

得分：97

评语：介绍完整

串口屏显示

串口屏可作为输出设备（显示）以及输入设备（按键），开发难度小，操作简单，软件要求低且拥有专门的上位机辅助开发。

一、前期准备

1.USB转TTL 我们要准备一个USB TO TTL模块。作用是用来上位机（电脑）与串口屏的通讯。



二、串口屏上位机使用方法以及界面设计

1.

打开USART HMI 软件，点击新建

2.

从工具箱中拖需要使用的组件。比较常用就是文本、按钮，曲线等。去设计我们想要的界面例如背景色，控件的大小等等。

3.


点击软件左上方工具按钮，我们要提前导入字体也就是字库制作，我们选择我们想要的字体以及大小即可以生成字库。每生成一个字库并且添加到工程中都会有一个字体ID


4.

当我们使用文本或者数字控件时，我们需要不同的字体或大小时，我们可以再右下角属性一栏中修改font参数，修改为你需要的字体ID即可。如下图所示：

属性

t3(文本)



type	116
id	3
objname	t3
vscope	全局
sta	单色
style	平面
key	无
font	1
bco	 34815
pco	 0
xcen	居中
ycen	居中
pw	否
txt	0

点击属性显示相应注释

比如以下是我们在电赛制作的串口屏页面：



三、STM32编程

tm32软件部分总体分为发送数据和接收数据。 发送数据

既然我们要发送数据我们就要符合相应的通讯协议，在我们发送数据，总要有个结尾标志，如果没有的话单片机就不知道你的数据发没发送完，从而卡死出不来。所以第一个重点就是要帧头和帧尾，向串口屏发送数据完要加开始符和结束符。 以下是我的代码：

```
#define RINGBUFF_LEN 500
#define FRAMELENGTH 6

float kp, ki, kd;

int fputc(int c, FILE* stream)
{
    DL_UART_Main_transmitDataBlocking(UART_HMI_INST,c);
}
```

```
    return c;
}

int fputs(const char *restrict s, FILE* restrict stream)
{
    uint16_t i, len;
    len = strlen(s);
    for (i= 0; i<len;i++)
    {
        DL_UART_Main_transmitDataBlocking(UART_HMI_INST, s[i]);
    }
    return len;
}

int puts(const char*_ptr)
{
    int count = fputs(_ptr, stdout);
    count += fputs("\n", stdout);
    return count;
}

void HMI_send_string(char* name, char* showdata)
{
    printf("%s=\"%s\"\\xff\\xff\\xff", name, showdata);
}

void HMI_send_number(char* name, int num)
{
    printf("%s=%d\\xff\\xff\\xff", name, num);
}

void HMI_send_float(char* name, float num)
{
    printf("%s=%d\\xff\\xff\\xff", name, (int)(num*100));
}

void HMI_Wave(char* name, int ch, int val)
{
    printf("add %s,%d,%d\\xff\\xff\\xff",name, ch, val);
}

void HMI_Wave_Fast(char* name, int ch, int count,int* show_data)
{
    int i;
    printf("addt %s,%d,%d\\xff\\xff\\xff", name,ch, count);
    delay_cycles(100);
    for (i= 0 ; i< count; i++)
        printf("%c", show_data[i]);
    printf("/\\xff/\\xff/\\xff");
}
```

```
}

void HMI_clear(char* name, int ch)
{
    printf("cle %s,%d\xff\xff\xff", name, ch);
}

int taskString = 0;

void data_parsing(void)
{
    while(usize() >= FRAMELENGTH)
    {
        if(u(0) != 0x99 || u(1) != 0x98)
        {
            udelete(1);
        }
        else
        {
            break;
        }
    }

    if(usize() >= FRAMELENGTH && u(0) == 0x99 && u(1) == 0x98 && u(2) == 0x11 &&
    u(3) == 0xff && u(4) == 0xff && u(5) == 0xff)
    {
        taskString = 1;
        udelete(FRAMELENGTH);
    }

    if(usize() >= FRAMELENGTH && u(0) == 0x99 && u(1) == 0x98 && u(2) == 0x22
    && u(3) == 0xff && u(4) == 0xff && u(5) == 0xff)
    {
        taskString = 2;
        udelete(FRAMELENGTH);
    }

    if(usize() >= FRAMELENGTH && u(0) == 0x99 && u(1) == 0x98 && u(2) == 0x33
    && u(3) == 0xff && u(4) == 0xff && u(5) == 0xff)
    {
        taskString = 3;
        udelete(FRAMELENGTH);
    }

    if(usize() >= FRAMELENGTH && u(0) == 0x99 && u(1) == 0x98 && u(2) == 0x44
    && u(3) == 0xff && u(4) == 0xff && u(5) == 0xff)
    {
        taskString = 4;
        udelete(FRAMELENGTH);
    }
}
```

```
/*
if(usize() >= FRAMELENGTH && u(0) == 0x44 && u(2) == 0x01 && u(3) == 0xff &&
u(4) == 0xff && u(5) == 0xff)
{
    int sin_data[255];
    int i = 0;
    int j = 0;
    for (i = 0; i<255; i++)
    {
        sin_data[i] = (int)((sin((i+1)*3.14/50)+1)*90);
        HMI_send_number("page5.n0.val", sin_data[i]);
        HMI_Wave("s0.id", 0, sin_data[i]);
    }
    udelete(FRAMELENGTH);
}

if(usize() >= FRAMELENGTH && u(0) == 0x44 && u(2) == 0x00 && u(3) == 0xff &&
u(4) == 0xff && u(5) == 0xff)
{
    HMI_clear("s0.id", 0);
    udelete(FRAMELENGTH);
}

if(usize() >= FRAMELENGTH && u(0) == 0x66 && u(3) == 0xff && u(4) == 0xff &&
u(5) == 0xff)
{
    int decimalValue0 = u(1);
    int decimalValue1 = u(2);
    kp = (decimalValue1* 256 + decimalValue0*1)/100;
    //HMI_send_number("page7.x3.val", decimalValue0);
    HMI_send_number("page7.x3.val",decimalValue1* 256 + decimalValue0*1);
    udelete(FRAMELENGTH);
}

if(usize() >= FRAMELENGTH && u(0) == 0x77 && u(3) == 0xff && u(4) == 0xff &&
u(5) == 0xff)
{
    int decimalValue0 = u(1);
    int decimalValue1 = u(2);
    ki = (decimalValue1* 256 + decimalValue0)/100;
    HMI_send_number("page7.x4.val",decimalValue1* 256 + decimalValue0*1);
    //HMI_send_float("page7.x4.val",ki);
    udelete(FRAMELENGTH);
}

if(usize() >= FRAMELENGTH && u(0) == 0x88 && u(3) == 0xff && u(4) == 0xff &&
u(5) == 0xff)
{
    int decimalValue0 = u(1);
    int decimalValue1 = u(2);
    kd = (decimalValue1* 256 + decimalValue0)/100;
    HMI_send_number("page7.x5.val",decimalValue1* 256 + decimalValue0*1);
    //HMI_send_float("page7.x5.val",kd);
}
```

```
        udelete(FRAMELENGTH);
    }
    */
}

typedef struct
{
    uint16_t Head;
    uint16_t Tail;
    uint16_t Lenght;
    uint8_t Ring_data[RINGBUFF_LEN];
}RingBuff_t;

uint8_t RxBuff[1];
RingBuff_t ringBuff;

void initRingBuff(void)
{
    //初始化
    ringBuff.Head = 0;
    ringBuff.Tail = 0;
    ringBuff.Lenght = 0;
}

void writeRingBuff(uint8_t data)
{
    if(ringBuff.Lenght >= RINGBUFF_LEN)
    {
        return ;
    }
    ringBuff.Ring_data[ringBuff.Tail]=data;
    ringBuff.Tail = (ringBuff.Tail+1)%RINGBUFF_LEN;//防止非法越界
    ringBuff.Lenght++;
}

void deleteRingBuff(uint16_t size)
{
    if(size >= ringBuff.Lenght)
    {
        initRingBuff();
        return;
    }
    for(int i = 0; i < size; i++)
    {
        if(ringBuff.Lenght == 0)//判断非空
        {
            initRingBuff();
            return;
        }
    }
}
```



```
        ringBuff.Head = (ringBuff.Head+1)%RINGBUFF_LEN;//防止非法越界
        ringBuff.Lenght--;

    }

}

uint8_t read1BFromRingBuff(uint16_t position)
{
    uint16_t realPosition = (ringBuff.Head + position) % RINGBUFF_LEN;

    return ringBuff.Ring_data[realPosition];
}

uint16_t getRingBuffLenght()
{
    return ringBuff.Lenght;
}

void UART_HMI_INST_IRQHandler(void)
{
    switch (DL_UART_Main_getPendingInterrupt(UART_HMI_INST)) {
        case DL_UART_MAIN_IIDX_RX:
            RxBuff[0] = DL_UART_Main_receiveData(UART_HMI_INST);
            writeRingBuff(RxBuff[0]);
            break;
        default:
            break;
    }
}
```

以上是基于TI板子，stm32同理可用。