AN1601E ATK-301 电容指纹识别模块使用说明

本应用文档(AN1601D)将教大家如何在 ALIENTEK 北极星 STM32F7/H7 开发板上使用 ATK-301 电容指纹识别模块。

本文档分为如下几部分：

1， ATK-301 电容指纹识别模块简介
2， 硬件连接
3， 软件实现
4， 验证

1、ATK-301 电容 指纹识别模块简介

ATK-301 电容 指纹识别模块（以下简称 LB301 模块）是 ALIENTEK 推出的一款高性能的电容半导体指纹识别模块。LB301 采用了瑞典 FPC 公司按压式电容指纹传感器，该传感器具有功耗低、稳定、图像一致性效果好、耐静电等级高的特点。模块搭配 GigaDevice（兆易创新)生产的指纹控制专用芯片，针对指纹传感器做出了大量的图像优化、速度优化、算法优化，使模块具有识别速度快，通过率高的等特点。相对传统光学指纹模块，本模块具有识别速度更快、体积更小、功耗更低等特点。

模块配备了串口通讯接口，用户无需研究复杂的图像处理及指纹识别算法，只需通过简单的串口通讯按照通讯协议便可控制模块。本模块可应用于各种考勤机、保险箱柜、指纹门禁系统、指纹锁等场合。技术指标如表 1.1 所示。

表 1.1 技术指标：

项目	说明
工作电压(V)	3.0~3.6V，典型值：3.3V
工作电流(mA)	20~50mA，典型值：40mA
静态功耗（uA）	3~10uA，典型值：5uA
工作环境	温度(°C):-20~70
传感器图像大小(pixel)	192*192pixel 分辨率 508DPI
对比速度	1:1<6ms
拒真率(FRR)	<1%
认假率(FAR)	<0.001%
指纹存容量	500 枚(ID:0~499)
使用寿命	1,000,000 次

2、硬件连接

2.1 模块接口

通讯接口：标准 UART TTL 电平波特率：默认 57600bps，1 位起始位，1 位停止位，3.3V TTL 电平

连接器：6Pin 1.25mm 间距端子

模块接口：引脚描述如表 2.1.1 所示。

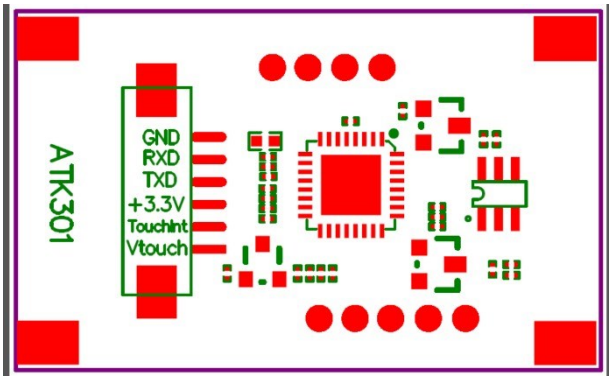


图 **2.1.1** 模块接口定义

表 2.1.1 ATK-301 电容 模块引脚描述

序号	名称	说明
1	V_TOUCH	2.8~3.6V 触摸唤醒电路 VCC（一直供电）
2	TOUCH_OUT	唤醒输出信号（模块 VCC 断电有效）
3	VCC	模块供电 3.3V
4	TX	UART TX
5	RX	UART RX
6	GND	接地

说明： TOUCH_OUT 仅在 VCC 断电情况下才有信号输出，VCC 正常供电情况下，输出低电平；VCC 断电情况下，没有手指按压输出低电平，有手指按压输出 V_TOUCH 高电平。

2.2 与开发板连接

LB301 模块与开发板连接关系如下图 2.2.1 所示，表 2.2.1 为 LB301 模块与开发板链接关系表。

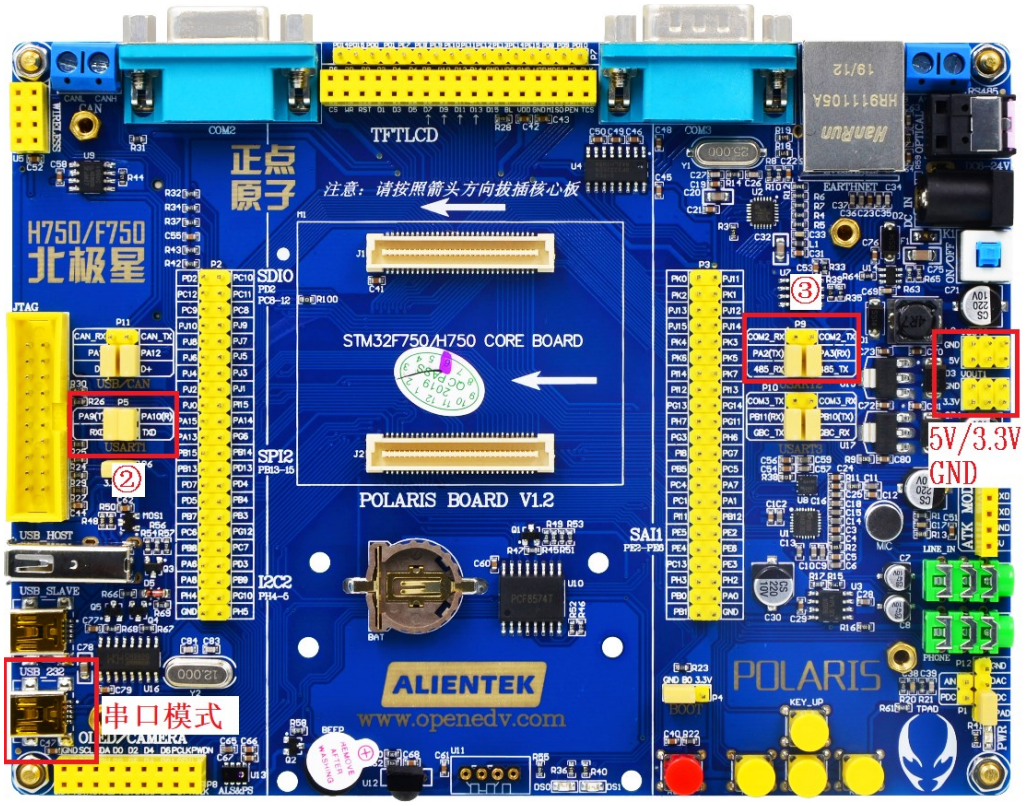


图 2.2.1　LB301 模块与北极星 STM32F7/H7 开发板连接关系图

- **例程实验演示模式：**LB301 模块的 Tx、Rx 分别连接到③中的 PA3(RX)、PA2(Tx)。
- **上位机串口测试：**LB301 模块的 Tx、Rx 分别连接到②中的 RXD、TXD，USB 数据线接到串口模式。

连接方式：取下跳线帽使用杜邦线连接。

LB301 模块指纹识别模块与开发板连接关系							
LB301 模块		VCC	TX	RX	GND	V_TOUCH	TOUCH_OUT
开发板	例程实验演示模式	3.3V	PA3(RX)	PA2(TX)	GND	3.3V	不接
	上位机串口测试	3.3V	RXD	TXD	GND	3.3V	不接

表 2.2.1　LB301 模块与北极星 STM32F7/H7 开发板连接关系表

注：上位机串口测试及使用串口助手调试的方法在模块资料\ATK-301 电容指纹识别模块用户手册中说明。本文档只说明例程实验演示模式。

3、软件实现 （以 STM32F750 为例）

本实验主要实现录入指纹、刷指纹（验证指纹）、使用 USMART 读取和修改模块参数等功能。程序是在北极星 STM32F7/H7 开发板的汉字显示实验和 T9 拼音输入法实验基础上进行修改的。并增加了 usart2.c、LB301.c，这里我们使用 usart2.c 与 LB301 模块通讯，usart2.c 里面使用结合定时器超时接收完成数据的机制。这里，我们就不再介绍 usart2.c,主要看 LB301.c 和 main.c 的代码，首先是 LB301.c，该文件是 LB301 模块的驱动代码，LB301.c 部分代码如下：

```
u32 LB301Addr = 0XFFFFFFFF; //默认
```

```
//串口发送一个字节
static void MYUSART_SendData(u8 data)
{
    while((USART2->SR&0X40)==0);
    USART2->DR = data;
}
//发送包头
static void SendHead(void)
{
    MYUSART_SendData(0xEF);
    MYUSART_SendData(0x01);
}
//发送地址
static void SendAddr(void)
{
    MYUSART_SendData(LB301Addr>>24);
    MYUSART_SendData(LB301Addr>>16);
    MYUSART_SendData(LB301Addr>>8);
    MYUSART_SendData(LB301Addr);
}
//发送包标识,
static void SendFlag(u8 flag)
{
    MYUSART_SendData(flag);
}
//发送包长度
static void SendLength(int length)
{
    MYUSART_SendData(length>>8);
    MYUSART_SendData(length);
}
//发送指令码
static void Sendcmd(u8 cmd)
{
    MYUSART_SendData(cmd);
}
//发送校验和
static void SendCheck(u16 check)
{
    MYUSART_SendData(check>>8);
    MYUSART_SendData(check);
}
//等待应答
```

```
//waittime：等待超时时间（单位 1ms）
//返回接收数组的首地址
static u8 *WaitBack(u16 waittime)
{
    char *data;
    u8 str[8];
    str[0]=0xef;                    str[1]=0x01;
    str[2]=LB301Addr>>24;  str[3]=LB301Addr>>16;
    str[4]=LB301Addr>>8;    str[5]=LB301Addr;
    str[6]=0x07;                    str[7]='\0';
    USART2_RX_STA=0;
    while(--waittime)
    {
        delay_ms(1);
        if(USART2_RX_STA&0X8000)//接收到一次数据
        {
            data=strstr((const char*)USART2_RX_BUF,(const char*)str);
            if(data)
                return (u8*)data;
        }
    }
    return 0;
}
//录入图像  PS_GetImage
//功能:探测手指，探测到后录入指纹图像存于 ImageBuffer。
//模块返回确认字
u8 PS_GetImage(void)
{
    u16 temp;
    u8   ensure;
    u8   *data;
    SendHead();
    SendAddr();
    SendFlag(0x01);//命令包标识
    SendLength(0x03);
    Sendcmd(0x01);
    temp =   0x01+0x03+0x01;
    SendCheck(temp);
    data=WaitBack(500);//等待返回（ms 超时机制）
    if(data)
        ensure=data[9];
    else
        ensure=0xff;
```

```
        memset(USART2_RX_BUF,0,USART2_RX_STA&0x7fff); //数据清 0
        USART2_RX_STA = 0;
        return ensure;
    }
```

LB301.c 代码比较多，里面很多都是指令，指令的格式都是一样，所以我们仅贴出部分代码进行讲解一下。

首先，是配置串口发送指令的包头、指令码、校验和之类。

第二个函数 static u8 * WaitBack (u16 waittime)，里面调用了<sting.h>中的 strstr(const char*str1, const char* str2);这个函数是判断 str2 是否包含在 str1 内，如果包含则返回包含数据的首地址，否则返回 NULL。这里我们用于判断串口中断接收的数据包中有没有包含应答指令的包头、模块地址、指令码（07）。参数 waittime 是等待判断的时间单位（1ms）。

第三个函数 u8 PS_GetImage(void)，这个函数是和 LB301 通讯获取图像的指令，里面包含发送包头、地址、校验和，和等待接收模块应答指令 WaitBack (500)。LB301.c 中其他指令函数格式都跟这条指令差不多，这里就不一一贴出来讲解。

LB301.c 我们就介绍到这里，我们再来看看 main.c 中主函数代码如下：

```
int main(void)
{
    u8 key;
    u8 ensure;
    u8 key_num;
    char *str;
    u16 count = 0;

    Cache_Enable();                 //打开 L1-Cache
    HAL_Init();                     //初始化 HAL 库
    Stm32_Clock_Init(432,25,2,9);   //设置时钟,216Mhz
    delay_init(216);                //延时初始化
    uart_init(115200);              //初始化串口波特率为 115200

    LED_Init();                     //初始化与 LED 连接的硬件接口
    MPU_Memory_Protection(); //保护相关存储区域
    SDRAM_Init();                   //初始化 SDRAM
    LCD_Init();                     //初始化 LCD
    KEY_Init();                     //初始化按键
    W25QXX_Init();                  //初始化 W25Q256
    my_mem_init(SRAMIN);            //初始化内部内存池
    my_mem_init(SRAMEX);            //初始化外部内存池
    my_mem_init(SRAMDTCM);          //初始化 DTCM 内存池
    my_mem_init(SRAMITCM);          //初始化 ITCM 内存池
    exfuns_init();                  //为 fatfs 相关变量申请内存
    f_mount(fs[0],"0:",1);          //挂载 SD 卡
    f_mount(fs[1],"1:",1);          //挂载 FLASH.
```

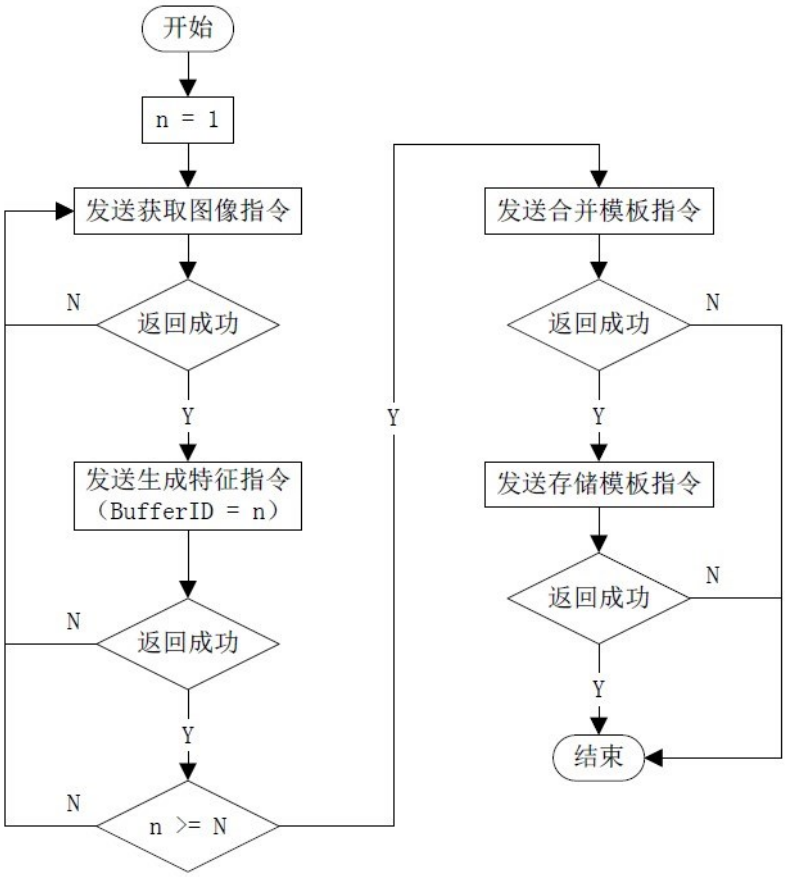
```
PCF8574_Init();                 //初始化 IO 扩展芯片 PCF8574
usart2_init(usart2_baund);      //初始化串口 2,用于与指纹模块通讯
PS_StaGPIO_Init();                  //初始化 FR 读状态引脚
tp_dev.init();                  //初始化触摸屏
while(font_init())              //检查字库
{
    LCD_Clear(WHITE);               //清屏
    POINT_COLOR=RED;                //设置字体为红色
    LCD_ShowString(30,30,200,16,16,"POLARIS STM32");
    while(SD_Init())                //检测 SD 卡
    {
        LCD_ShowString(30,50,200,16,16,"SD Card Failed!");
        delay_ms(200);
        LCD_Fill(30,50,200+30,50+16,WHITE);
        delay_ms(200);
    }
    LCD_ShowString(30,50,200,16,16,"SD Card OK");
    LCD_ShowString(30,70,200,16,16,"Font Updating...");
    key=update_font(20,90,16,"0:");//更新字库
    while(key)//更新失败
    {
        LCD_ShowString(30,90,200,16,16,"Font Update Failed!");
        delay_ms(200);
        LCD_Fill(20,90,200+20,90+16,WHITE);
        delay_ms(200);
    }
    LCD_ShowString(30,90,200,16,16,"Font Update Success!    ");
    delay_ms(1500);
    LCD_Clear(WHITE);//清屏
}

if(!(tp_dev.touchtype&0x80))//如果是电阻屏
{
    Show_Str_Mid(0,30,"是否进行触摸屏校准",16,240);
    POINT_COLOR=BLUE;
    Show_Str_Mid(0,60,"是:KEY2     否:KEY0",16,240);
    while(1)
    {
        key_num=KEY_Scan(0);
        if(key_num==KEY0_PRES)
            break;
        if(key_num==KEY2_PRES)
        {
```

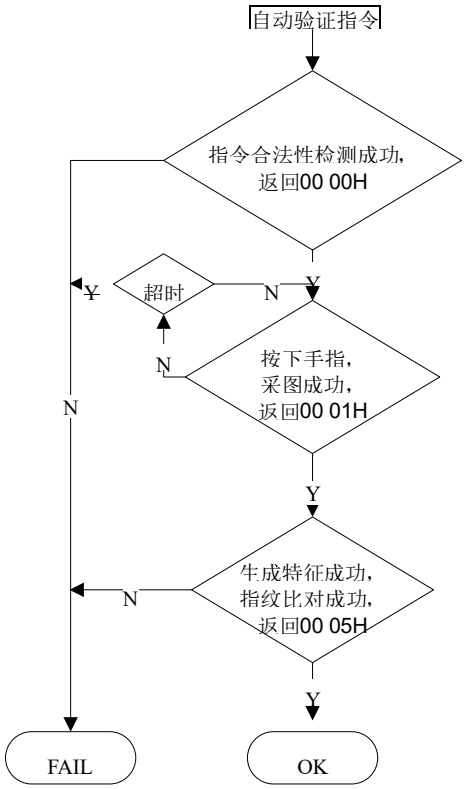
```
                    LCD_Clear(WHITE);
                    TP_Adjust();          //屏幕校准
                    TP_Save_Adjdata();//保存校准参数
                    break;
                }
            }
        }
        /*加载指纹识别实验界面*/
        LCD_Clear(WHITE);
        POINT_COLOR=RED;
        Show_Str_Mid(0,0,"LB301 指纹识别模块测试程序",16,240);
        Show_Str_Mid(0,20,"正点原子 @ALIENTEK",16,240);
        POINT_COLOR=BLUE;
        Show_Str_Mid(0,40,"与 LB301 模块握手....",16,240);
        while(PS_GetImage() == 0xff)//与 LB301 模块握手（读取图像指令有数据返回说明模块已经连接）
        {
            delay_ms(400);
            LCD_Fill(0,40,240,80,WHITE);
            Show_Str_Mid(0,40,"未检测到模块!!!",16,240);
            delay_ms(800);
            LCD_Fill(0,40,240,80,WHITE);
            Show_Str_Mid(0,40,"尝试连接模块...",16,240);
        }
        LCD_Fill(30,40,240,100,WHITE);
        Show_Str_Mid(0,40,"通讯成功!!!",16,240);
        str=mymalloc(SRAMIN,30);
        sprintf(str,"波特率:%d    地址:%x",usart2_baund,LB301Addr);
        Show_Str(0,60,240,16,(u8*)str,16,0);
        ensure=PS_ValidTempleteNum(&ValidN);//读库指纹个数
        if(ensure!=0x00)
            ShowErrMessage(ensure);//显示确认码错误信息
        ensure=PS_ReadSysPara(&LB301Para);  //读参数
        if(ensure==0x00)
        {
            mymemset(str,0,50);
            sprintf(str," 库 容 量 :%d          对 比 等 级 : %d",LB301Para.PS_max-ValidN,LB301Para.PS_level);
            Show_Str(0,80,240,16,(u8*)str,16,0);
        }
        else
            ShowErrMessage(ensure);
        myfree(SRAMIN,str);
```

```
        LB301_load_keyboard(0,170,(u8**)kbd_menu);//加载虚拟键盘
        while(1)
        {
            key_num=LB301_get_keynum(0,170);
            if(key_num)
            {
                if(key_num==1)Del_FR();          //删指纹
                if(key_num==3)Add_FR();          //录指纹
            }
            //查询刷指纹
            if(++count > 10)
            {
                count = 0;
                press_FR();
            }
            delay_ms(1);
        }
    }
```

Main 函数比较简单，初始化硬件→是否触摸校准（电阻屏）→与 LB301 模块通讯→通讯成功读取模块参数→显示模块参数→加载虚拟键盘→while 循环获取触摸键值→判断键值进入录指纹或删指纹流程→查询验证指纹。下面我们讲一下录入指纹和验证指纹：录入指纹流程：


开始
n = 1
发送获取图像指令
N
返回成功
Y
发送生成特征指令
（BufferID = n）
N
返回成功
Y
N
n >= N
Y
发送合并模板指令
返回成功
N
Y
发送存储模板指令
返回成功
N
Y
结束

验证指纹：


自动验证指令
指令合法性检测成功，
返回00 00H
超时
N
Y
N
按下手指，
采图成功，
返回00 01H
Y
N
生成特征成功，
指纹比对成功，
返回00 05H
Y
FAIL
OK

我们的例程代码中分别使用了 2 个 switch 语句按照以上流程图一步步完成录入指纹、刷指纹两个流程。代码简单易懂，这里就不贴出来讲解了，大家可以打开实验工程并参考此流程图理解。

4、验证

首先，使用杜邦线将模块连接到开发板，连接方式按照上述表 2.2.1 LB301 模块与开发板连接关系表中的例程实验演示模式连接。本文档以 ALIENTEK 开发板及 2.8’LCD 进行实验，下载代码到开发板上，显示如下图：

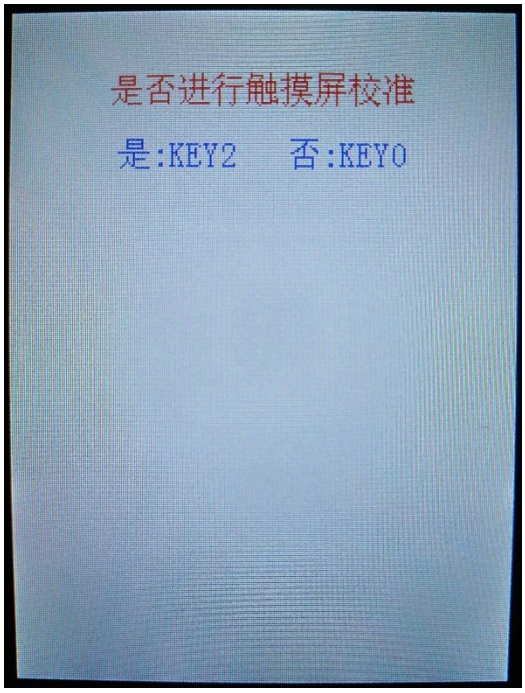


图 4.1 触摸校准界面

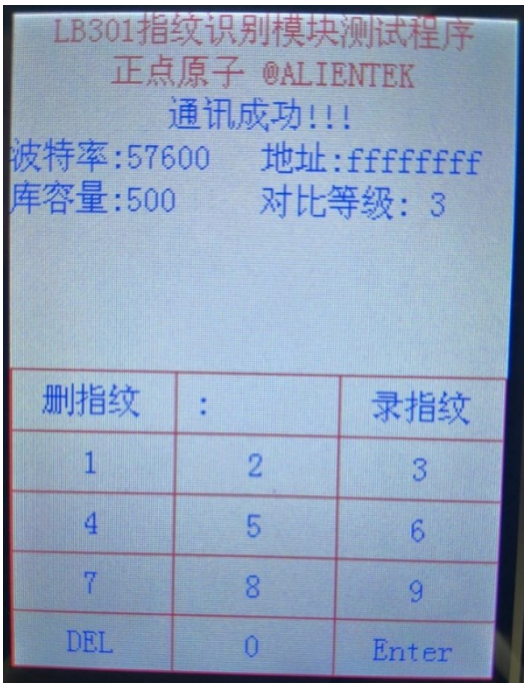


图 4.2 指纹测试主界面

图 4.1 触摸校准界面：当使用电阻屏（2.8’\3.5’LCD）时，字库初始化成功之后弹出此界面，电容屏（4.3’\7’LCD）则无此界面。电容屏不需要触摸校准。

图 4.2 指纹测试主界面：当与 LB301 模块通讯成功之后弹出此界面。界面中库容量即：当前还剩余多少枚指纹容量。对比等级：安全等级。键盘中冒号空格：显示键入的数值。在此界面可触摸功能键有“删指纹”、“录指纹”，还可以刷指纹。

录指纹：按下“录指纹”后按照提示操作如下图：

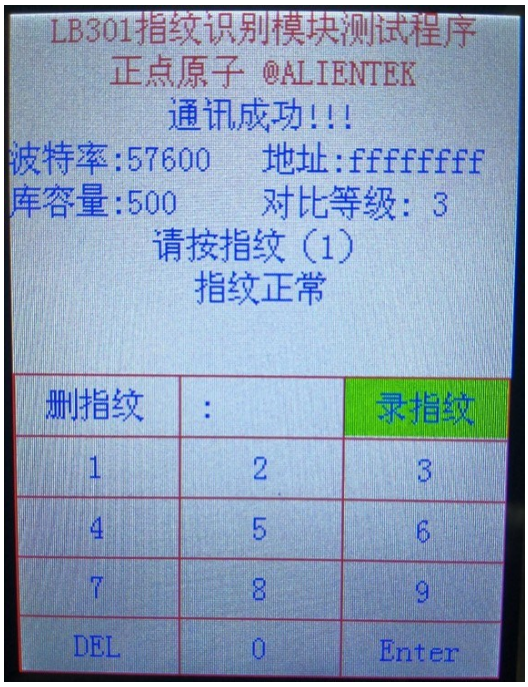


图 4.3 录指纹

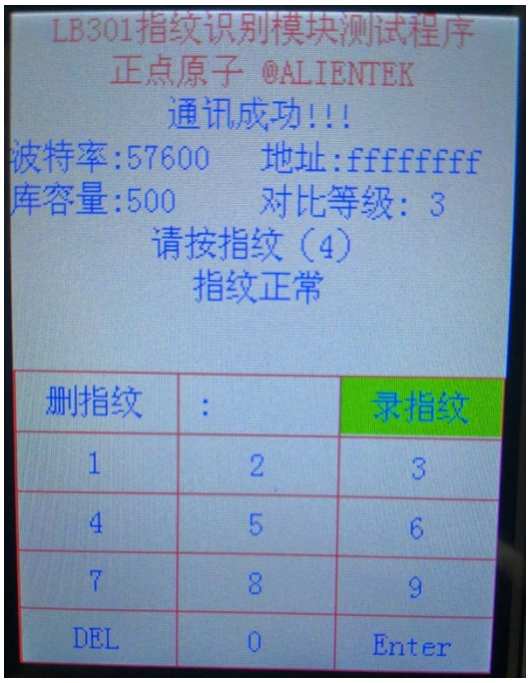


图 4.4 再按一次指纹

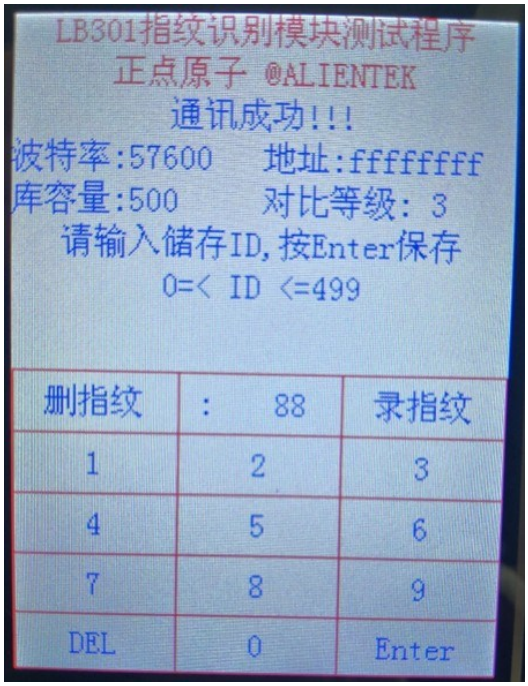


图 4.5 输入存储 ID

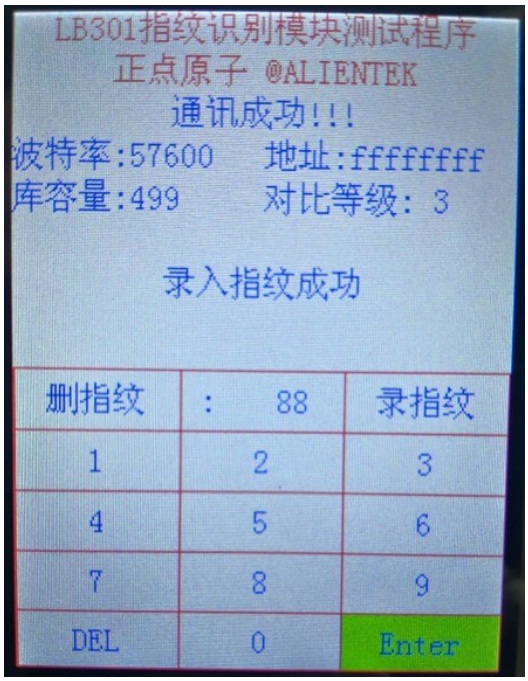


图 4.6 录入指纹成功

当在测试主界面按下“录指纹”，系统会提示“请按指纹”如图 4.3 所示。然后根据提示，按下手指录入第一次指纹，后续一共需要录入 4 次指纹，录入第四次指纹成功之后系统会提示如图 4.4 所示。当第四次指纹也录入成功，系统就会生成指纹模板，接着就会提示“请输入存储 ID，按 Enter 保存”如图 4.5 所示。最后显示录入指纹成功，如图 4.6 所示。录指纹的每一个步骤成功或失败都会提示。

提示：程序中判断在两次按手指时如果 5 次读指纹图像都没有指纹，系统会自动退出

录指纹流程。

刷指纹：在测试主界面下，我们按下没有录入指纹的手指和已经录入指纹的手指，结果分别如下图：

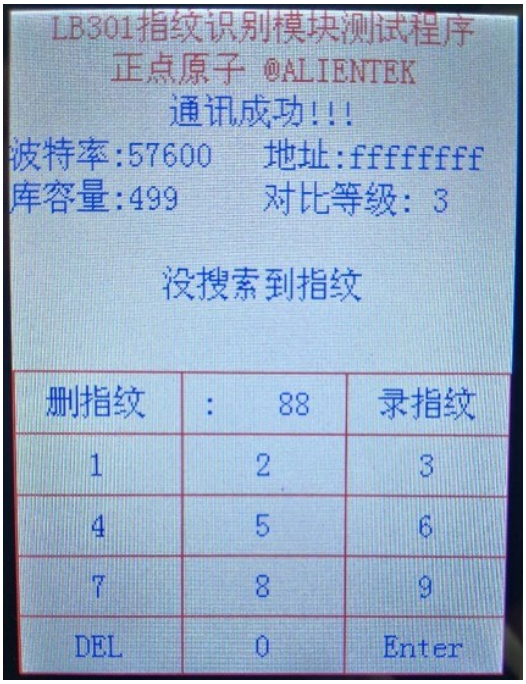


图 4.7 按下无指纹的手指

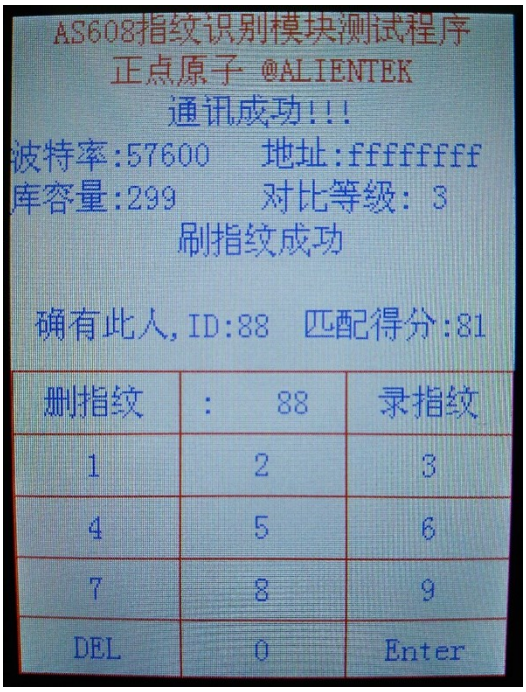


图 4.8 按下有指纹的手指

删指纹：在测试主界面下，按下“删指纹”后按照提示操作如下图：

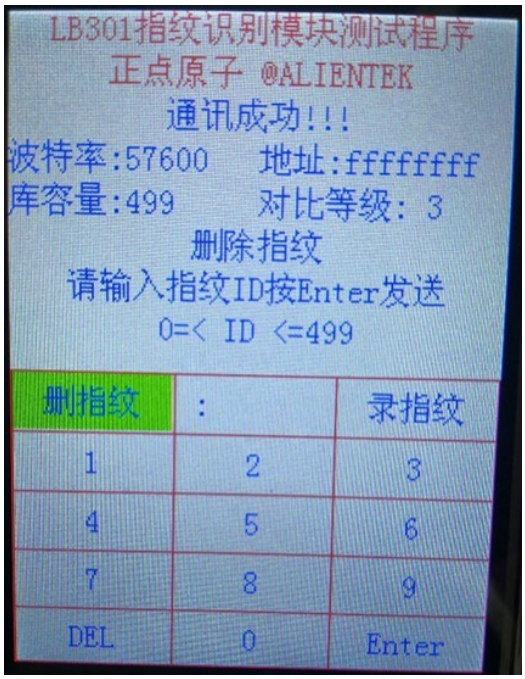


图 4.9 按下“删指纹”

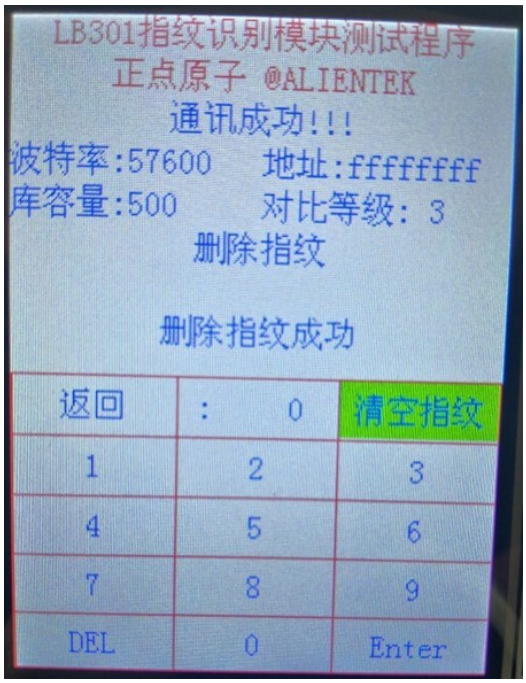


图 4.10 清空指纹库

在删指纹的界面，我们可以删除单个指纹，如图 4.9，我们输入一个指纹 ID 之后再按“Enter”就可以删除单个指纹了。也可以清空指纹库，如图 4.10 所示。如果不想删除了，也可以按“返回”键返回测试主界面。

至此，关于 ATK-301 电容模块的使用介绍，我们就讲完了，本文档详细介绍了 ATK-301 电容模块的使用和使用过程的注意事项。请大家务必按照文档说明操作，如有疑问可联系我们，谢谢！

正点原子@ALIENTEK
2016-6-20
公司网址：www.alientek.com
技术论坛：www.openedv.com
电话：020-38271790
传真：020-36773971

正点原子