附页1: **main.c** & **lcd.c**

main.c

```
void lcd_test();

int main(void)

{

  AF();

  return 0;

}
```

lcd.c

```
#include "ascii.h"

#include "pic.h"

typedef unsigned int u32;

typedef unsigned short u16;

#define GPF0CON (*(volatile unsigned long *)0xE0200120)

#define GPF1CON (*(volatile unsigned long *)0xE0200140)

#define GPF2CON (*(volatile unsigned long *)0xE0200160)

#define GPF3CON (*(volatile unsigned long *)0xE0200180)

#define GPD0CON (*(volatile unsigned long *)0xE02000A0)

#define GPD0DAT (*(volatile unsigned long *)0xE02000A4)

#define CLK_SRC1 (*(volatile unsigned long *)0xe0100204)

#define CLK_DIV1 (*(volatile unsigned long *)0xe0100304)

#define DISPLAY_CONTROL (*(volatile unsigned long *)0xe0107008)

#define VIDCON0 (*(volatile unsigned long *)0xF8000000)

#define VIDCON1 (*(volatile unsigned long *)0xF8000004)

#define VIDTCON2 (*(volatile unsigned long *)0xF8000018)

#define WINCON0 (*(volatile unsigned long *)0xF8000020)

#define WINCON2 (*(volatile unsigned long *)0xF8000028)

#define SHADOWCON (*(volatile unsigned long *)0xF8000034)

#define VIDOSD0A (*(volatile unsigned long *)0xF8000040)

#define VIDOSD0B (*(volatile unsigned long *)0xF8000044)

#define VIDOSD0C (*(volatile unsigned long *)0xF8000048)

#define VIDW00ADD0B0 (*(volatile unsigned long *)0xF80000A0)
```

```c
#define VIDW00ADD1B0 (*(volatile unsigned long *)0xF80000D0)

#define VIDTCON0 (*(volatile unsigned long *)0xF8000010)

#define VIDTCON1 (*(volatile unsigned long *)0xF8000014)


#define HSPW (40)   // 1~40

#define HBPD (10 - 1) // 46

#define HFPD (240 - 1) // 16 210 354

#define VSPW (20)   // 1~20

#define VBPD (10 - 1) // 23

#define VFPD (30 - 1) // 7 22 147


// FB地址

#define FB_ADDR (0x23000000)

#define ROW (600)

#define COL (1024)

#define HOZVAL (COL - 1)

#define LINEVAL (ROW - 1)


#define XSIZE COL

#define YSIZE ROW


u32 *pfb = (u32 *)FB_ADDR;


// 常用颜色定义

#define BLUE 0x0000FF

#define RED 0xFF0000

#define GREEN 0x00FF00

#define WHITE 0xFFFFFF

#define BLACK 0x000000


// 定义操作寄存器的宏

#define GPH0CON 0xE0200C00

#define GPH0DAT 0xE0200C04
```

```c
#define GPH2CON 0xE0200C40

#define GPH2DAT 0xE0200C44


#define rGPH0CON (*(volatile unsigned int *)GPH0CON)

#define rGPH0DAT (*(volatile unsigned int *)GPH0DAT)

#define rGPH2CON (*(volatile unsigned int *)GPH2CON)

#define rGPH2DAT (*(volatile unsigned int *)GPH2DAT)


// 初始化按键

void key_init(void)

{

  rGPH2CON &= ~(0xFFFF << 0);

  rGPH0CON &= ~(0xFF << 8);

}


void delay20ms(void)

{

  int i, j;

  for (i = 0; i < 100; i++)

  {

    for (j = 0; j < 1000; j++)

    {

      i *j;

    }

  }

}


static void delay(void)

{

  volatile u32 i, j;

  for (i = 0; i < 4000; i++)

    for (j = 0; j < 1000; j++)

      ;

}
```

```
void lcd_init(void)
{
  // 配置引脚用于LCD功能
  GPF0CON = 0x22222222;
  GPF1CON = 0x22222222;
  GPF2CON = 0x22222222;
  GPF3CON = 0x22222222;

  // 打开背光 GPD0_0 （PWMTOUT0）
  GPD0CON &= ~(0xf << 0);
  GPD0CON |= (1 << 0); // output mode
  GPD0DAT &= ~(1 << 0); // output 0 to enable backlight

  // 10: RGB=FIMD I80=FIMD ITU=FIMD
  DISPLAY_CONTROL = 2 << 0;

  // bit[26~28]:使用RGB接口
  // bit[18]:RGB 并行
  // bit[2]:选择时钟源为HCLK_DSYS=166MHz
  VIDCON0 &= ~((3 << 26) | (1 << 18) | (1 << 2));

  // bit[1]:使能lcd控制器
  // bit[0]:当前帧结束后使能lcd控制器
  VIDCON0 |= ((1 << 0) | (1 << 1));

  // bit[6]:选择需要分频
  // bit[6~13]:分频系数为5，即VCLK = 166M/(4+1) = 33M
  VIDCON0 |= 4 << 6 | 1 << 4;

  // H43-HSD043I9W1.pdf(p13) 时序图：VSYNC和HSYNC都是低脉冲
  // s5pv210芯片手册(p1207) 时序图：VSYNC和HSYNC都是高脉冲有效，所以需要反转
  VIDCON1 |= 1 << 5 | 1 << 6;
  // 设置时序
  VIDTCON0 = VBPD << 16 | VFPD << 8 | VSPW << 0;
  VIDTCON1 = HBPD << 16 | HFPD << 8 | HSPW << 0;
  // 设置长宽(物理屏幕)
  VIDTCON2 = (LINEVAL << 11) | (HOZVAL << 0);

  // 设置window0
  // bit[0]:使能
  // bit[2~5]:24bpp （RGB888）
  WINCON0 |= 1 << 0;
  WINCON0 &= ~(0xf << 2);
  WINCON0 |= (0xB << 2) | (1 << 15);

#define LeftTopX 0
#define LeftTopY 0
#define RightBotX 1023
#define RightBotY 599

  // 设置window0的上下左右
  // 设置的是显存空间的大小
  VIDOSD0A = (LeftTopX << 11) | (LeftTopY << 0);
  VIDOSD0B = (RightBotX << 11) | (RightBotY << 0);
  VIDOSD0C = (LINEVAL + 1) * (HOZVAL + 1);

  // 设置fb的地址
  VIDW00ADD0B0 = FB_ADDR;
  VIDW00ADD1B0 = (((HOZVAL + 1) * 4 + 0) * (LINEVAL + 1)) & (0xffffff);

  // 使能channel 0传输数据
  SHADOWCON = 0x1;
}

void lcd_draw_pixel(int x, int y, int color)
{
  unsigned long *pixel = (unsigned long *)FB_ADDR;
  *(pixel + y * COL + x) = color;
  return;
}

static void lcd_draw_background(u32 color)
{
  u32 i, j;
  for (j = 0; j < ROW; j++)
  {
    for (i = 0; i < COL; i++)
    {
      lcd_draw_pixel(i, j, color);
    }
  }
}

void lcd_draw_pictures(unsigned int px, unsigned int py, unsigned int pc, unsigned int pr, const
unsigned char *pData)
{
  u32 x, y, color, p = 0, pco = 0;
  for (x = px; x < px + pc; x++)
  {
    for (y = py; y < py + pr; y++)
    {
      lcd_draw_pixel(x, y, (pData[p] & (2 ^ pco)) ? 0x0000C0 : WHITE);
      if (pco > 7)
      {
        pco = 0;
        p++;
      }
      pco++;
    }
  }
}
```

```
void lcd_draw_picture(const unsigned char *pData)
{
  u32 x, y, color, p = 0, pco = 0;
  for (x = 0; x < COL; x++)
  {
    for (y = 0; y < ROW; y++)
    {
      lcd_draw_pixel(x, y, (pData[p] & (2 ^ pco)) ? 0x0000C0 : WHITE);
      if (pco > 7)
      {
        pco = 0;
        p++;
      }
      pco++;
    }
  }
}

static void show_8_16(unsigned int x, unsigned int y, unsigned int color, unsigned char *data)
{
  // count记录当前正在绘制的像素的次序
  int i, j, count = 0;
  for (j = y; j < (y + 16); j++)
  {
    for (i = x; i < (x + 8); i++)
    {
      if (i < XSIZE && j < YSIZE)
      {
        if (data[count / 8] & (1 << (count % 8)))
          lcd_draw_pixel(i, j, color);
      }
      count++;
    }
  }
}

void draw_ascii_ok8(unsigned int x, unsigned int y, unsigned int color, unsigned char *str)
{
  int i;
  unsigned char *ch;
  for (i = 0; str[i] != '\0'; i++)
  {
    ch = (unsigned char )ascii_8_16[(unsigned char)str*[i] - 0x20];
    show_8_16(x, y, color, ch);
    x += 10;
    if (x >= XSIZE)
    {
      x -= XSIZE; // 回车
      y += 16;  // 换行
    }
  }
}

static void show_16_16(unsigned int x, unsigned int y, unsigned int color, unsigned char *data)
{
  int i, j, count = 0;

  for (j = y; j < (y + 16); j++)
  {
    for (i = x; i < (x + 16); i++)
    {
      if (i < XSIZE && j < YSIZE)
      {
        if (data[count / 8] & (1 << (count % 8)))
          lcd_draw_pixel(i, j, color);
      }
      count++;
    }
  }
}

void draw_ascii_ok16(unsigned int x, unsigned int y, unsigned int color, unsigned char *str)
{
  int i;
  unsigned char *ch;
  for (i = 0; str[i] != '\0'; i++)
  {
    ch = (unsigned char )ascii_16_16[((unsigned char)str*[i] - 97) * 2];
    show_16_16(x, y, color, ch);

    x += 20;
    if (x >= XSIZE)
    {
      x -= XSIZE; // 回车
      y += 16;  // 换行
    }
  }
}

static void show_32_32(unsigned int x, unsigned int y, unsigned int color, unsigned char *data)
{
  int i, j, count = 0;

  for (j = y; j < (y + 32); j++)
  {
    for (i = x; i < (x + 32); i++)
    {
      if (i < XSIZE && j < YSIZE)
      {
        if (data[count / 8] & (1 << (count % 8)))
          lcd_draw_pixel(i, j, color);
      }
      count++;
    }
  }
}

void draw_ascii_ok32(unsigned int x, unsigned int y, unsigned int color, unsigned char *str)
{
  int i;
  unsigned char *ch;
  for (i = 0; str[i] != '\0'; i++)
  {
    ch = (unsigned char )jmu_32_32[((unsigned char)str*[i] - 97) * 8];
    show_32_32(x, y, color, ch);
    x += 40;
    if (x >= XSIZE)
    {
      x -= XSIZE; // 回车
      y += 32;  // 换行
    }
  }
}

void draw_circular(unsigned int centerX, unsigned int centerY, unsigned int radius, unsigned int
color)
{
  int x, y;
  for (y = 0; y < XSIZE; y++)
  {
    for (x = 0; x < YSIZE; x++)
    {
      if (((y - centerY) * (y - centerY) + (x - centerX) * (x - centerX)) <= radius * radius)
        lcd_draw_pixel(x, y, color);
    }
  }
```

```
}


void draw_rectangle(unsigned int x1, unsigned int y1, unsigned int x2, unsigned int y2, unsigned
int color)

{

 int x, y, temp;

 if (x1 > x2)

 {

  temp = x2;

  x2 = x1;

  x1 = temp;

  temp = y2;

  y2 = y1;

  y1 = temp;

 }


 if (y1 < y2)

 {

  for (x = x1; x <= x2; x++)

  {

   for (y = y1; y <= y2; y++)


   {

    lcd_draw_pixel(x, y, color);

   }

  }

 }

 else

 {

  for (x = x1; x <= x2; x++)

  {

   for (y = y1; y >= y2; y--)

   {

    lcd_draw_pixel(x, y, color);

   }
```

```
    }
  }
}

void draw_triangle(int x1, int y1, int x2, int y2, int x3, int y3, unsigned int color)
{
  int x, y;
  for (x = 0; x <= COL; x++)
  {
    for (y = 0; y <= ROW; y++)
    {
      if (((x2 - x1) * (y - y1) - (y2 - y1) * (x - x1)) + ((x3 - x2) * (y - y2) - (y3 - y2) * (x - x2)) + ((x3 - x1) * (y -
y1) - (y3 - y1) * (x - x1)) <= (x2 - x1) * (y3 - y1) - (y2 - y1) * (x3 - x1))
        lcd_draw_pixel(x, y, color);
    }
  }
}

void draw_line(unsigned int x1, unsigned int y1, unsigned int x2, unsigned y2, unsigned int color)
{
  int x, y;
  int temp;
  if (x1 > x2)
  {
    temp = x1;
    x1 = x2;
    x2 = temp;
    temp = y1;
    y1 = y2;
    y2 = temp;
  }

  if (y1 < y2)
  {
    for (x = x1; x <= x2; x++)
    {
      for (y = y1; y <= y2; y++)
      {
        if ((x1 - x) * (y2 - y) == (y1 - y) * (x2 - x))
        {
          lcd_draw_pixel(x, y, color);
        }
      }
    }
  }
  else
  {
    for (x = x1; x <= x2; x++)
    {
      for (y = y1; y >= y2; y--)
      {
        if ((x1 - x) * (y2 - y) == (y1 - y) * (x2 - x))
        {
          lcd_draw_pixel(x, y, color);
        }
      }
    }
  }
}

void glib_line(unsigned int x1, unsigned int y1, unsigned int x2, unsigned int y2, unsigned int color)
{
  int dx, dy, e;
  dx = x2 - x1;
  dy = y2 - y1;
  if (dx >= 0)
  {
    if (dy >= 0) // dy>=0
    {
```

```
  if (dx >= dy) // 1/8 octant
  {
    e = dy - dx / 2;
    while (x1 <= x2)
    {
      lcd_draw_pixel(x1, y1, color);
      if (e > 0)
      {
        y1 += 1;
        e -= dx;
      }
      x1 += 1;
      e += dy;
    }
  }
  else // 2/8 octant
  {
    e = dx - dy / 2;
    while (y1 <= y2)
    {
      lcd_draw_pixel(x1, y1, color);
      if (e > 0)
      {
        x1 += 1;
        e -= dy;
      }
      y1 += 1;
      e += dx;
    }
  }
}
else // dy<0
{
  dy = -dy; // dy=abs(dy)
```

```
    if (dx >= dy) // 8/8 octant
    {
      e = dy - dx / 2;
      while (x1 <= x2)
      {
        lcd_draw_pixel(x1, y1, color);
        if (e > 0)
        {
          y1 -= 1;
          e -= dx;
        }
        x1 += 1;
        e += dy;
      }
    }
    else // 7/8 octant
    {
      e = dx - dy / 2;
      while (y1 >= y2)
      {
        lcd_draw_pixel(x1, y1, color);
        if (e > 0)
        {
          x1 += 1;
          e -= dy;
        }
        y1 -= 1;
        e += dx;
      }
    }
  }
}
else //dx<0
```

```
{
  dx = -dx;  //dx=abs(dx)
  if (dy >= 0) // dy>=0
  {
    if (dx >= dy) // 4/8 octant
    {
      e = dy - dx / 2;
      while (x1 >= x2)
      {
        lcd_draw_pixel(x1, y1, color);
        if (e > 0)
        {
          y1 += 1;
          e -= dx;
        }
        x1 -= 1;
        e += dy;
      }
    }
    else // 3/8 octant
    {
      e = dx - dy / 2;
      while (y1 <= y2)
      {
        lcd_draw_pixel(x1, y1, color);
        if (e > 0)
        {
          x1 -= 1;
          e -= dy;
        }
        y1 += 1;
        e += dx;
      }
    }
```

```
}
else // dy<0
{
  dy = -dy; // dy=abs(dy)

  if (dx >= dy) // 5/8 octant
  {
    e = dy - dx / 2;
    while (x1 >= x2)
    {
      lcd_draw_pixel(x1, y1, color);
      if (e > 0)
      {
        y1 -= 1;
        e -= dx;
      }
      x1 -= 1;
      e += dy;
    }
  }
  else // 6/8 octant
  {
    e = dx - dy / 2;
    while (y1 >= y2)
    {
      lcd_draw_pixel(x1, y1, color);
      if (e > 0)
      {
        x1 -= 1;
        e -= dy;
      }
      y1 -= 1;
      e += dx;
    }
```

```
      }
    }
  }
}

void AF(void)
{
  int menucase = 0;
  lcd_init();
  lcd_draw_background(0x000000);
  int pic_case = 0;
  while (1)
  {
    if (!(rGPH0DAT & (1 << 2))) //left合照
    {
      delay20ms();
      if (!(rGPH0DAT & (1 << 2)))
      {
        if (menucase == 1)
        {
          lcd_draw_background(WHITE);
          lcd_draw_pictures(288, 164, 448, 272, gImage_pic);
          pic_case = 1;
        }
      }
    }
    else if (!(rGPH2DAT & (1 << 1))) //right 图形
    {
      delay20ms();
      if (!(rGPH2DAT & (1 << 1)))
      {
        if (menucase == 1)
        {
          lcd_draw_background(WHITE);
      delay();
      draw_rectangle(700, 200, 1000, 400, BLUE);
      draw_line(400, 400, 500, 200, BLUE);
      draw_line(600, 400, 500, 200, BLUE);
      draw_line(600, 400, 400, 400, BLUE);
      glib_line(0, 245, 260, 245, BLUE);
      glib_line(130, 150, 210, 390, BLUE);
      glib_line(50, 390, 260, 245, BLUE);
      glib_line(0, 245, 210, 390, BLUE);
      glib_line(50, 390, 130, 150, BLUE);
      draw_circular(399, 239, 50, BLUE);
    }
    menucase = 0;
  }
}
else if (!(rGPH0DAT & (1 << 3))) //down 文字
{
  delay20ms();
  if (!(rGPH0DAT & (1 << 3)))
  {
    if (menucase == 1)
    {
      if (pic_case != 1)
      {
        lcd_draw_background(WHITE);
        pic_case = 0;
      }
      draw_ascii_ok32(352, 150, RED, "abcdefgh"); //jmu，cycu
      draw_ascii_ok16(288, 450, BLUE, "ijk");   //FY
      draw_ascii_ok8(288, 480, BLUE, "Chen Feiyuan");
      draw_ascii_ok8(288, 500, BLUE, "201741053072");
      draw_ascii_ok16(560, 450, 0XF19EC2, "lmn"); //JM
      draw_ascii_ok8(560, 480, BLUE, "Lin Junming");
      draw_ascii_ok8(560, 500, BLUE, "201741053057");
      draw_ascii_ok16(420, 450, BLUE, "opq"); //ZH
      draw_ascii_ok8(420, 480, BLUE, "Wu Zhenhang");
      draw_ascii_ok8(420, 500, BLUE, "201741053075");
    }
    menucase == 1;
  }
}
else if (!(rGPH2DAT & (1 << 0))) //up屏幕刷新
{
  delay20ms();
  if (!(rGPH2DAT & (1 << 0)))
  {
    if (menucase == 1)
    {
      lcd_draw_background(RED);
      delay();
      lcd_draw_background(GREEN);
      delay();
      lcd_draw_background(BLUE);
      delay();
      lcd_draw_background(BLACK);
      delay();
      lcd_draw_background(WHITE);
      delay();
      lcd_draw_background(0XC0C0C0);
      delay();
      lcd_draw_background(0X808080);
      delay();
      lcd_draw_background(0X404040);
      delay();
      lcd_draw_background(BLACK);
    }
    menucase = 0;
  }
```

```
    }
    else if (!(rGPH2DAT & (1 << 3))) //menu 菜单
    {
      delay20ms();
      if (!(rGPH2DAT & (1 << 3)))
      {
        menucase = 1;
        lcd_draw_background(WHITE);
        draw_ascii_ok8(427, 250, BLUE, "MENU");
        draw_ascii_ok8(427, 275, BLUE, "Screen Fresh <-up");
        draw_ascii_ok8(427, 300, BLUE, "Text      <-down");
        draw_ascii_ok8(427, 325, BLUE, "Group Photo <-left");
        draw_ascii_ok8(427, 350, BLUE, "Graphics   <-right");
      }
    }
    else if (!(rGPH2DAT & (1 << 2))) //back
    {
      delay20ms();
      if (!(rGPH2DAT & (1 << 2)))
      {
        lcd_draw_background(0x000000);
        menucase = 0;
      }
    }
  }
}
```