        unsigned char c = rand()%8;
        switch(c)
        {
            case 0:
            return BLACK;
            break;
            case 1:
            return BLUE;
            break;
            case 2:
            return RED;
            break;
            case 3:
            return GREEN;
            break;
            case 4:
            return CYAN;
            break;
            case 5:
            return MAGENTA;
```

```
                break;
                case 6:
                return YELLOW;
                break;
                case 7:
                return WHITE;
                break;
        }
        return BLACK;
}
//------------------------------------------------------------
void TFT9341_Draw_Char(int x, int y, unsigned int color, unsigned int phone,
                                unsigned char charcode, unsigned char size)
{
        switch(size)
        {
                int i,h;
                case 1:
        for(h=0;h<8;h++)
        {
                for(i=0;i<8;i++)
                {
                        if ((pgm_read_byte(&chars8[charcode-0x20][h])>>(7-i))&0x01)
                        {
                                TFT9341_DrawPixel(x+i,y+h,color);
                        }
                        else
                        {
                                TFT9341_DrawPixel(x+i,y+h,phone);
                        }
                }
        }
        break;
case 2:
        for(h=0;h<16;h++)
```

```
{
        for(i=0;i<8;i++)
        {
                if ((pgm_read_byte(&chars16[charcode-0x20][h*2])>>(7-i))&0x01)
                {
                        TFT9341_DrawPixel(x+i,y+h,color);
                }
                else
                {
                        TFT9341_DrawPixel(x+i,y+h,phone);
                }
                if ((pgm_read_byte(&chars16[charcode-0x20][h*2+1])>>(7-i))&0x01)
                {
                        TFT9341_DrawPixel(x+i+8,y+h,color);
                }
                else
                {
                        TFT9341_DrawPixel(x+i+8,y+h,phone);
                }
                }
                }
                break;
        }
}
//------------------------------------------------------------
void TFT9341_String(unsigned int x, unsigned int y, unsigned int color, unsigned int phone,
                                char *str, unsigned char size)
{
        while (*str)
        {
                if ((x+(size*8))>X_SIZE)
                {
                        x = 1;
                        y = y + (size*8);
                }
```

```
                TFT9341_Draw_Char(x,y,color,phone,*str,size);
                x += size*8;
                *str++;
        }
}
//-----------------------------------------------------------
void TFT9341_ini(void)
{
        char str[10];
        port_ini();
        TFT9341_reset();
        _delay_ms(1000);
        dtt=TFT9341_ReadReg(0xD3);
        CS_ACTIVE;
        setpos(0,1);
        sprintf(str,"0x%08lX",dtt);
        str_lcd(str);
        TFT9341_SendCommand(0x01);//Software Reset
        TFT9341_SendCommand(0xCB);//Power Control A
        TFT9341_SendData(0x39);
        TFT9341_SendData(0x2C);
        TFT9341_SendData(0x00);
        TFT9341_SendData(0x34);
        TFT9341_SendData(0x02);
        TFT9341_SendCommand(0xCF);//Power Control B
        TFT9341_SendData(0x00);
        TFT9341_SendData(0xC1);
        TFT9341_SendData(0x30);
        TFT9341_SendCommand(0xE8);//Driver timing control A
        TFT9341_SendData(0x85);
        TFT9341_SendData(0x00);
        TFT9341_SendData(0x78);
        TFT9341_SendCommand(0xEA);//Driver timing control B
        TFT9341_SendData(0x00);
        TFT9341_SendData(0x00);
```

```
TFT9341_SendCommand(0xED);//Power on Sequence control
TFT9341_SendData(0x64);
TFT9341_SendData(0x03);
TFT9341_SendData(0x12);
TFT9341_SendData(0x81);
TFT9341_SendCommand(0xF7);//Pump ratio control
TFT9341_SendData(0x20);
TFT9341_SendCommand(0xC0);//Power Control 1
TFT9341_SendData(0x10);
TFT9341_SendCommand(0xC1);//Power Control 2
TFT9341_SendData(0x10);
TFT9341_SendCommand(0xC5);//VCOM Control 1
TFT9341_SendData(0x3E);
TFT9341_SendData(0x28);
TFT9341_SendCommand(0xC7);//VCOM Control 2
TFT9341_SendData(0x86);
TFT9341_SetRotation(0);
TFT9341_SendCommand(0x3A);//Pixel Format Set
TFT9341_SendData(0x55);//16bit
TFT9341_SendCommand(0xB1);
TFT9341_SendData(0x00);
TFT9341_SendData(0x18);// Частота кадров 79 Гц
TFT9341_SendCommand(0xB6);//Display Function Control
TFT9341_SendData(0x08);
TFT9341_SendData(0x82);
TFT9341_SendData(0x27);//320 строк
TFT9341_SendCommand(0xF2);//Enable 3G (пока не знаю что это за режим)
TFT9341_SendData(0x00);//не включаем
TFT9341_SendCommand(0x26);//Gamma set
TFT9341_SendData(0x01);//Gamma Curve (G2.2) (Кривая цветовой гаммы)
TFT9341_SendCommand(0xE0);//Positive Gamma  Correction
TFT9341_SendData(0x0F);
TFT9341_SendData(0x31);
TFT9341_SendData(0x2B);
TFT9341_SendData(0x0C);
```

```
	TFT9341_SendData(0x0E);
	TFT9341_SendData(0x08);
	TFT9341_SendData(0x4E);
	TFT9341_SendData(0xF1);
	TFT9341_SendData(0x37);
	TFT9341_SendData(0x07);
	TFT9341_SendData(0x10);
	TFT9341_SendData(0x03);
	TFT9341_SendData(0x0E);
	TFT9341_SendData(0x09);
	TFT9341_SendData(0x00);
	TFT9341_SendCommand(0xE1);//Negative Gamma  Correction
	TFT9341_SendData(0x00);
	TFT9341_SendData(0x0E);
	TFT9341_SendData(0x14);
	TFT9341_SendData(0x03);
	TFT9341_SendData(0x11);
	TFT9341_SendData(0x07);
	TFT9341_SendData(0x31);
	TFT9341_SendData(0xC1);
	TFT9341_SendData(0x48);
	TFT9341_SendData(0x08);
	TFT9341_SendData(0x0F);
	TFT9341_SendData(0x0C);
	TFT9341_SendData(0x31);
	TFT9341_SendData(0x36);
	TFT9341_SendData(0x0F);
	TFT9341_SendCommand(0x11);//Выйдем из спящего режим
	_delay_ms(150);
	TFT9341_SendCommand(0x29);//Включение дисплея
	TFT9341_SendData(0x2C);
	_delay_ms(150);
}
```

==========

ili9341.h

```
---------------------------------------------------------------------------------------------------------
#ifndef ILI9341_H_
#define ILI9341_H_

#include <stdlib.h>
#include "main.h"
#include "twi.h"
#include "lcdtwi.h"

#define swap(a,b) {int16_t t=a;a=b;b=t;}
#define  DATA_DDR DDRD
#define  DATA_PORT PORTD
#define  DATA_PIN PIND
#define  COMMAND_DDR DDRB
#define  COMMAND_PORT PORTB
#define  LCD_CS 2//Chip Select
#define  LCD_CD 1//Command/Data
#define  LCD_WR 3//LCD Write
#define  LCD_RD 4//LCD Read
#define  LCD_RESET 0//LCD Reset
#define  RESET_IDLE COMMAND_PORT|=(1<<LCD_RESET)
#define  CS_IDLE COMMAND_PORT|=(1<<LCD_CS)
#define  WR_IDLE COMMAND_PORT|=(1<<LCD_WR)
#define  RD_IDLE COMMAND_PORT|=(1<<LCD_RD)
#define  RESET_ACTIVE COMMAND_PORT&=~(1<<LCD_RESET)
#define  CS_ACTIVE COMMAND_PORT&=~(1<<LCD_CS)
#define  WR_ACTIVE COMMAND_PORT&=~(1<<LCD_WR)
#define  RD_ACTIVE COMMAND_PORT&=~(1<<LCD_RD)
#define  CD_COMMAND COMMAND_PORT&=~(1<<LCD_CD)
#define  CD_DATA COMMAND_PORT|=(1<<LCD_CD)

#define  BLACK 0x0000
#define  BLUE 0x001F
#define  RED 0x0F800
#define  GREEN 0x07E0
```

```
#define  CYAN 0x07FF
#define  MAGENTA 0xF81F
#define  YELLOW 0xFFE0
#define  WHITE 0xFFFF

#define   setReadDir() DATA_DDR=0x00
#define   setWriteDir() DATA_DDR=0xFF

#define  WR_STROBE {WR_ACTIVE;WR_IDLE;}

void TFT9341_ini(void);
void TFT9341_FillScreen(unsigned int color);
void TFT9341_FillRectangle(unsigned int color,unsigned int x1, unsigned int y1,
                                    unsigned int x2, unsigned int y2);
unsigned int TFT9341_RandColor(void);
void TFT9341_DrawPixel(int x, int y, unsigned int color);
void TFT9341_DrawLine(unsigned int color,unsigned int x1, unsigned int y1,
unsigned int x2, unsigned int y2);
void TFT9341_DrawRect(unsigned int color,unsigned int x1, unsigned int y1,
unsigned int x2, unsigned int y2);
void TFT9341_DrawCircle(unsigned int x0, unsigned int y0, int r, unsigned int color);
void TFT9341_Draw_Char(int x, int y, unsigned int color, unsigned int phone,
unsigned char charcode, unsigned char size);
void TFT9341_String(unsigned int x, unsigned int y, unsigned int color, unsigned int phone,
char *str, unsigned char size);

#endif /* ILI9341_H_ */
```