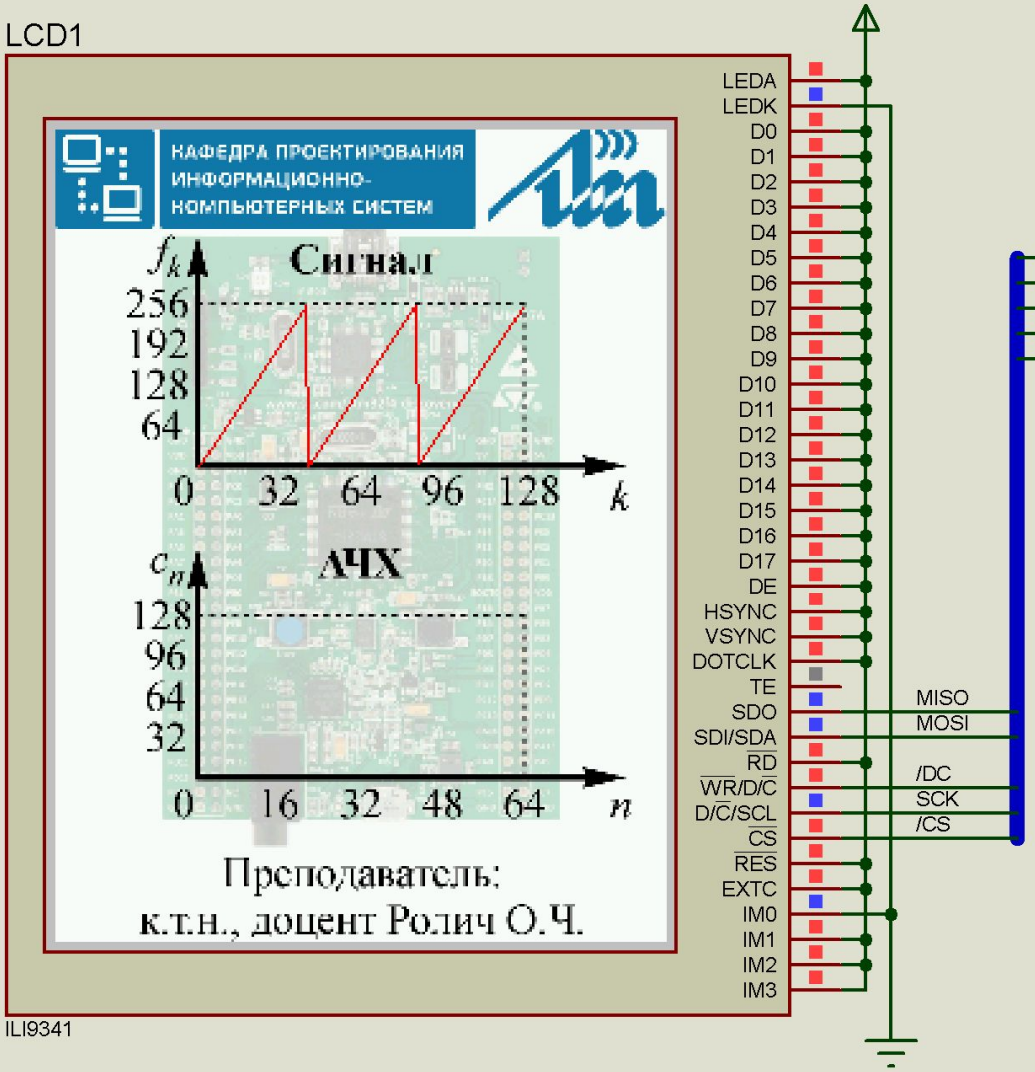
КАФЕДРА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫХ СИСТЕМ

Управление портами микроконтроллера в режиме альтернативных функций

Преподаватель:
Олег Чеславович Ролич
к.т.н., доцент

Постановка задачи

Согласно заданным принципиальной схеме и управляющей программе, на базе микроконтроллера ATSAM3N4C и TFT-дисплея ILI9341 построить модель устройства отображения графической информации и обеспечить возможность пошаговой его отладки



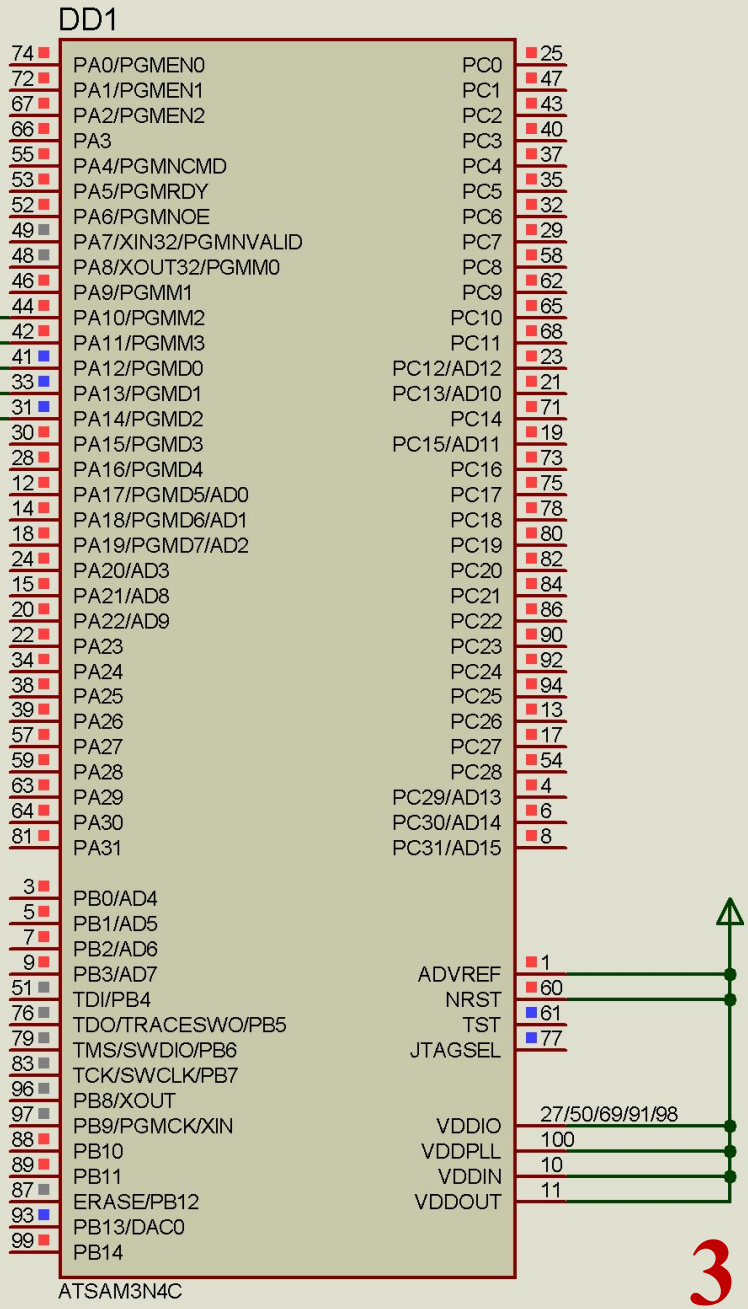


Схема в Proteus

Постановка задачи

Новый проект рекомендуется создавать на базе одного из предыдущих, ранее отлаженных, например, на основе «MMVP_LW1_ATSAM3N4C_LedDisplay» путём его копирования через буфер обмена

Программа управления TFT-дисплеем посредством SPI

Файлы

startup_sam3n.c и *flash.ld*

остаются без изменений по сравнению с проектом «MMVP_LW1_ATSAM3N4C_LedDisplay»

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
#include "sam3n4c.h"
#include "pmc.h"
#include "pio.h"
#include "board.h"

#define MAINCK  (32000000)    // 32 MHz

#define N  128
// TFT resolution 240 * 320
#define ILI9341_TFTWIDTH  240
#define ILI9341_TFTHEIGHT 320
```

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
#define MIN_X  0
#define MIN_Y  0
#define MAX_X(ILI9341_TFTWIDTH - 1)
#define MAX_Y(ILI9341_TFTHEIGHT - 1)
/**
  * @brief Various internal ILI9341 registers name labels
  */
#define ILI9341_READ_ID10xDA
#define ILI9341_READ_ID20xDB
```

Программа управления TFT-дисплеем посредством SPI

```
#define ILI9341_NOP    0x00
#define ILI9341_SWRESET0x01
#define ILI9341_RDDID  0x04
#define ILI9341_RDDST  0x09
```

*Файл **main.c***

```
#define ILI9341_SLPIN    0x10
#define ILI9341_SLPOUT   0x11
#define ILI9341_PTLON  0x12
#define ILI9341_NORON 0x13
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
#define ILI9341_RDMODE    0x0A
#define ILI9341_RDMADCTL  0x0B
#define ILI9341_RDPIXFMT  0x0C
#define ILI9341_RDIMGFMT  0x0A
#define ILI9341_RDSELFDIAG 0x0F

#define ILI9341_INVOFF   0x20
#define ILI9341_INVON    0x21
#define ILI9341_GAMMASET  0x26
#define ILI9341_DISPOFF 0x28
#define ILI9341_DISPON  0x29
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
#define ILI9341_CASET    0x2A
#define ILI9341_RASET    0x2B
#define ILI9341_RAMWR 0x2C
#define ILI9341_RAMRD  0x2E
#define ILI9341_PTLAR    0x30
#define ILI9341_MADCTL    0x36
#define ILI9341_PIXFMT  0x3A
#define ILI9341_FRMCTR1   0xB1
#define ILI9341_FRMCTR2   0xB2
#define ILI9341_FRMCTR3   0xB3
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
#define ILI9341_PWCTR1 0xC0
#define ILI9341_PWCTR2 0xC1
#define ILI9341_PWCTR3 0xC2
#define ILI9341_PWCTR4 0xC3
#define ILI9341_PWCTR5 0xC4
#define ILI9341_VMCTR1 0xC5
#define ILI9341_VMCTR2 0xC7
#define ILI9341_RDID1 0xDA
#define ILI9341_RDID2 0xDB
#define ILI9341_RDID3 0xDC
#define ILI9341_RDID4 0xDD
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
#define ILI9341_GMCTRP1   0xE0
#define ILI9341_GMCTRN1   0xE1
/*
#define ILI9341_PWCTR6  0xFC
*/


// Color definitions
#define ILI9341_BLACK    0x0000   /*   0,   0,   0 */
#define ILI9341_NAVY  0x000F   /*   0,   0, 128 */
#define ILI9341_DARKGREEN    0x03E0   /*   0, 128,   0 */
#define ILI9341_DARKCYAN 0x03EF   /*   0, 128, 128 */
```

Программа управления TFT-дисплеем посредством SPI

```
#define ILI9341_MAROON  0x7800     /* 128,   0,   0 */
#define ILI9341_PURPLE    0x780F     /* 128,   0, 128 */
#define ILI9341_OLIVE 0x7BE0     /* 128, 128,   0 */
#define ILI9341_LIGHTGREY   0xC618     /* 192, 192, 192 */
#define ILI9341_DARKGREY   0x7BEF     /* 128, 128, 128 */
#define ILI9341_BLUE   0x001F     /*   0,   0, 255 */
#define ILI9341_GREEN0x07E0     /*   0, 255,   0 */
#define ILI9341_CYAN   0x07FF     /*   0, 255, 255 */
#define ILI9341_RED     0xF800     /* 255,   0,   0 */
#define ILI9341_MAGENTA     0xF81F     /* 255,   0, 255 */
#define ILI9341_YELLOW   0xFFE0     /* 255, 255,   0 */
#define ILI9341_WHITE     0xFFFF     /* 255, 255, 255 */
#define ILI9341_ORANGE   0xFD20     /* 255, 165,   0 */
#define ILI9341_GREENYELLOW 0xAFE5     /* 173, 255,  47 */
#define ILI9341_PINK    0xF81F
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
/* PORTA[12 - 14]:MISO, MOSI, SCK
 * PA9 --- /CS
 * PA10 --- D/C
 */
/**
  * @brief ILI9341 control pins definition
  */
/* ILI9341 CS control ( /CHIP SELECT )*/
#define PA10_CS_PIN    {PIO_PA10, (AT91S_PIO *) PIOA,
AT91C_ID_PIOA, PIO_OUTPUT_1, PIO_DEFAULT}
/* ILI9341 D/C control ( DATA/COMMAND )*/
```

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
/* SPI:PA12 --- MISO, PA13 --- MOSI, PA14 --- SCK
 * (p. 431 doc. "sam3n_series.pdf")
 */
#define SPI_MISO_PIN    {PIO_PA12A_MISO,
 (AT91S_PIO *) PIOA, AT91C_ID_PIOA,
PIO_PERIPH_A, PIO_DEFAULT}
#define SPI_MOSI_PIN    {PIO_PA13A_MOSI,
 (AT91S_PIO *) PIOA, AT91C_ID_PIOA,
PIO_PERIPH_A, PIO_DEFAULT}
#define SPI_SCK_PIN   {PIO_PA14A_SPCK,
```

Программа управления TFT-дисплеем посредством SPI

```
/// Pins to configure for TFT LCD control & data bus.
Pin TFTControlAndBusPins[] = {
    PA10_CS_PIN,
    PA11_DC_PIN,
    SPI_MISO_PIN,
    SPI_MOSI_PIN,
    SPI_SCK_PIN,
};
#define CS_INDEX   0
#define DC_INDEX   1
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
volatile int32_t ITM_RxBuffer;
extern const unsigned char gImage_image[];
unsigned short image_coor[] = {0, 0, 0, 0};

void Delay ( unsigned long nTime );
void SPI_Init();
void SPI_Transfer ( uint8_t uData );
void WRITE_DATA ( uint8_t data );
uint8_t Read_Register ( uint8_t Addr, uint8_t xParameter);
void TFT_SendCMD ( uint8_t index );
uint8_t readID ( void );
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
void TFT_Init ( void );
void TFT_SetCol ( uint16_t StartCol, uint16_t EndCol );
void TFT_SetPage ( uint16_t StartPage, uint16_t EndPage );
void TFT_SetXY ( uint16_t poX, uint16_t poY );
void TFT_SetPixel ( uint16_t poX, uint16_t poY,uint16_t color );
void TFT_DrawLine ( uint16_t x0, uint16_t y0,
uint16_t x1, uint16_t y1, uint16_t color);
void TFT_DrawBackgroundLine ( uint16_t x0, uint16_t y0,
uint16_t x1, uint16_t y1, const unsigned char * image );
void TFT_DrawRectangle ( uint16_t poX, uint16_t
poY, uint16_t length, uint16_t width, uint16_t color );
```

Программа управления TFT-дисплеем посредством SPI

```
void TFT_DrawCircle ( int poX, int poY, int r, uint16_t color );
void TFT_FillCircle (int poX, int poY, int r, uint16_t color );
void TFT_DrawTraingle ( int poX1, int poY1, int poX2, int poY2, int poX3, int poY3, uint16_t color );
void TFT_FillScreen ( uint16_t XL, uint16_t XR, uint16_t YU, uint16_t YD, uint16_t color );
void TFT_FillRectangle ( uint16_t poX, uint16_t poY, uint16_t length, uint16_t width, uint16_t color );
void TFT_DrawHorizontalLine ( uint16_t poX, uint16_t poY, uint16_t length, uint16_t color);
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

```
void TFT_DrawVerticalLine ( uint16_t poX, uint16_t poY, uint16_t length, uint16_t color );
void _fillScreen();
```

*Файл **main.c***

```
/*
 * TFT_CS control block definitions
 */
/* /CS = 0*/
void TFT_CS_LOW() {
 PIO_Clear ( TFTControlAndBusPins + CS_INDEX );
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
/* /CS = 1*/
void TFT_CS_HIGH() {
    PIO_Set ( TFTControlAndBusPins + CS_INDEX );
}
/*
 * TFT_DC control block definitions
 */
/* D/C == 1 (DATA)*/
void TFT_DC_HIGH() {
   PIO_Set ( TFTControlAndBusPins + DC_INDEX );
}
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
/* D/C == 0 (COMMAND)*/
void TFT_DC_LOW() {
PIO_Clear ( TFTControlAndBusPins + DC_INDEX );
}
uint16_t f[N];
int main ( void ) {
image_coor[2]= ((unsigned short)gImage_image[2] << 8) + gImage_image[3];
image_coor[3]= ((unsigned short)gImage_image[4] << 8) + gImage_image[5];


// Настройка системного таймера для управления задержками: 1 tick = 1 ms
SysTick_Config ( MAINCK / 100000 );
```

Программа управления TFT-дисплеем посредством SPI

```
// Инициализация SPI для управления TFT-дисплеем
SPI_Init();
// Инициализация TFT-дисплея
TFT_Init();
while (1) {
    uint8_t q;
    uint16_t k;
    for (q = 1; q < 5; q++) {
        for (k = 0; k < N; k++) {
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
// Координаты области для отображения сигнала:
//  левый нижний угол (56, 132)
//  правый верхний угол (184, 68)
//  Высота области = 64
//  Ширина области = 128
   for (k = 0; k < (N-1); k++) {
     TFT_DrawLine ( 56 + k, 132 - f[k] / 4,
     57 + k, 132 - f[k+1] / 4, ILI9341_RED);
   }
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
// Координаты области для отображения АЧХ:
//  левый нижний угол (56, 255)
//  правый верхний угол (184, 191)
//  Высота области = 64
//  Ширина области = 128
Delay(20000);
// Удаление сигнала путём восстановления фонового изображения
for (k = 0; k < (N-1); k++) {
    TFT_DrawBackgroundLine ( 56 + k, 132 - f[k] / 4,
    57 + k, 132 - f[k+1] / 4, gImage_image );
} } }
return 0;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
// Инициализация SPI для управления TFT-дисплеем
void SPI_Init() {
int iIndex;
unsigned int configuration;

for (iIndex = 0;
iIndex < sizeof(TFTControlAndBusPins) / sizeof(Pin);
iIndex++) {
PIO_Configure ( TFTControlAndBusPins + iIndex,
PIO_LISTSIZE ( TFTControlAndBusPins[iIndex] ) );
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
/* Enable SPI peripheral clock
 *  p. 357 "PMC Peripheral Clock E
 *p. 34 "Peripheral Identifiers" ---> ID_SPI == 21
 */
PMC_EnablePeripheral ( ID_SPI );// PMC - Power Management Controller

configuration = SPI_MR_MSTR;
// Configure the SPI in the desired mode
SPI_Configure ( (AT91S_SPI *) SPI, ID_SPI, configuration );
```

Программа управления TFT-дисплеем посредством SPI

```
configuration = SPI_CSR_BITS_8_BIT
    | SPI_CSR_SCBR ( 8 );
SPI_ConfigureNPCS ( (AT91S_SPI *) SPI, 0,
configuration);
```

*Файл **main.c***

```
// Enable SPI
SPI_Enable ( (AT91S_SPI *) SPI );
}
void SPI_Transfer ( uint8_t uData ) {
SPI_Write ( (AT91S_SPI *) SPI, 0, uData );// NPCS0 = 0
}
```

Программа управления TFT-дисплеем посредством SPI

```
void TFT_SendCMD ( uint8_t index )
{
    TFT_DC_LOW();
    TFT_CS_LOW();
    SPI_Transfer ( index );
    TFT_CS_HIGH();
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

```
void TFT_SendData ( uint16_t data ) {
    uint8_t data1 = data >> 8;
    uint8_t data2 = data & 0xff;
    TFT_DC_HIGH();
    TFT_CS_LOW();
     SPI_Transfer ( data1 );
     SPI_Transfer ( data2 );
    TFT_CS_HIGH();
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

```
void WRITE_DATA( uint8_t data )
{
  TFT_DC_HIGH();
  TFT_CS_LOW();
   SPI_Transfer ( data );
  TFT_CS_HIGH();
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

```
uint8_t Read_Register ( uint8_t Addr, uint8_t xParameter) {
    uint8_t data = 0;
    TFT_SendCMD ( 0xD9 ); /* ext command                    */
    WRITE_DATA ( 0x10 + xParameter ); /* 0x11 is the first
    Parameter  */
    TFT_DC_LOW();
    TFT_CS_LOW();
    SPI_Transfer ( Addr );
    TFT_DC_HIGH();
    SPI_Transfer ( 0 );
    data = SPI_Read ( (AT91S_SPI *) SPI );
    TFT_CS_HIGH();
    return data; }
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
uint8_t readID ( void ) {
    uint8_t i = 0;
    uint8_t data[3] ;
    uint8_t ID[3] = { 0x00, 0x93, 0x41 };
    for ( i = 0; i < 3; i++ ) {
        data[i] = Read_Register ( 0xD3, i + 1 );
        if ( data[i] != ID[i] ) {
            return 0;
    } }
    return 1;
}
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
// Инициализация TFT-дисплея
void TFT_Init ( void ) {
    uint8_t i = 0;

    TFT_CS_HIGH();
    TFT_DC_HIGH();
    for ( i = 0; i < 3; i++ ) {
        readID();
    }
   Delay ( 5 );
   TFT_SendCMD ( ILI9341_SWRESET );
   Delay ( 2 );
```

Программа управления TFT-дисплеем посредством SPI

```
TFT_SendCMD ( ILI9341_PWCTR1 );/* Power control */
 WRITE_DATA ( 0x1B ); /* VRH[5:0]                 */

TFT_SendCMD ( ILI9341_PWCTR2 );/* Power control */
 WRITE_DATA ( 0x10 );    /* SAP[2:0];BT[3:0] */

TFT_SendCMD ( ILI9341_VMCTR1 );/* VCM control*/
 WRITE_DATA ( 0x3F );
 WRITE_DATA ( 0x3C );
TFT_SendCMD ( ILI9341_VMCTR2 );/* VCM control2*/
 WRITE_DATA ( 0XB7 );
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
TFT_SendCMD ( ILI9341_MADCTL );/* Memory Access Control */
WRITE_DATA ( 0x08 );
TFT_SendCMD ( ILI9341_PIXFMT );
WRITE_DATA ( 0x55 );
TFT_SendCMD ( ILI9341_FRMCTR1 );
WRITE_DATA ( 0x00 );
WRITE_DATA ( 0x1B );
TFT_SendCMD ( ILI9341_DFUNCTR );/* Display Function Control */
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
    TFT_SendCMD ( ILI9341_GAMMASET );/*
Gamma curve selected */
    WRITE_DATA ( 0x01 );
  TFT_SendCMD ( ILI9341_GMCTRP1 ); /* Set Gamma */
    WRITE_DATA ( 0x0F );
    WRITE_DATA ( 0x2A );
    WRITE_DATA ( 0x28 );
    WRITE_DATA ( 0x08 );
    WRITE_DATA ( 0x0E );
    WRITE_DATA ( 0x08 );
    WRITE_DATA ( 0x54 );
```

Программа управления TFT-дисплеем посредством SPI

```
WRITE_DATA ( 0XA9 );
WRITE_DATA ( 0x43 );
WRITE_DATA ( 0x0A );
WRITE_DATA ( 0x0F );
WRITE_DATA ( 0x00 );
WRITE_DATA ( 0x00 );
WRITE_DATA ( 0x00 );
WRITE_DATA ( 0x00 );
```

Файл ***main.c***

Программа управления TFT-дисплеем посредством SPI

```
TFT_SendCMD ( ILI9341_GMCTRN1 );/* Set Gamma */
    WRITE_DATA ( 0x00 );
    WRITE_DATA ( 0x15 );
    WRITE_DATA ( 0x17 );
    WRITE_DATA ( 0x07 );
    WRITE_DATA ( 0x11 );
    WRITE_DATA ( 0x06 );
    WRITE_DATA ( 0x2B );
    WRITE_DATA ( 0x56 );
    WRITE_DATA ( 0x3C );
    WRITE_DATA ( 0x05 );
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
    WRITE_DATA ( 0x10 );
    WRITE_DATA ( 0x0F );
    WRITE_DATA ( 0x3F );
    WRITE_DATA ( 0x3F );
    WRITE_DATA ( 0x0F );
  TFT_SendCMD ( ILI9341_SLPOUT );/* Exit Sleep */
    Delay ( 1 );
  TFT_SendCMD ( ILI9341_DISPON );/* Display on */
    _fillScreen();
}
```

Программа управления TFT-дисплеем посредством SPI

Файл ***main.c***

```
void TFT_SetCol ( uint16_t StartCol,
uint16_t EndCol ) {
   /* Column Command address       */
TFT_SendCMD(ILI9341_CASET );
   TFT_SendData ( StartCol );
   TFT_SendData ( EndCol );
}
```

Программа управления TFT-дисплеем посредством SPI

```
void TFT_SetPage ( uint16_t
StartPage, uint16_t EndPage ) {
   /* Column Command address       */
   TFT_SendCMD ( ILI9341_RASET );
   TFT_SendData ( StartPage );
   TFT_SendData ( EndPage );
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

```
int16_t constrain ( int16_t iCoor, int16_t
      iCoorL, int16_t iCoorR ) {
   if (iCoor >= iCoorR) {
      return (iCoorR - 1);
   } else if (iCoor < iCoorL) {
      return iCoorL;
   } else {
      return iCoor;
   }
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

```
void TFT_FillScreen(uint16_t XL,uint16_t XR,
   uint16_t YU, uint16_t YD,uint16_t color) {
   unsigned long  XY = 0;
   unsigned long i = 0;


   if ( XL > XR ) {
       XL = XL ^ XR;
       XR = XL ^ XR;
       XL = XL ^ XR;
   }
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
if ( YU > YD ) {
    YU = YU ^ YD;
    YD = YU ^ YD;
    YU = YU ^ YD;
}
XL = constrain ( XL, MIN_X, MAX_X );
XR = constrain ( XR, MIN_X, MAX_X );
YU = constrain ( YU, MIN_Y, MAX_Y );
YD = constrain ( YD, MIN_Y, MAX_Y );
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
XY = (XR - XL + 1);
XY = XY * (YD - YU + 1);

TFT_SetCol ( XL, XR );
TFT_SetPage ( YU, YD );
/* start to write to display ram */
TFT_SendCMD ( ILI9341_RAMWR );

TFT_DC_HIGH();
TFT_CS_LOW();
```

Программа управления TFT-дисплеем посредством SPI

```
    uint8_t Hcolor = color >> 8;
    uint8_t Lcolor = color & 0xff;
    for ( i = 0; i < XY; i++ ) {
        SPI_Transfer ( Hcolor );
        SPI_Transfer ( Lcolor );
    }

    TFT_CS_HIGH();
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
void _fillScreen ( void ) {
    uint32_t i;
    uint32_t imax = 2 * image_coor[2] * image_coor[3];
    TFT_SetCol ( 0, MAX_X );
    TFT_SetPage ( 0, MAX_Y );
    TFT_SendCMD ( ILI9341_RAMWR );/* start to write to
                                    display ram */
    TFT_DC_HIGH();
    TFT_CS_LOW();
    for ( i = 0; i < imax; i++) {
        SPI_Transfer ( gImage_image[i + 8] );
   }
   TFT_CS_HIGH(); }
```

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
void TFT_SetXY ( uint16_t poX, uint16_t poY ) {
    TFT_SetCol ( poX, poX );
    TFT_SetPage ( poY, poY );
    TFT_SendCMD ( ILI9341_RAMWR );
}

void TFT_SetPixel ( uint16_t poX, uint16_t
          poY, uint16_t color ) {
    TFT_SetXY ( poX, poY );
    TFT_SendData ( color );
}
```

Программа управления TFT-дисплеем посредством SPI

```
void TFT_FillRectangle ( uint16_t poX, uint16_t poY,
    uint16_t length, uint16_t width, uint16_t color ) {
 TFT_FillScreen ( poX, poX+length, poY, poY+width, color );
}
void TFT_DrawHorizontalLine ( uint16_t poX, uint16_t
poY, uint16_t length, uint16_t color)  {
    uint16_t i;
    TFT_SetCol ( poX, poX + length );
    TFT_SetPage ( poY, poY );
    TFT_SendCMD ( ILI9341_RAMWR );
    for ( i = 0; i < length; i++ ) {
        TFT_SendData ( color );
    } }
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

```
void TFT_DrawVerticalLine ( uint16_t poX,
uint16_t  poY, uint16_t length, uint16_t color ){
    uint16_t i;
    TFT_SetCol ( poX, poX );
    TFT_SetPage ( poY, poY + length );
    TFT_SendCMD ( ILI9341_RAMWR );
    for ( i = 0; i < length; i++ ) {
        TFT_SendData ( color );
    }
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
void TFT_DrawLine ( uint16_t x0, uint16_t y0,
    uint16_t x1, uint16_t y1, uint16_t color) {
    int x = x1 - x0;
    int y = y1 - y0;
    int dx = x < 0 ? -x : x, sx = x0 < x1 ? 1 : -1;
    int dy = y < 0 ? y : -y, sy = y0 < y1 ? 1 : -1;
    int err = dx + dy, e2;/* error value e_xy              */
    for (;;) {/* loop                                     */
        TFT_SetPixel ( x0, y0, color );
        e2 = 2 * err;
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
if (e2 >= dy) {/* e_xy+e_x > 0                    */
    if (x0 == x1) {
        break;
    }
    err += dy;
    x0 += sx;
}
if (e2 <= dx) {/* e_xy+e_y < 0                    */
    if (y0 == y1) {
        break;
    }
    err += dx;
    y0 += sy;}}}
```

Программа управления TFT-дисплеем посредством SPI

```
void TFT_DrawBackgroundLine ( uint16_t x0,
uint16_t y0, uint16_t x1, uint16_t y1, const  unsigned
char * image ) {
    int iPixelIndex;
    unsigned short color;
    unsigned short width = ((unsigned short)image[2] << 8) + image[3];
    int x = x1 - x0;
    int y = y1 - y0;
    int dx = x < 0 ? -x : x, sx = x0 < x1 ? 1 : -1;
    int dy = y < 0 ? y : -y, sy = y0 < y1 ? 1 : -1;
    int err = dx + dy, e2;/* error value e_xy
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

```
for (;;) {/* loop                                    */
    iPixelIndex = 2 * (y0 * width + x0) + 8;
color = ((unsigned short)image[iPixelIndex] << 8) + image[iPixelIndex + 1];
    TFT_SetPixel ( x0, y0, color );
    e2 = 2 * err;
    if (e2 >= dy) {/* e_xy+e_x > 0                   */
        if (x0 == x1) {
            break;
        }
        err += dy;
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

```
        if (e2 <= dx) {/* e_xy+e_y < 0 */
            if (y0 == y1) {
                break;
            }
            err += dx;
            y0 += sy;
        }
    }
}
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

```
void TFT_DrawRectangle ( uint16_t poX, uint16_t
poY, uint16_t length, uint16_t width, uint16_t   color )
{
    TFT_DrawHorizontalLine(poX, poY, length, color);
    TFT_DrawHorizontalLine ( poX, poY + width,
    length, color );
    TFT_DrawVerticalLine ( poX, poY, width, color );
    TFT_DrawVerticalLine ( poX + length, poY, width,
        color );
}
```

*Файл **main.c***

Программа управления TFT-дисплеем посредством SPI

*Файл **main.c***

```
void TFT_DrawCircle ( int poX, int poY, int r,
    uint16_t color ) {
    int x = -r, y = 0, err = 2 - 2 * r, e2;
    do {
        TFT_SetPixel ( poX - x, poY + y, color );
        TFT_SetPixel ( poX + x, poY + y, color );
        TFT_SetPixel ( poX + x, poY - y, color );
        TFT_SetPixel ( poX - x, poY - y, color );
        e2 = err;
```

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

```
        if (e2 <= y) {
            err += ++y * 2 + 1;
            if (-x == y && e2 <= x) {
                e2 = 0;
            }
        }
        if (e2 > x) {
            err += ++x * 2 + 1;
        }
    } while (x <= 0);
}
```

Программа управления TFT-дисплеем посредством SPI

```
void TFT_FillCircle(int poX, int poY, int r, uint16_t color ){
    int x = -r, y = 0, err = 2 - 2 * r, e2;
    do {
TFT_DrawVerticalLine(poX - x, poY - y, 2 * y, color);
TFT_DrawVerticalLine(poX + x, poY - y, 2* y, color);
        e2 = err;
        if (e2 <= y) {
            err += ++y * 2 + 1;
            if (-x == y && e2 <= x) {
                e2 = 0;
            }
```

Файл **main.c**

Программа управления TFT-дисплеем посредством SPI

Файл **main.c**

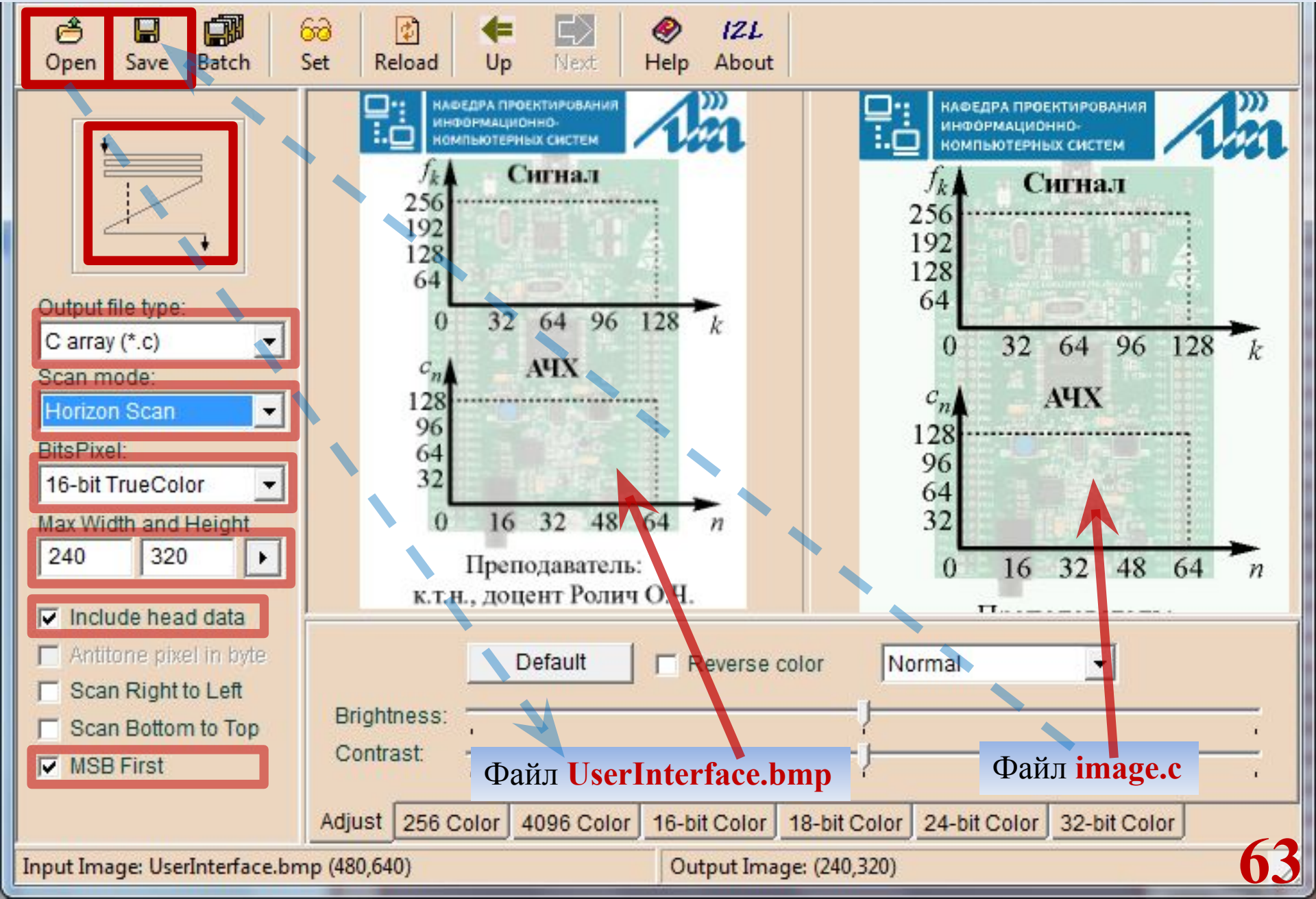
```
    }
    if (e2 > x) {
      err += ++x * 2 + 1;
    }
  } while (x <= 0);
}
```

Программа управления TFT-дисплеем посредством SPI

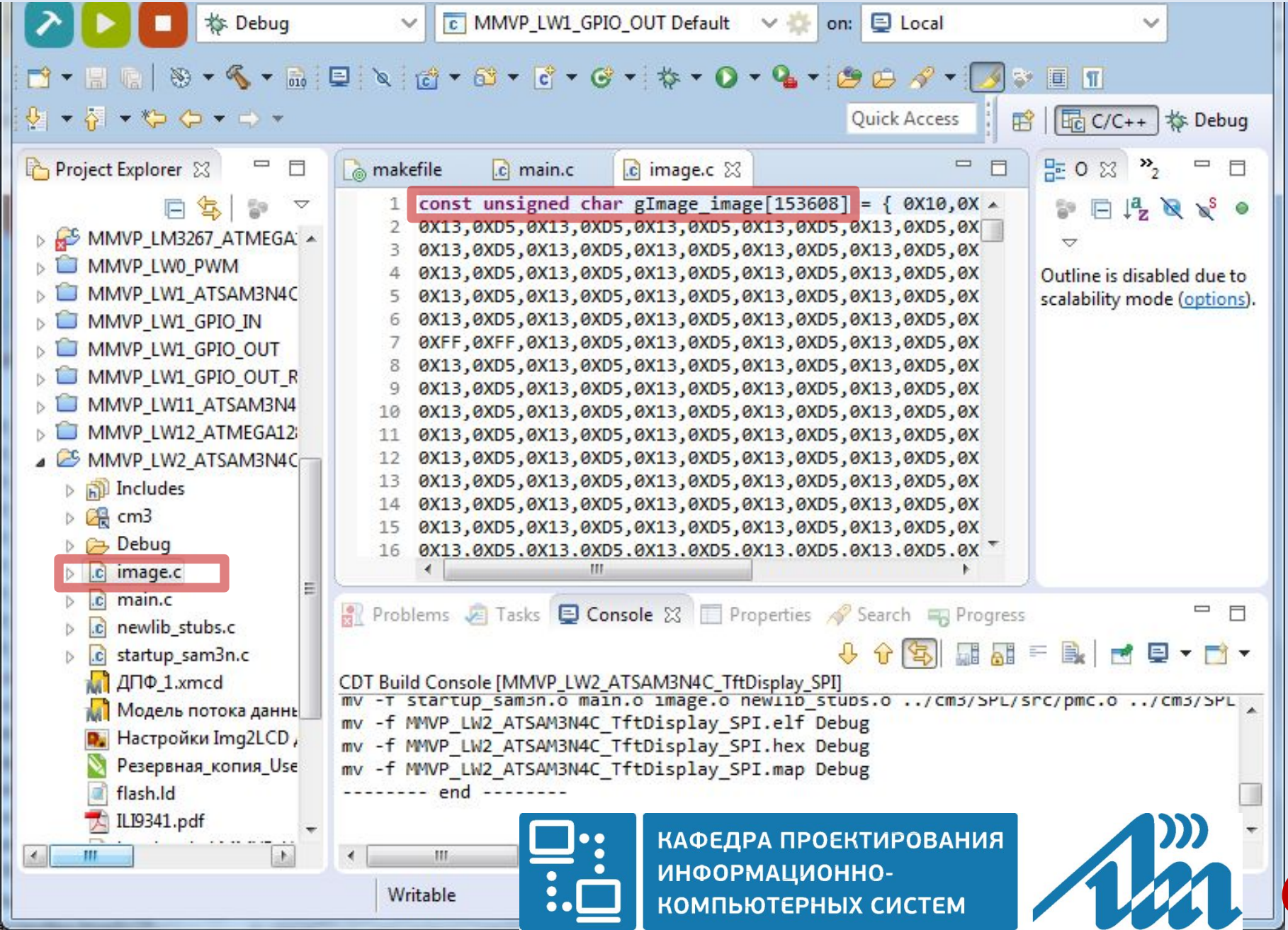
Файл main.c

```
void TFT_DrawTraingle ( int poX1, int poY1,
    int poX2, int poY2, int poX3, int poY3,
    uint16_t color ) {
   TFT_DrawLine ( poX1, poY1, poX2, poY2, color );
   TFT_DrawLine ( poX1, poY1, poX3, poY3, color );
   TFT_DrawLine ( poX2, poY2, poX3, poY3, color );
}
```

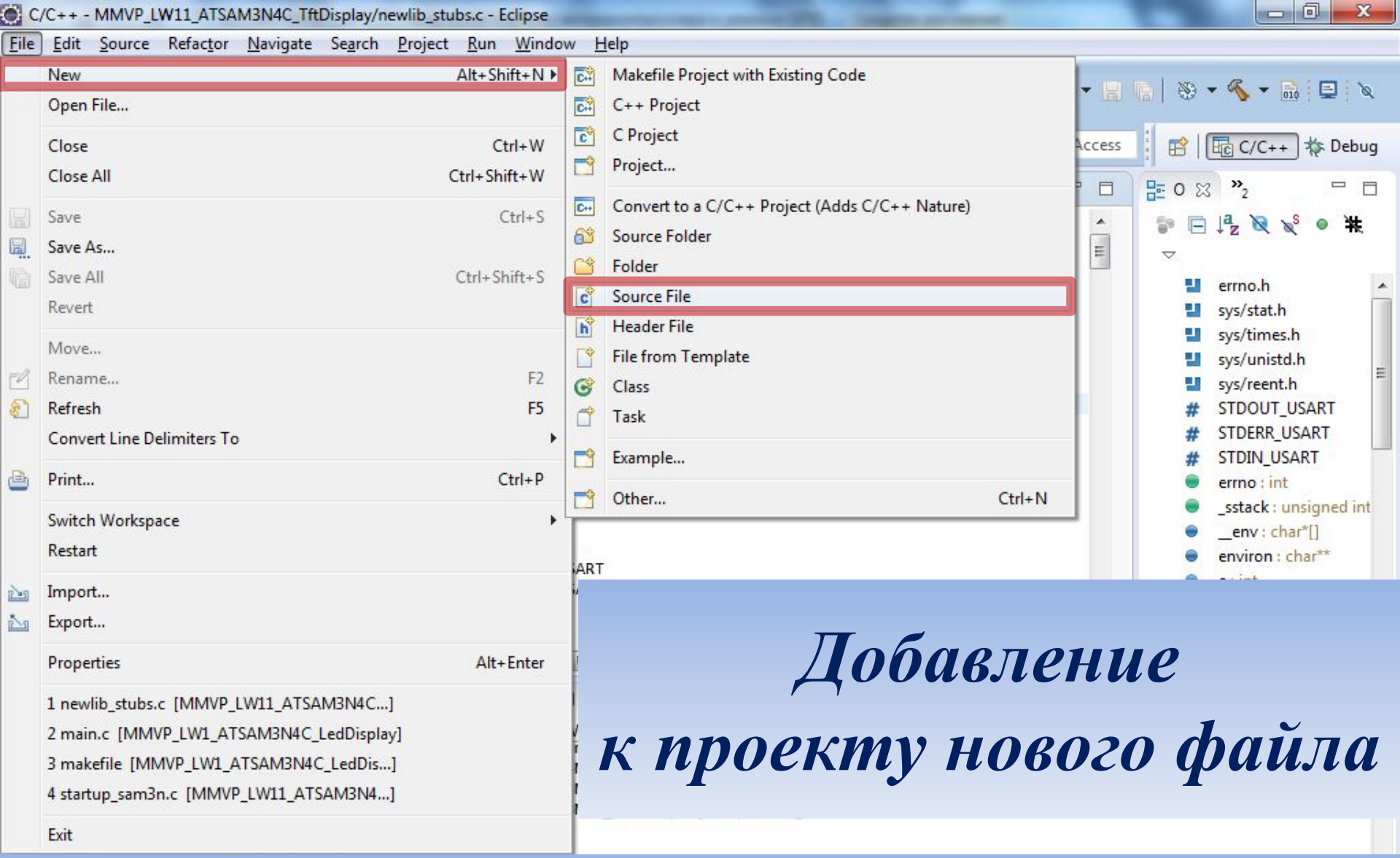
Генерирование файла image.c с помощью утилиты Image2Lcd v3.2



Генерирование файла image.c с помощью утилиты Image2Lcd v3.2



КАФЕДРА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫХ СИСТЕМ



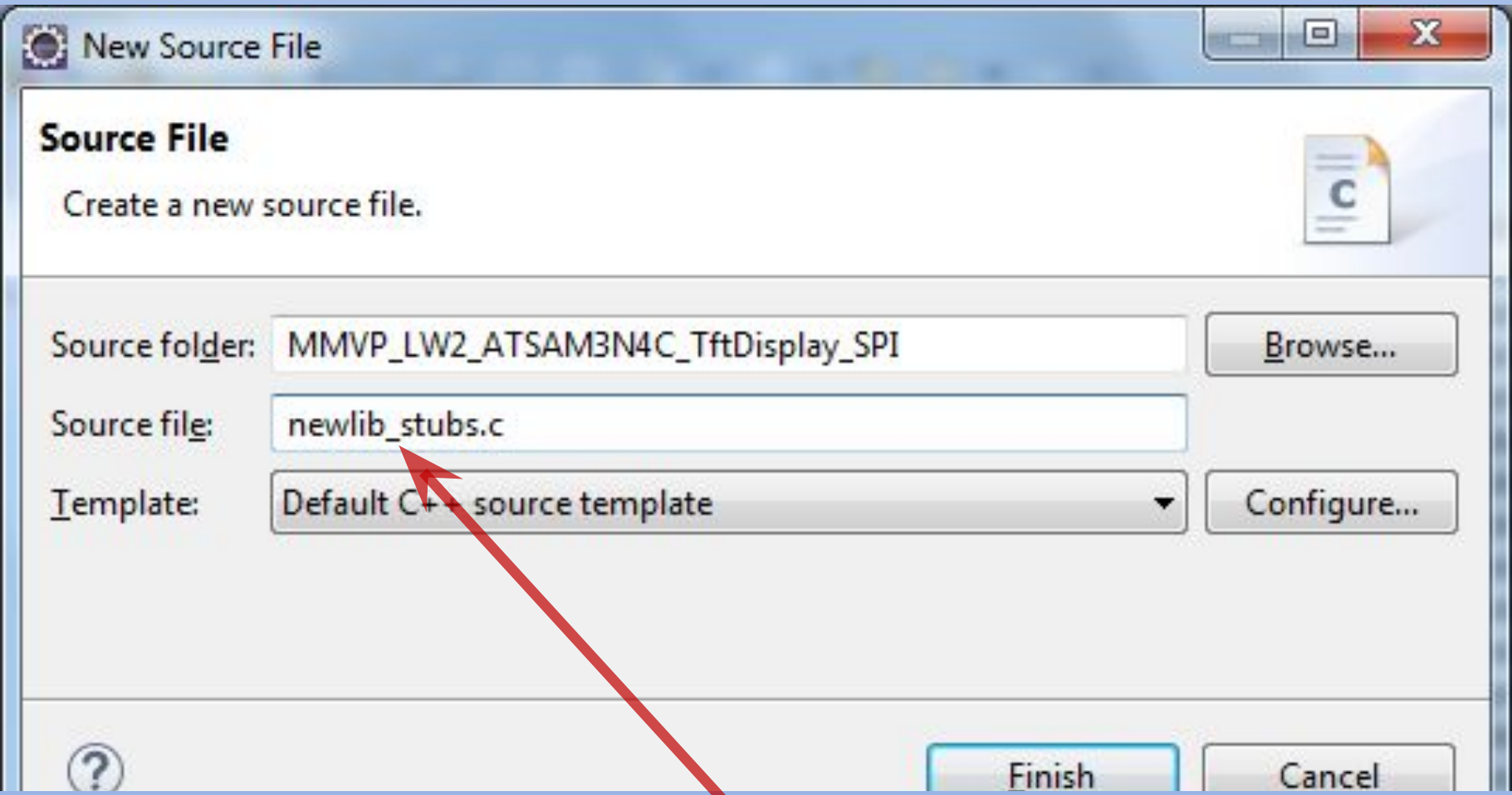
Добавление к проекту нового файла

Шаг 1. <u>Один из вариантов</u> начальной стадии добавления в проект нового файла

Главное меню «File → New → Source File»

Программа управления TFT-дисплеем посредством SPI

КАФЕДРА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫХ СИСТЕМ



Шаг 2. Завершающая стадия добавления в проект нового файла

Source file: имя_файла.c

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
/*
 * newlib_stubs.c
 *
 *  Created on: 01.04.2013
 *      Author: Администратор
 */
#include <errno.h>
#include <sys/stat.h>
#include <sys/times.h>
#include <sys/unistd.h>
#include <sys/reent.h>
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
#ifndef STDOUT_USART
#define STDOUT_USART    2
#endif

#ifndef STDERR_USART
#define STDERR_USART    2
#endif

#ifndef STDIN_USART
#define STDIN_USART     2
#endif
```

Программа управления TFT-дисплеем посредством SPI

Файл **newlib_stubs.c**

```
#undef errno
extern int errno;
extern unsigned int _sstack;

/*
 environ
 A pointer to a list of environment variables and their values.
 For a minimal environment, this empty list is adequate:
 */
char *__env[1] = { 0 };
char **environ = __env;
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
int e = 0;
int* __errno() {
    return &e;
}
int _write ( int file, char *ptr, int len);

void _exit ( int status) {
   _write(1, "exit", 4);
   while (1) {
       ;
   }
}
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
int _close ( int file) {
    return -1;
}
/*
 execve
 Transfer control to a new process. Minimal implementation
(for a system without processes):
 */
int _execve ( char *name, char **argv, char **env) {
    errno = ENOMEM;
    return -1;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 fork
 Create a new process. Minimal implementation
(for a system without processes):
 */

int _fork() {
    errno = EAGAIN;
    return -1;
}
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
/*
 fstat
 Status of an open file. For consistency with other minimal
implementations in these examples,
 all files are regarded as character special devices.
 The `sys/stat.h' header file required is distributed in the
`include' subdirectory for this C library.
 */
int _fstat ( int file, struct stat *st) {
    st->st_mode = S_IFCHR;
    return 0;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 getpid
 Process-ID; this is sometimes used to generate strings
unlikely to conflict with other processes. Minimal
implementation, for a system without processes:
 */

int _getpid() {
    return 1;
}
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
/*
 isatty
 Query whether output stream is a terminal. For consistency with the
other minimal implementations,
 */
int _isatty ( int file) {
    switch (file){
    case STDOUT_FILENO:
    case STDERR_FILENO:
    case STDIN_FILENO:
        return 1;
    default:
        errno = EBADF;
        return 0;          }}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 kill
 Send a signal. Minimal implementation:
 */
int _kill ( int pid, int sig) {
    errno = EINVAL;
    return (-1);
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 link
 Establish a new name for an existing file. Minimal
implementation:
 */

int _link ( char *old, char *new) {
    errno = EMLINK;
    return -1;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 lseek
 Set position in a file. Minimal implementation:
 */
int _lseek ( int file, int ptr, int dir) {
    return 0;
}
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
/*
 sbrk
 Increase program data space.
 Malloc and related functions depend on this
 */
caddr_t _sbrk ( int incr) {
    extern char _ebss; // Defined by the linker
    static char *heap_end;
    char *prev_heap_end;

    if (heap_end == 0) {
        heap_end = &_ebss;
    }
    prev_heap_end = heap_end;
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
char * stack = (char *) &_sstack;
    if (heap_end + incr >  stack)
    {
        _write (STDERR_FILENO, "Heap and stack
collision\n", 25);
        errno = ENOMEM;
        return  (caddr_t) -1;
    }

   heap_end += incr;
   return (caddr_t) prev_heap_end;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 read
 Read a character to a file. `libc' subroutines will use this system routine for input from all files, including stdin
 Returns -1 on error or blocks until the number of characters have been read.
 */
int _read ( int file, char *ptr, int len) {
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
/*
 stat
 Status of a file (by name). Minimal implementation:
 int    _EXFUN(stat,( const char *__path, struct stat
*__sbuf ));
 */
int _stat ( const char *filepath, struct stat *st) {
    st->st_mode = S_IFCHR;
    return 0;
}
```

Программа управления TFT-дисплеем посредством SPI

Файл newlib_stubs.c

```
/*
 times
 Timing information for current process.
Minimal implementation:
 */

clock_t _times ( struct tms *buf) {
    return -1;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 unlink
 Remove a file's directory entry. Minimal implementation:
 */
int _unlink ( char *name) {
    errno = ENOENT;
    return -1;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 wait
 Wait for a child process. Minimal
implementation:
 */
int _wait ( int *status) {
    errno = ECHILD;
    return -1;
}
```

Программа управления TFT-дисплеем посредством SPI

*Файл **newlib_stubs.c***

```
/*
 write
 Write a character to a file. `libc' subroutines will use this system routine for output to all files, including stdout
 Returns -1 on error or number of bytes sent
 */
int _write ( int file, char *ptr, int len) {
    return 1;
}
```

Программа управления TFT-дисплеем посредством SPI

Файл makefile

Отличительные строки make-файла

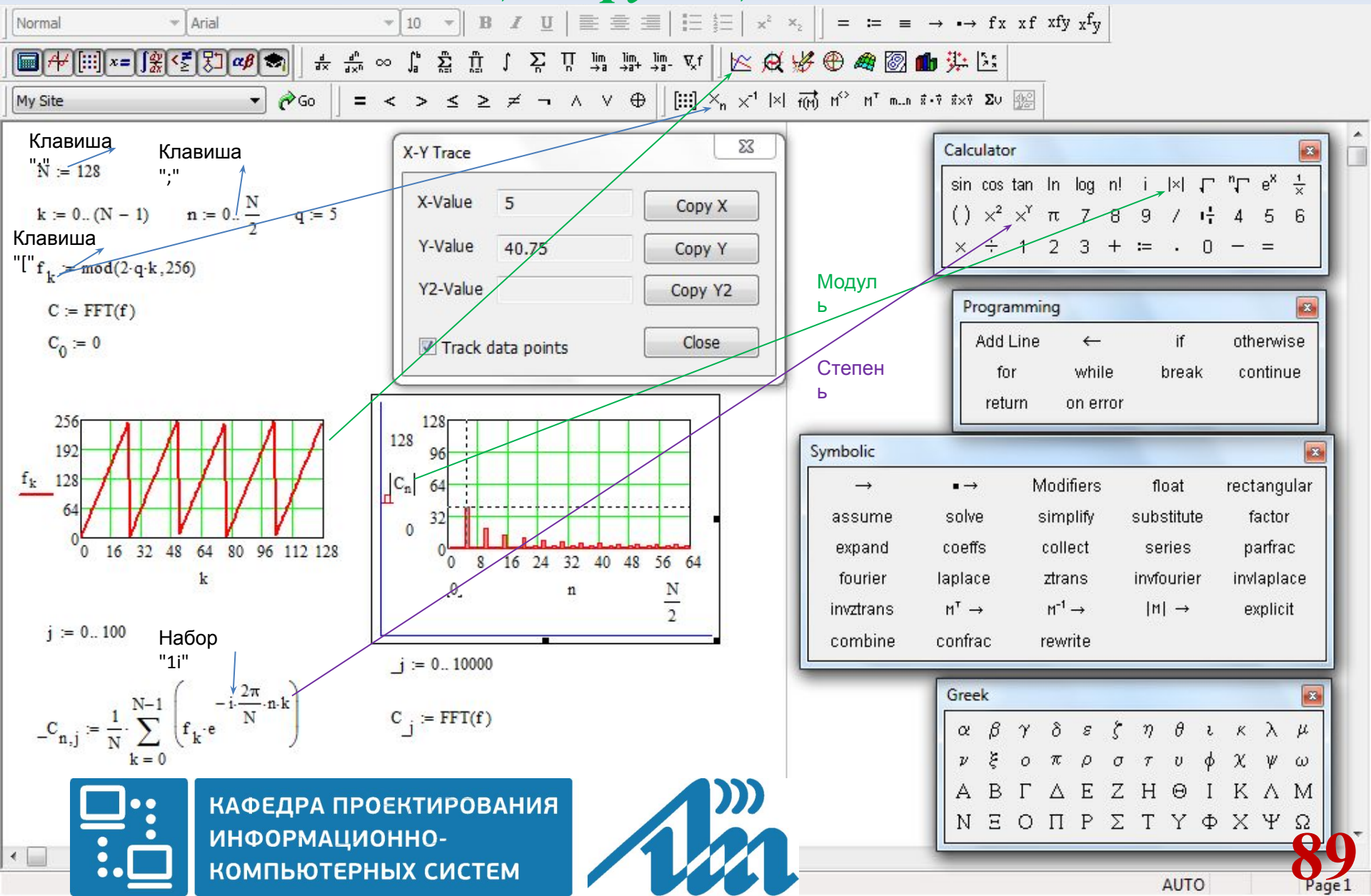
```
# Target file name (without extension)
TARGET=MMVP_LW2_ATSAM3N4C_TftDisplay_SPI
.....................................
SOURCE=startup_sam3n.c \
    main.c \
    newlib_stubs.c \
    image.c \
    $(CM3_SPL_DIR)/src/pmc.c \
    $(CM3_SPL_DIR)/src/pio.c \
    $(CM3_SPL_DIR)/src/spi.c
```

КАФЕДРА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫХ СИСТЕМ

Коррекция проекта завершена

Удачного его построения и проверки работы!

Сравнительный анализ ДПФ и БПФ с помощью функций MathCAD



КАФЕДРА ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫХ СИСТЕМ

Постановка задачи сравнительного анализа ДПФ и БПФ

Путём варьирования верхних пределов индексов *j* (например, 10000) и *_j* (например, 1000000) с помощью секундомера оценить соотношение времён выполнения ДПФ «в лоб» и БПФ функцией FFT()