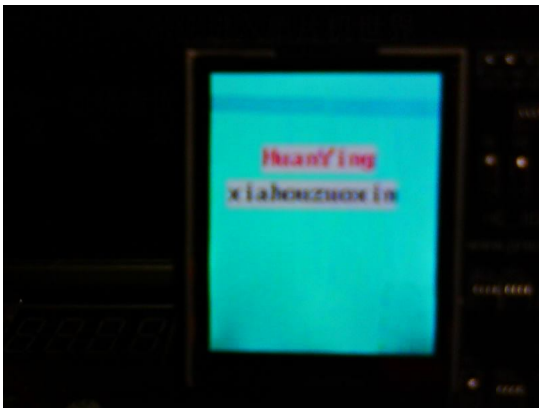
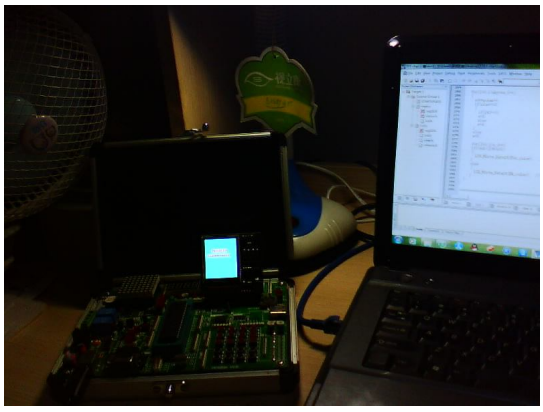
2011/7/16　　星期六　　晚上有雨

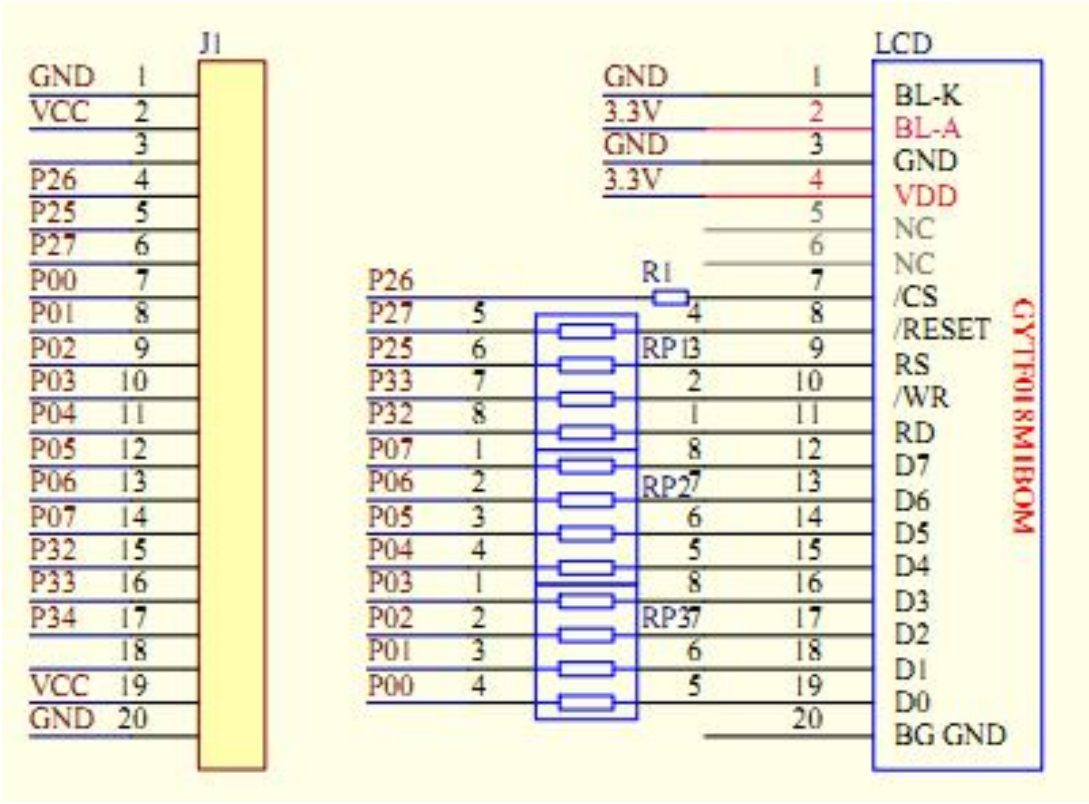
今天闲来无事，整了会儿 TFT 液晶显示，呵呵，终于能显示字符串了，下图是显示效果。





我使用的 TFT 驱动是 C1L1-05，TFT 显示的基本程序包括 TFT 与单片机接口的通信处理，TFT 液晶初始化以及字符编码。

1. 接口与通信原理



接口	LCD_CS	LCD_RST	LCD_RS	LCD_WR	LCD_RD
单片机管脚	P2^6	P2^7	P2^5	P3^3	P3^2
功能	片选	复位	命令/数据	写使能	读使能

写入命令时顺序：LCD_CS=0; LCD_RS=0; LCD_WR=0;写 8 位数据; LCD_WR=1; LCD_CS=0;

写入数据时顺序：LCD_CS=0; LCD_RS=1; LCD_WR=0;写 8 位命令; LCD_WR=1; LCD_CS=0;

```
/*********************************************************************************
                              写入单字节函数
*********************************************************************************/
void Write_LCD(uchar Type, uchar Value)
```

```
{
    LCD_CS = 0;
    LCD_RS = Type;
    LCD_WR = 0;
    LCDDATA = Value;
    LCD_WR = 1;
    LCD_CS = 1;
}
/*********************************************************************************
                              写 16 位数据，主要是写入颜色
*********************************************************************************/
void Write_ColorData(uint Value)
{
    LCD_CS = 0;
    LCD_RS = 1;
    LCD_WR = 0;
    LCDDATA = (uchar)Value;
    LCD_WR = 1;
    LCD_WR = 0;
    LCDDATA = (Value>>8) & 0xFF;
    LCD_WR = 1;
    LCD_CS = 1;
}
/*写寄存器*/
void Write_LCD_Regster(uchar reg, uchar value)
{
    Write_LCD(COMMAND, reg);
    Write_LCD(DATA, value);
}
```

2、初始化 TFT，按下述步骤操作即可

```
Write register MODE_SEL1(0x01h)=Resolution_Select
Write register MODE_SEL2(0x02h)=0x12h
Write register MODE_SEL3(0x03h)=DATA_Input_Select1   /*for CPU I/F*/
Write register MODE_SEL4(0x04h)=DATA_Input_Select2   /*for SPI I/F*/
Write register VCO_Mode(0x05h)=VCO_Select
Write register VCOMH_CTRL(0x07h)=0x7Fh
Write register VCOML_CTRL(0x08h)=0x17h
Write register PWS_X(0x09h)=0x00h
Write register PWS_Y(0x10h)=0x00h
Write register PWE_X(0x11h)=PWE_X_Select
Write register PWE_Y(0x12h)=PWE_Y_Select
Write register SRAM_POSITION_X(0x18h)= Start_X_Select
Write register SRAM_POSITION_Y(0x19h)= Start_Y_Select
Write register SRAM_Control(0x17h)= Scan_Direction_Select

integer x,y
for y = 0 to PWE_Y_Select
  for x = 0 to PWE_X_Select
    Write display data to SRAM
Write register DAC_OP_CTRL(0x06h)=0xC7h
```

```
/*******************************************************************************
                                设置显示窗口子函数
*******************************************************************************/
void LCD_SetRamAddr(uint xStart, uint xEnd, uint yStart, uint yEnd)
{
    uint VerPos,HorPos,StartAddr;

    HorPos    = (uint)(xStart | (xEnd<<8));
    VerPos    = (uint)(yStart | (yEnd<<8));
    StartAddr = (uint)(xStart | (yStart<<8));

    Write_LCD_Regster(0x09, xStart);
    Write_LCD_Regster(0x10,yStart);
    Write_LCD_Regster(0x11,xEnd);
    Write_LCD_Regster(0x12,yEnd);

    Write_LCD_Regster(0x18, xStart);
   Write_LCD_Regster(0x19, yStart);
    Write_LCD(COMMAND,0x22);          // 0x22
}
/*******************************************************************************
                                    LCD 初始化
*******************************************************************************/
void LCD_Init(void)
{
    uint num;

   Write_LCD_Regster(0x0001,0x0002);  //MODE_SEL1
   Write_LCD_Regster(0x0002,0x0012);   //MODE_SEL2
   Write_LCD_Regster(0x0003,0x0000);   //MODE_SEL3
   Write_LCD_Regster(0x0004,0x0010);  //MODE_SEL4
   Write_LCD_Regster(0x0005,0x0008);   //VCO_MODE
   Write_LCD_Regster(0x0007,0x007f);  //VCOMHT_CTRL
   Write_LCD_Regster(0x0008,0x0017);   //VCOMLT_CTRL
   Write_LCD_Regster(0x0009,0x0000);   //write SRAM window start X point
   Write_LCD_Regster(0x0010,0x0000);   //write SRAM window start y point
   Write_LCD_Regster(0x0011,0x0083);   //write SRAM window end x point
   Write_LCD_Regster(0x0012,0x009f);   //write SRAM window end y point
   Write_LCD_Regster(0x0017,0x0000);   //SRAM contrl
   Write_LCD_Regster(0x0018,0x0000);   //SRAM x position
   Write_LCD_Regster(0x0019,0x0000);   //SRAM y position
   Write_LCD_Regster(0x0006,0x00c5);   //DAC_OP_CTRL2

   LCD_SetRamAddr(0,127, 0,159);
```

```
    for(num=20480;num>0;num--)   //128*160=20480，或参考步骤使用 2 次 for 循环
        Write_ColorData(0xffff);
    delay_ms(10); //延时
}}
/*******************************************************************************
                                清屏子函数,n 代表 Color[n]
*******************************************************************************/
void   LCD_clear(uchar n)
{
    uint num;

    LCD_SetRamAddr(0,127, 0,159);
    for(num=20480;num>0;num--)              //160*128=20480
    {
      Write_ColorData(colors[n]);
    }
}
```

3、字符编码

定义数组 const unsigned char code Font8x16[96][16]按 ASCII 码表进行 8x16 的编码，可以寻找编码软件编码，其中数组中 96 表示编码的 ASCII 符号个数（因为前 32 个字符不使用不进行编码），16 代表 8x16 中的 16 即每个字符由 16 个 8 位 2 进制序列表示。

4、字符显示

```
/*******************************************************************************
                            字符显示子函数，点阵显示原理
*******************************************************************************/
void LCD_ShowChar(uchar x,uchar y,uint For_color,uint Bk_color, char ch)
{
    uchar temp;
    uchar pos,t;
    uchar   CHAR_W,CHAR_H;

    CHAR_W = 8;             //8*16，字体宽 8，高 16
    CHAR_H = 16;

    if(x>(LCD_SIZE_X-CHAR_W)||y>(LCD_SIZE_Y-CHAR_H))   //超出
    return;

    LCD_SetRamAddr(x, x+CHAR_W-1, y,y+CHAR_H-1);
    ch = ch-32;                                 //按照 ASCII 编码顺序的到相应字母的编码

    for(pos=0;pos<CHAR_H;pos++)
    {
        temp= Font8x16[ch][pos];
        for(t=0;t<CHAR_W;t++)
```

```
        {
          if(temp&0x80)
          Write_ColorData(For_color);    //textcolor
          else
          Write_ColorData(Bk_color);    //backcolor
           temp<<=1;
        }
    }
}
```

最后，在主函数 for 循环中调用 LCD_ShowChar 就可以显示字符串了。嘿嘿，大功告成喽……借此可以与其他资源比如键盘，红外，DS18B20 等结合起来，实现好玩的玩意儿，那些下次再整吧。以前写完程序后很少做整理，悔恨至极，现方觉将自己的成果整理出来亦不为一种快乐。