Technical Note

HMI-ZDP14X0 应用开发手册

基于显示专用芯片 ZDP14x0 系列

TN01010101 1.2.00 Date:2024/6/21

类别	内容
关键词	HMI-ZDP14x0、应用开发手册
摘要	本文介绍HMI-ZDP14x0D的UI应用开发步骤

修订历史

版本	日期	原因
V1.0.00	2023/08/30	创建文档
V1.1.00	2023/11/23	更新内容描述
V1.2.00	2024/5/23	更新部分配图

目 录

1. HMI-ZDP14X0 显示方案介绍

1.1 HMI-ZDP14X0 方案特点

立功科技基于 AWTK 推出的 HMI-ZDP14x0 显示应用方案，快速启动秒开机，显示更炫酷，设计更灵活。

本方案基于 UI 设计引擎 AWTK Designer 作为上位机设计软件，具有丰富的控件，一拖一拽即可完成界面设计，每个控件都有丰富的可编辑属性、样式、状态，支持弹窗、移动、旋转等丰富的动画效果。界面 UI 设计完成以后，可一键打包资源文件，生成代码，在代码中插入串口数据交互部分代码，只需极低的代码编辑即可实现与设备的通信，数据处理等功能。

1.1.1 方案框图

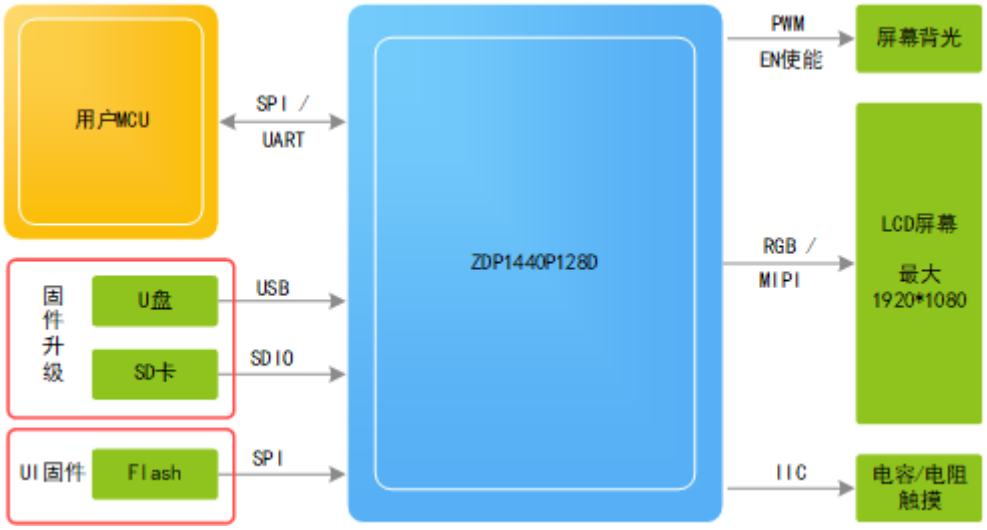

图 1.1 方案框图

1.1.2 方案特性

- 主控采用 ZDP1440P128D/ZDP1460P128D，内置 16MB/64MB 显示内存；
- 基于 AWTK Designer 的拖拽式 UI 设计，所见即所得；
- 丰富的 GUI 控件，提供窗口、对话框和各种常用的控件；
- 内置协议解析器，支持自定义通信协议；
- 上位机配置 LCD 参数，一键 UI 编译打包，快捷方便；
- 支持 SD 卡和 U 盘升级 UI 固件；
- 快速启动，秒开机。

1.2 HMI-ZDP14X0 开发步骤

HMI-ZDP14X0 开发主要分为三大部分，如图 1.2，分别是屏幕参数适配，点亮屏幕；基于 AWTK Designer 设计 UI、编写逻辑功能程序；上位机一键编译打包生成 UI 固件，U 盘或 SD 卡升级固件。

图 1.2　HMI-ZDP14X0 开发流程

1.2.1 屏幕适配

ZDP14x0 系列配套专用的参数配置上位机，用户可根据各自外围设计和所选屏幕等进行参数配置，参数配置支持导入和导出，配置完成后可一键下载到 HMI 板子，接好电源和屏幕，即可点亮屏幕。参数填写和下载，细节操作见《ZDP14x0 系列小技巧-屏幕配置》应用笔记，非常详细，本文不再赘述。

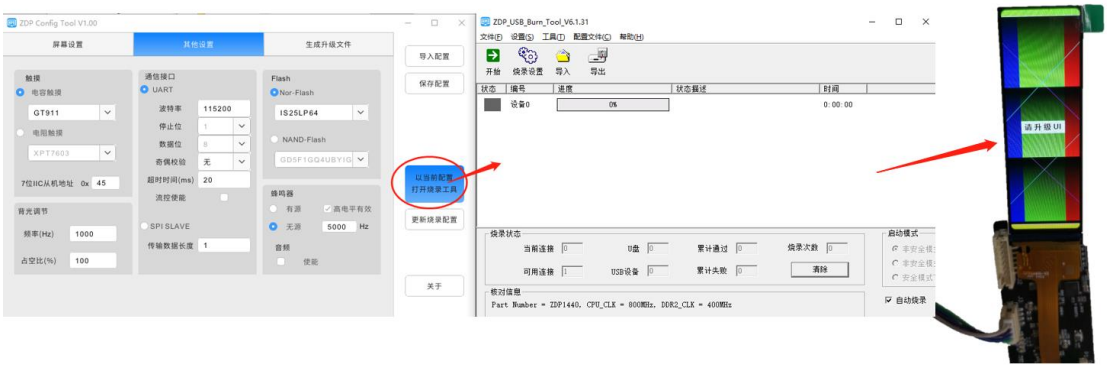

图 1.3　参数配置上位机

1.2.2 UI 开发

1. UI 设计

AWTK Designer 是专门用来制作 AWTK 应用程序 UI 界面的实用工具。只要通过拖拽和点击就可以完成复杂的界面设计，操作简单；可以随时预览效果，所见即所得，并支持一键生成 UI 代码。如图 1.4 所示 AWTK Designer 设计界面。

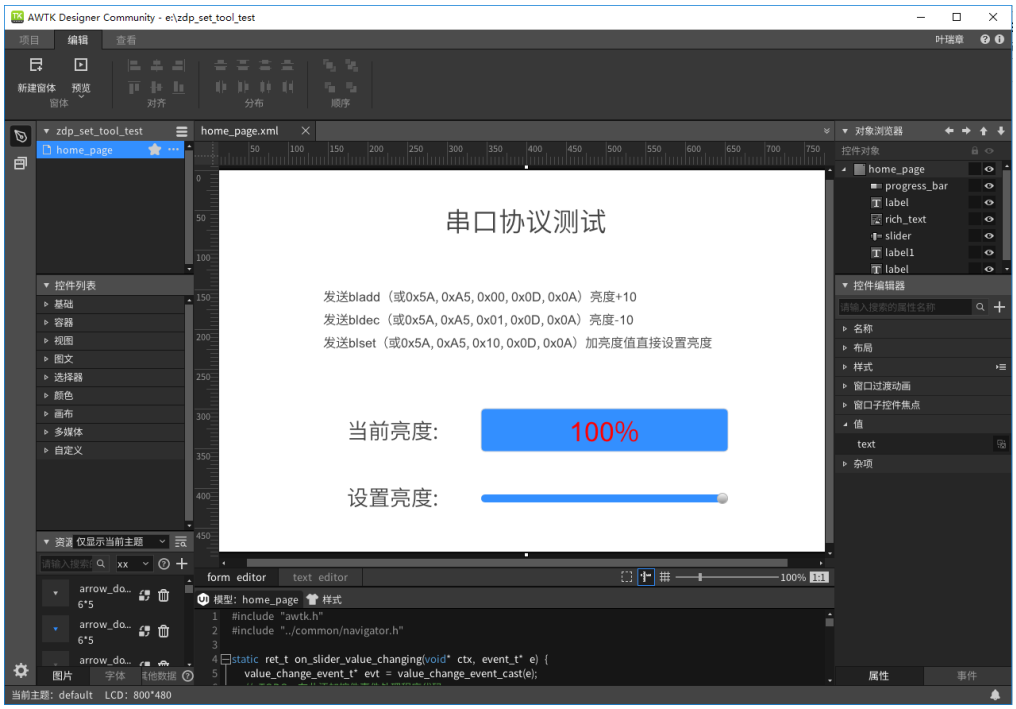

图 1.4　AWTK Designer 设计 UI

2.　编写逻辑功能程序

设计完成 UI 界面后，添加协议解析器或移植现有通信协议代码，增加串口数据处理，添加逻辑功能程序。

如表 1.1 为示例 UI 演示 Demo 命令，UI 运行过程中可以通过串口命令调整屏幕背光亮度。这里默认使用协议解析器，注册字符串命令和 16 进制命令，即发送字符串命令或 16 进制命令都可以调整背光，实际应用时串口命令或通信协议可自行定义。

表 1.1　使用协议解析器注册命令

命令功能	字符串命令	16 进制命令
增加屏幕亮度	bladd	0x5A, 0xA5, 0x00, 0x0D, 0x0A
降低屏幕亮度	bldec	0x5A, 0xA5, 0x01, 0x0D, 0x0A
设置屏幕亮度	blset	0x5A, 0xA5, 0x10, 0x0D, 0x0A

如图 1.5 为根据定义的串口命令，调用命令注册接口注册命令以及回调函数。当收到的数据与命令一致时，执行回调函数。

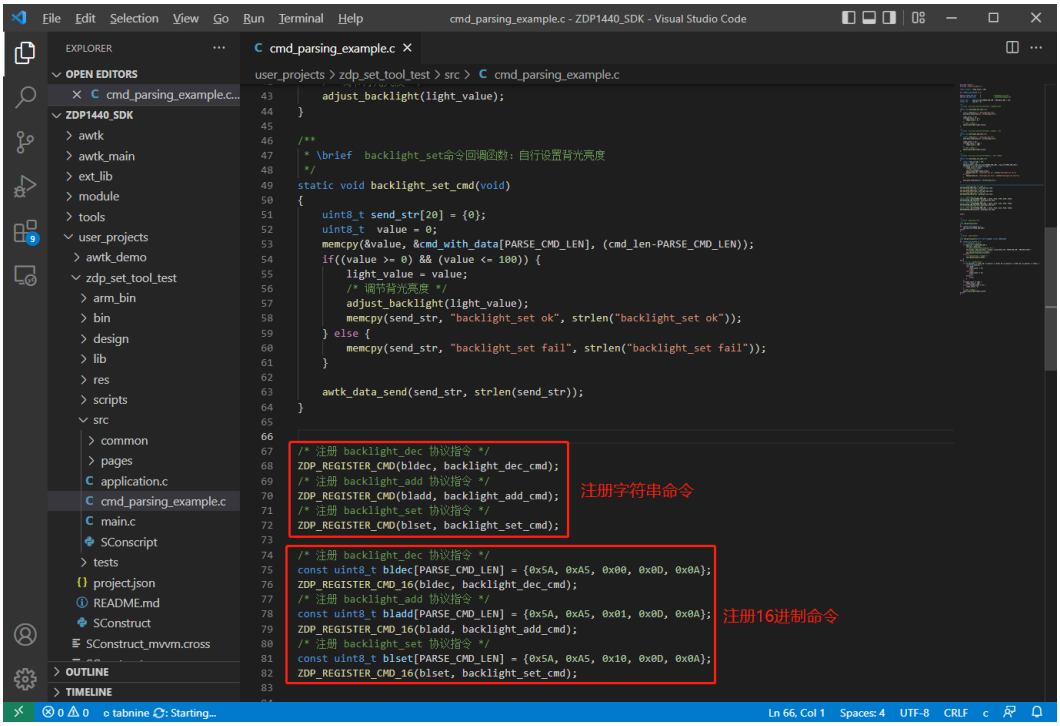

图 1.5　串口命令注册程序

1.2.3 固件升级

1. UI 固件编译

UI 工程逻辑功能程序添加好后上位机一键编译打包即可生成 UI 固件，后续用于 UI 升级。如图 1.6 为上位机一键编译生成 UI 固件的界面。

图 1.6　上位机一键编译生成 UI 固件

2. U 盘或 SD 卡升级

ZDP14x0 支持挂载 U 盘和 SD 卡设备，用于 UI 固件升级非常方便。只需要将 zdp_set_tool.exe 上位机编译打包生成的 UI 固件拷贝到 U 盘或 SD 卡根目录下，然后插入到 HMI-ZDP14x0 硬件主板对应升级接口，便会自动读取并升级 UI 固件。

UI 固件升级完成后自动运行，如图 1.7 为升级完成的演示 Demo 界面。

图 1.7　UI 升级完成自动运行

2. 开发环境搭建

2.1 AWTK 开发环境搭建

AWTK Designer 是专门用来制作 AWTK 应用程序 UI 界面的实用工具。只要通过拖拽和点击就可以完成复杂的界面设计，操作简单；可以随时预览效果，所见即所得。

关于 AWTK 的环境搭建、使用教程和示例 demo 等，AWTK 官网提供了详细的教程及资料，可以通过如下链接获取，本文不在赘述。

1. AWStudio 是用来整合 AWTK Designer 的工具，可以管理 AWTK Designer 创建的项目，通过 AWStudio 可以打开项目对应的 Designer 来编辑项目。
2. AWStudio 安装包下载链接为 https://awtk.zlg.cn/awstudio/download.html。
3. AWStudio 的安装与注册，请参考如下链接在线文档：
4. https://awtk.zlg.cn/docs/awstudio_docs/AWStudio_User_Manual/2.InstallAndRegister.html 。
5. 关于更多 AWTK 的介绍和使用说明，请访问 https://awtk.zlg.cn/docs。

图 2.1 AWTK 社区

3. 开发平台介绍

本文档用于 HMI-ZDP14X0 显示方案软件的开发和应用，在搭建好开发环境后，即可踏上应用开发。本章简单介绍一下本文 UI Demo 示例所依托的相关的硬件平台以及 HMI-ZDP14X0 SDK 包的架构。

3.1 硬件平台说明

HMI-ZDP14X0 是立功科技基于 ZDP14x 设计的 UI 显示方案硬件。主要用于显示专用芯片的评估和功能开发，同时提供硬件相关的参考设计，帮助客户快速入门，缩短研发周期。

HMI-ZDP14X0 方案评估套件实物如图 3.1。

图 3.1 HMI-ZDP14X0 显示评估套件实物图

HMI-ZDP14X0 方案的软件基于 HMI-ZDP14X0 硬件进行开发，有关硬件接口的详细描述请参考【数据手册】HMI-ZDP14X0.pdf。

3.2 软件平台说明

ZDP14x0 的软硬件资料已在 Gitee 开源发布，点击 https://gitee.com/zlgmcuopen/

HMI_ZDP1440D 链接进入，选择克隆或下载即可获取资料。

资料目录	资料说明
01.快速入门手册	快速入门手册，介绍开发流程和芯片特点
02.串口屏应用开发手册	详细介绍从开发环境搭建到应用开发，提供详细指导
03.UI_build_project	提供参数配置上位机，支持一键UI源码编译及打包
04.UI示例固件	UI示例固件，点亮屏幕后可以用于验证硬件
05.硬件设计参考	提供芯片外围电路设计的硬件设计参考，原理图库
06.芯片手册	芯片手册
07.相关技术笔记	芯片相关搭配的技术笔记与文档

图 3.2　ZDP14X0_SDK 包目录结构

这里特别介绍 03.UI_build_project 路径，基本上 UI 开发等都围绕此路径进行，用户开发都基于此路径。

路径	
awtk	AWTK源码存放路径
awtk_main	芯片硬件接口函数
ext_lib	库文件
module	命令解析器等模块
tools	编译工具
user_projects	UI存放路径
CHANGELOG.md	修改记录
clean.bat	清除编译临时文件
zdp_set_tool.exe	上位机
zdp_set_tool使用说明_V1.21.pdf	上位机使用说明

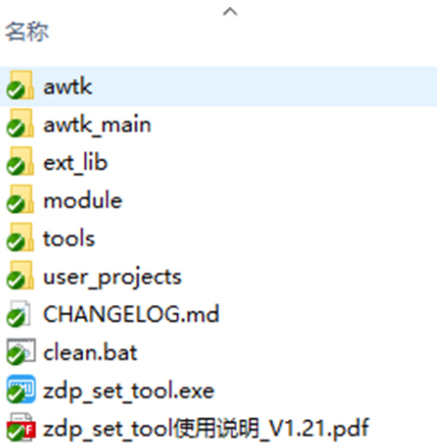

图 3.3　03.UI_build_project 目录说明

如图 3.3 所示为 SDK 的目录结构，下面分别介绍各目录存放的内容及作用：

- awtk：存放 AWTK 源码，用于编译 AWTK 库以及提供 AWTK 的头文件；
- awtk_main：存放 AWTK 向内核提供的函数接口源码，无需修改；
- ext_lib：存放 AWTK 库、C 标准库等库文件；
- module：存放命令解析器、PC 虚拟串口接口文件；
- tools：存放配置烧录工具、编译工具链、打包工具等；
- user_projects：存放 UI，AWTK Designer 新建的 UI 选择放此目录；
- clean.bat：临时文件清理脚本，双击清理临时文件；
- zdp_set_tool.exe：配置工具，包括 LCD 参数配置、触摸型号、通信接口配置、以及 UI 编译打包。

4. 接口说明

HMI-ZDP14x0 UI 工程提供了一些函数接口用于满足用户自定义功能需求，包括背光亮度调节接口、通信接口、调试接口、音频接口等功能函数。

功能函数接口在 awtk_main 下的 awtk_func.h 中声明，下面分别介绍各功能接口的使用。

4.1 printk 调试接口

HMI-ZDP14x0 UI 工程提供了 printk 调试接口函数，调用调试接口可以在调试串口输出格式化数据，函数原型及使用说明如下：

```
/**
 * \brief 调试信息输出接口
 *  注意：单次最大输出字符长度 1023
 *
 * \param[in] fmt : 格式化输出参数
 *
 * \retval 返回 0 : 成功
 * \retval  其他 : 失败
 */
int printk(const char * fmt, ... );
```

- fmt：格式化输出参数。

4.2 数据发送接口

HMI-ZDP14x0 UI 工程提供了数据发送接口函数，函数原型及使用说明如下：

```
/**
 * \brief 数据发送接口
 *
 * \param[in] p_data : 数据发送缓冲区
 * \param[in] nbytes : 发送的数据长度
 *
 * \retval 返回实际发送的数据长度
 */
int awtk_data_send(const uint8_t* p_data, uint32_t nbytes);
```

- p_data：数据发送缓冲区；
- nbytes：发送的数据长度。

4.3 串口波特率设置接口

HMI-ZDP14x0 UI 工程提供了串口波特率设置接口函数，函数原型及使用说明如下：

```
/**
 * \brief 串口屏通信串口波特率设置
 *
 * \param[in] baud_rate : 波特率参数，范围 1200~3000000
 *
 * \retval 返回 0 : 成功
```

```
 * \retval 其他  : 失败
 */
int uart_set_baud(uint32_t baud_rate);
```

- baud_rate：串口波特率，范围 1200~3000000。

4.4 蜂鸣器控制接口

HMI-ZDP14x0 UI 工程提供了蜂鸣器控制接口函数，函数原型及使用说明如下：

```
/**
 * \brief 蜂鸣器鸣叫接口
 *
 * \param[in] nms : 蜂鸣器鸣叫时间，单位 ms
 *
 * \retval 无
 */
void beep_on_ms(uint32_t nms);
```

- nms：蜂鸣器鸣叫时间，单位毫秒。

4.5 背光亮度调节接口

HMI-ZDP14x0 UI 工程提供了背光亮度调节接口函数，函数原型及使用说明如下：

```
/**
 * \brief LCD 背光亮度调节
 *
 * \param[in] arg : PWM 占空比，范围 0~100
 *
 * \retval 无
 */
void adjust_backlight(unsigned long arg);
```

- arg：PWM 占空比，范围为 0~100。

4.6 RTC 实时时钟接口

HMI-ZDP14x0 UI 工程提供了 RTC 的接口函数，接口函数原型及功能介绍如表 4.1 所示。

表 4.1 RTC 实时时钟接口

函数原型	功能介绍
int rtc_cfg_init(void);	RTC 初始化函数
int rtc_set_time(systime_t sys_time);	RTC 时间设置函数
int rtc_get_time(systime_t *sys_time);	RTC 时间获取函数

1. RTC 初始化函数

```
/**
 * \brief RTC 配置初始化
 *
 * \retval 返回 0 : 成功
```

```
 * \retval 其他  : 失败
 */
int rtc_cfg_init(void);
```

- 调用初始化函数，进行 RTC 硬件初始化；

2. RTC 时间设置函数

```
/**
 * \brief RTC 时间设置
 *
 * \param[in] sys_time : 待设置的 RTC 时间参数
 *
 * \retval 返回 0 : 成功
 * \retval 其他  : 失败
 */
int rtc_set_time(systime_t sys_time);
```

- sys_time：用于设置时间的结构体。

3. RTC 时间获取函数

```
/**
 * \brief RTC 时间获取
 *
 * \param[in] sys_time : 获取的 RTC 时间参数
 *
 * \retval 返回 0 : 成功
 * \retval 其他  : 失败
 */
int rtc_get_time(systime_t *sys_time);
```

- sys_time：用于获取时间的结构体指针。

4.7 音频播放接口

HMI-ZDP14x0 UI 工程提供了音频播放的接口函数，接口函数原型及功能介绍如表 4.2 所示。

表 4.2 音频播放接口

函数原型	功能介绍
int audio_play_file(char* audio_file);	音频文件播放
int audio_volume_value_get(void);	播放音量获取
int audio_volume_value_set(int value);	播放音量设置
int audio_gain_value_get(void);	播放增益获取
int audio_gain_value_set(int value);	播放增益设置
void audio_suspend_set(int suspend);	音频暂停
void audio_close(void);	音频关闭

1. 音频文件播放

```
/**
```

```
 * \brief 音频文件播放
 *
 * \param[in] audio_file : 文件路径
 *
 * \retval 返回 0 : 成功
 * \retval 其他  : 失败
 */
int audio_play_file(char* audio_file);
```

- audio_file：指定播放文件的路径和文件名；

2. 播放音量获取

```
/**
 * \brief 音频播放音量获取
 *
 * \retval 音量大小 -90～20
 */
int audio_volume_value_get(void);
```

- 返回值：播放音量大小，范围-90～20。

3. 播放音量设置

```
/**
 * \brief 音频播放音量设置
 *
 * \param[in] value : 音量大小 -90～20
 *
 * \retval 返回 0 : 成功
 * \retval 其他  : 失败
 */
int audio_volume_value_set(int value);
```

- value：设置播放音量大小，范围-90～20。

4. 播放增益获取

```
/**
 * \brief 音频播放增益获取
 *
 * \retval 增益大小 0～6
 */
int audio_gain_value_get(void);
```

- 返回值：播放增益大小，范围 0～6。

5. 播放增益设置

```
/**
 * \brief 音频播放增益设置
 *
 * \param[in] value : 增益大小 0～6
```

```
 *
 * \retval 返回 0： 成功
 * \retval 其他  ： 失败
 */
int audio_gain_value_set(int value);
```

- value：设置播放增益大小，范围 0～6。

6. 音频暂停

```
/**
 * \brief 音频暂停
 *
 * \param[in] suspend : 1 暂停，0 继续播放
 *
 * \retval 无
 */
void audio_suspend_set(int suspend);
```

- suspend：1 暂停，0 继续播放。

7. 音频关闭

```
/**
 * \brief 音频关闭
 *
 * \retval 无
 */
void audio_close(void);
```

- 调用音频关闭函数，关闭当前播放的音频。

4.8 电阻屏触摸校准接口

HMI-ZDP14x0 UI 工程提供了电阻屏触摸校准接口函数，函数原型及使用说明如下：

```
/**
 * \brief 电阻屏触摸校准
 *
 * \retval 无
 */
void ts_calibrate_restart(void);
```

- 调用电阻屏触摸校准函数，UI 进入校准界面，可重新校准触摸。

5. 开发示例

5.1 UI 设计示例

5.1.1 创建 UI 项目

首先使用 AWTK Designer 创建 UI 工程，项目路径选择 ZDP1440_SDK/user_projects 目录，这样 UI 设计完成后可在 ZDP Config Tool 上位机一键编译生成 UI 固件。根据屏幕信息修改 LCD 宽度、高度以及颜色格式，如分辨率较高建议使用 16bit 颜色深度，参考图 5.1。

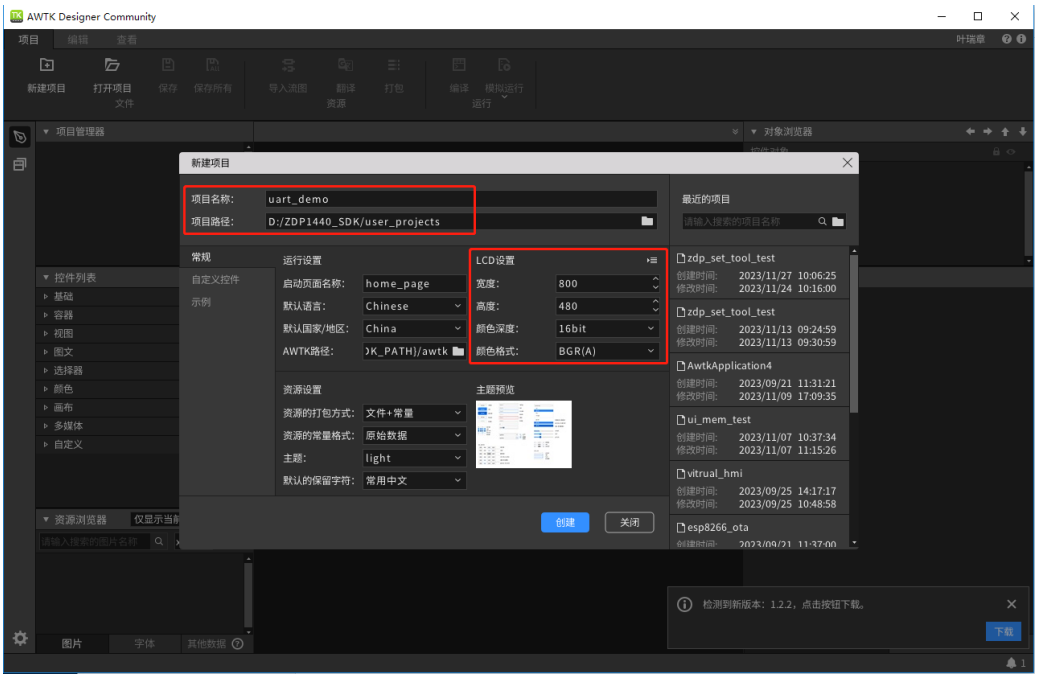

图 5.1 创建 UI 工程

5.1.2 创建 UI 示例

然后拖拽控件生成 UI 界面，选中控件，在属性中可以修改该控件的名称、样式和大小，此次示例中，生成四个控件，包括两个“标签”、一个“按钮”还有一个“进度条”，同时修改按钮控件的名称为“send_btn”，详见图 5.2。

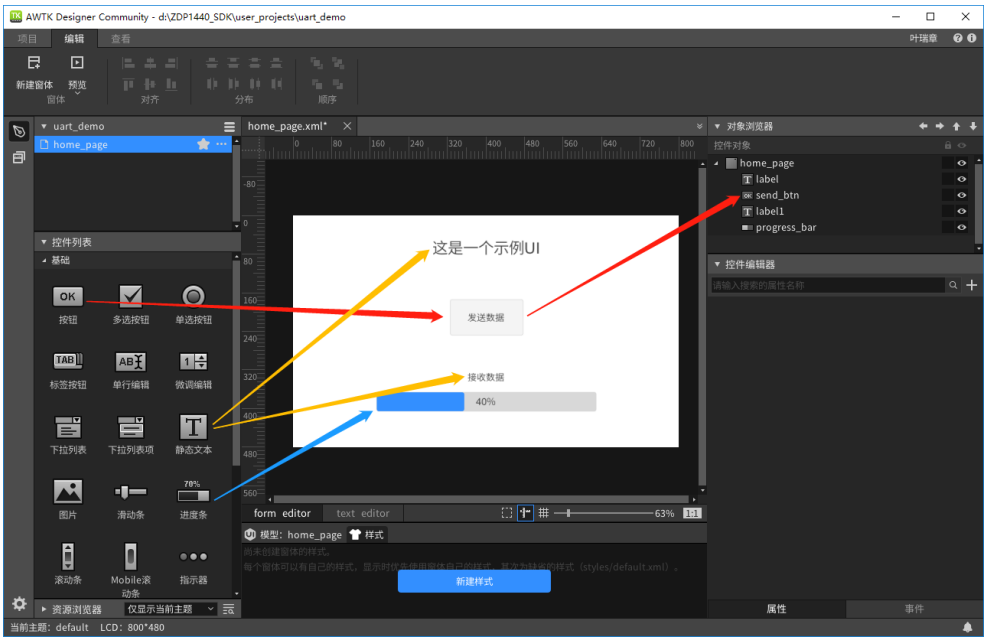

图 5.2 拖拽生成 UI

5.1.3 添加回调函数

选中按钮控件，在右下角点击“事件”栏，点击“+”号，新增 CLICK 事件，动作选择“执行回调函数”，这样，每次点击按钮，都会执行该按钮的回调函数，详见图 5.3。

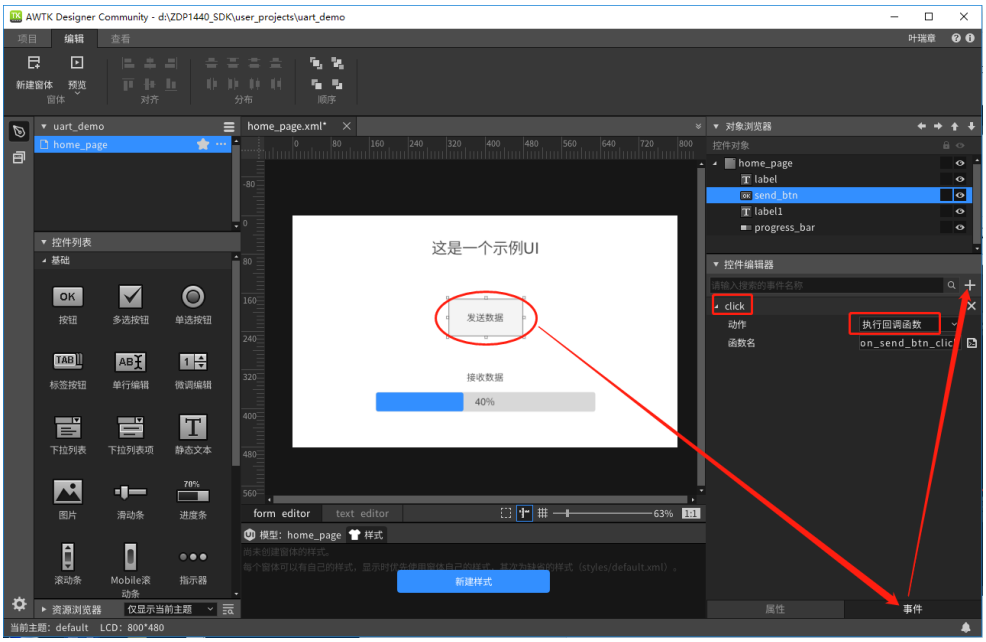

图 5.3 添加按钮回调函数

5.1.4 裁剪字体

嵌入式平台由于资源的限制，中文字符数量庞大，字体文件需要通过裁剪，仅保留需要用到的中文字符；如图 5.4，在左下角选中字体栏，点击字体栏上方的裁剪设置，可以先将默认的“保留的字符”清空，再添加字符集中的“拉丁语”，拉丁语中包含基本的符号、数字以及大小写英文字母：

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
```

然后将 UI 中实际用到的或可能用到的中文字符添加在其后面，在此示例中，用到的中文字符为：“这是一个示例发送接收数据”。配置好需要保留的字符后，点击确定关闭弹窗，再点击字体栏中对应字体右侧的“裁剪字体按钮”。裁剪完成后字体文件中仅存有需要保留的字符，字体文件大小也缩小为“17.4KB”。

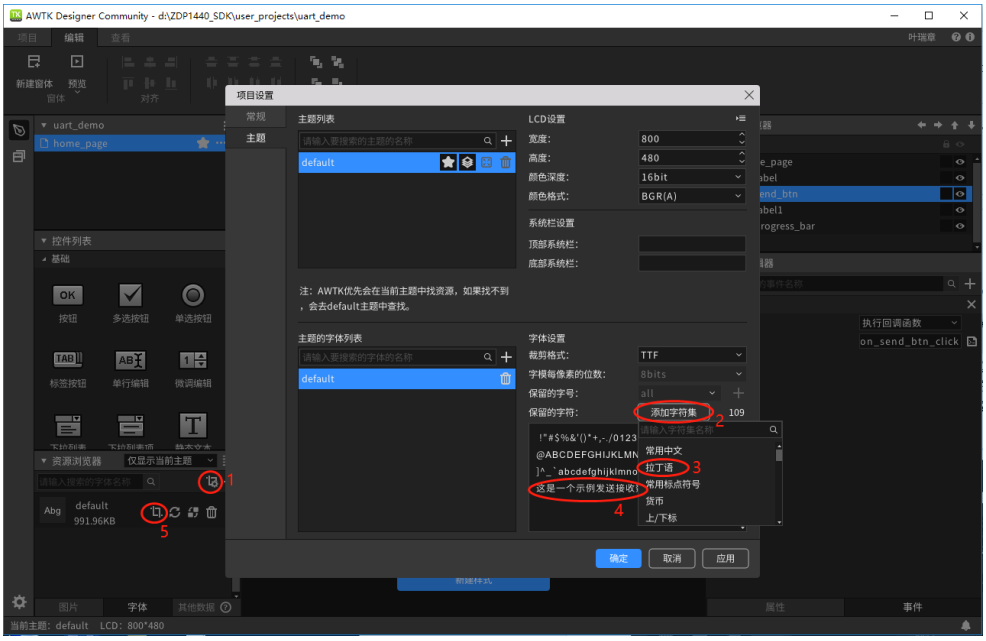

图 5.4　裁剪字体

5.1.5 打包 UI

上述工作完成后，点击打包按钮，软件会自动将 UI 打包，如图 5.5 所示；打包的作用为：将字体、图片、翻译等文件中的内容转换成二进制数组，以便嵌入式平台更加方便地取用。

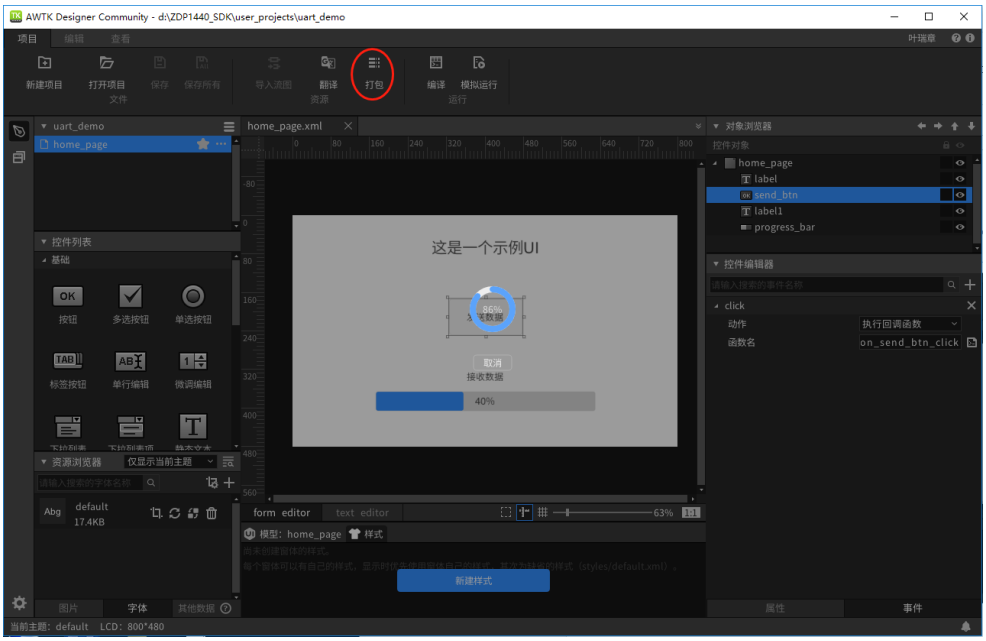

图 5.5 打包 UI

5.2 添加逻辑功能程序

5.2.1 定义串口命令示例

定义一个简单的串口指令，用来反馈按钮的状态及进度条的数据：

- 串口接收帧的帧头为：0xA5 0x5A
- 串口发送帧的帧头为：0x5A 0xA5
- 帧尾统一为：“\r\n”也就是：0x0D 0x0A

点击按钮，串口发送“0x5A, 0xA5, 0x01, 0x0D, 0x0A”

接收到“0xA5, 0x5A, 0xXX, 0x0D, 0x0A”时，进度条的值设置成 XX 对应的十进制值；

5.2.2 添加串口发送指令

打开 ZDP1440_SDK\user_projects\uart_demo\src\page\home_page.c 文件，如图 5.6，在按钮点击事件回调函数中，添加数据发送程序，调用数据发送接口 awtk_data_send 发送定义的数据。

至此，点击按钮发送数据程序已经编写完成，接下来添加接收数据处理程序。

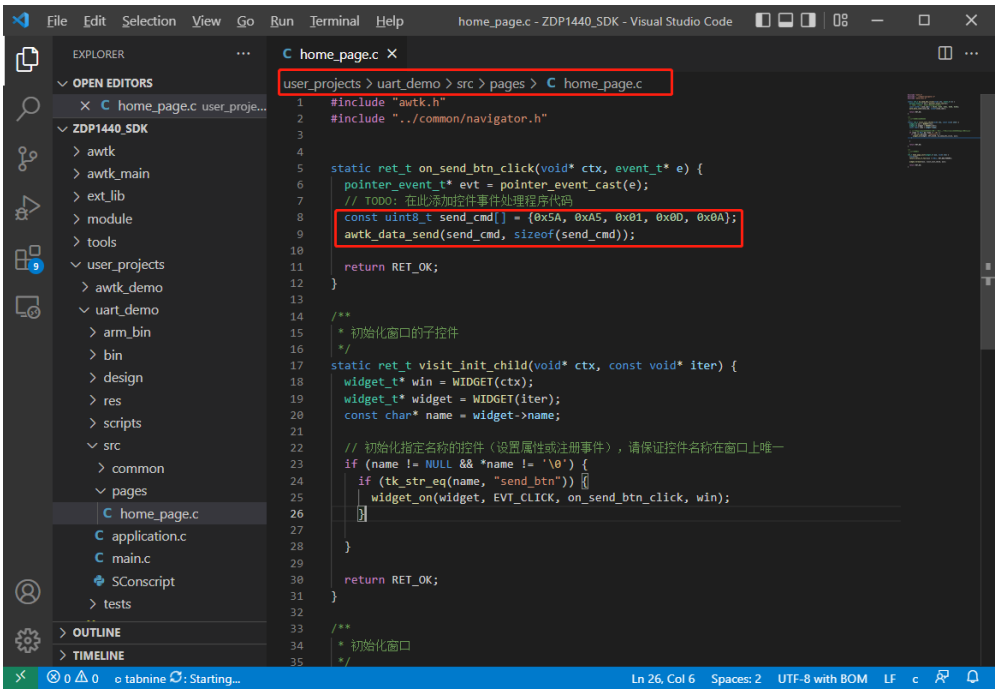

图 5.6　添加按钮事件回调处理

5.2.3 添加协议解析器

首先打开 ZDP Config Tool 上位机，如图 5.7，在【生成升级文件】页面，点击【刷新】按钮，然后选择前面新建的 UI 工程，点击【添加 PC 端虚拟串口&命令解析器文件】。

由于添加的命令解析器不支持中间带数据的协议解析，所以这个 demo 不使用默认的协议解析器，而是自行解析，因此不勾选【使能命令解析器】。

图 5.7　添加协议解析器模板

5.2.4 串口数据处理

如图 5.8，添加协议解析模板后，在 UI 的 src 目录多了 cmd_parsing_example.c 文件。接下来基于模板，修改协议解析。

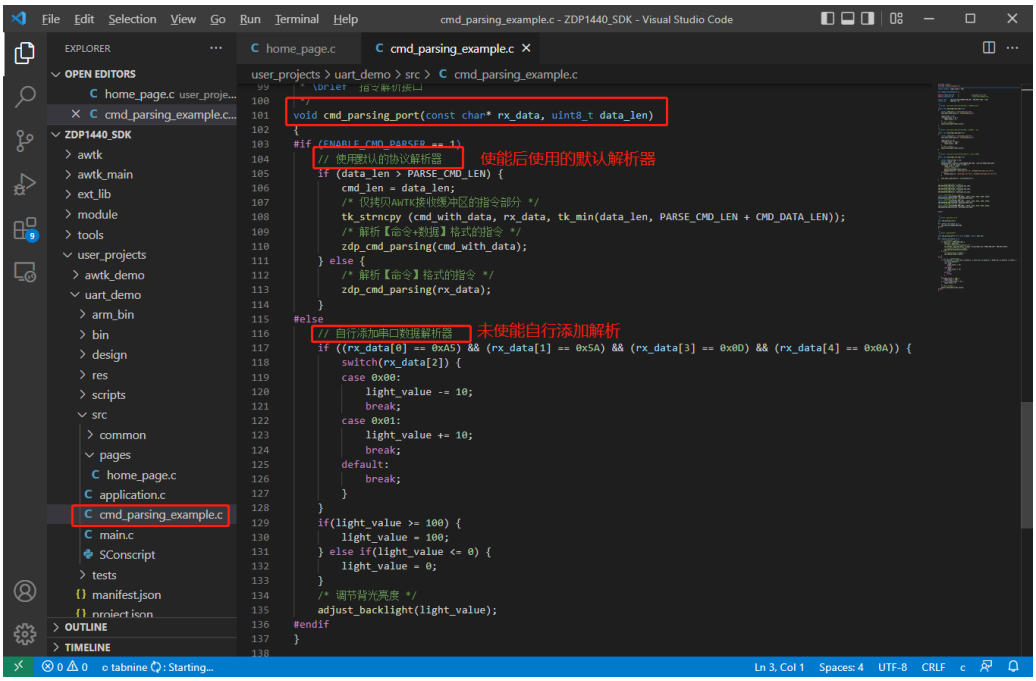

图 5.8　协议解析器模板

在指令解析接口函数中，自行解析取出接收到的数据，详见程序清单 1。

程序清单 1　自行解析命令取出数据程序

```
/* 定义一个全局变量 */
uint8_t progress_value = 0;
/**
 * \brief   指令解析接口
 */
void cmd_parsing_port(const char* rx_data, uint8_t data_len)
{
#if (ENABLE_CMD_PARSER == 1)
    // 使用默认的协议解析器
    if (data_len > PARSE_CMD_LEN) {
        cmd_len = data_len;
        /* 仅拷贝 AWTK 接收缓冲区的指令部分 */
        tk_strncpy (cmd_with_data, rx_data, tk_min(data_len, PARSE_CMD_LEN + CMD_DATA_LEN));
        /* 解析【命令+数据】格式的指令 */
        zdp_cmd_parsing(cmd_with_data);
    } else {
        /* 解析【命令】格式的指令 */
        zdp_cmd_parsing(rx_data);
    }
```

```
#else
    // 自行添加串口数据解析器
    if ((rx_data[0] == 0xA5) && (rx_data[1] == 0x5A) && (rx_data[3] == 0x0D) && (rx_data[4] == 0x0A)) {
        /* 指令正确，取出对应数据，赋值到全局变量 */
        progress_value = rx_data[2];
    }
    if(progress_value >= 100) {
        progress_value = 100;
    } else if(progress_value <= 0) {
        progress_value = 0;
    }
#endif
}
```

5.2.5 UI 显示更新

数据处理完成，接下来修改 home_page 页面对应的 home_page.c 文件，注册 progress_bar 的 idle 回调函数，idle 回调函数会在每一帧都执行一次，确保显示的实时性，详见程序清单 2。

程序清单 2　idle 回调时刷新进度条

```
extern uint8_t progress_value;
static ret_t refresh_bar_idle(const idle_info_t* idle)
{
    widget_t* progress_bar = WIDGET(idle->ctx);
    if ((uint8_t)progress_bar_get_percent(progress_bar) != progress_value) {
        progress_bar_set_value(progress_bar, progress_value);
    }
    return RET_REPEAT;
}
/**
 * 初始化窗口的子控件
 */
static ret_t visit_init_child(void* ctx, const void* iter) {
  widget_t* win = WIDGET(ctx);
  widget_t* widget = WIDGET(iter);
  const char* name = widget->name;

  // 初始化指定名称的控件（设置属性或注册事件），请保证控件名称在窗口上唯一
  if (name != NULL && *name != '\0') {
    if (tk_str_eq(name, "send_btn")) {
      widget_on(widget, EVT_CLICK, on_send_btn_click, win);
    } else if (tk_str_eq(name, "progress_bar")) {
        widget_add_idle(widget, refresh_bar_idle);
    }
  }
```

```
    return RET_OK;
}
```

至此，串口数据解析和 UI 逻辑功能程序皆已完成，接下来编译打包 UI 生成升级固件。

5.3 测试验证

5.3.1 一键生成 UI 固件

编译打包 UI 前，先点击【其他设置】，根据硬件参数配置好 Flash 型号，然后切换到【生成升级文件】页面，如图 5.9，选择需要编译的 UI 工程，接着点击【生成固件】，一键编译生成 UI 固件，用于 U 盘、SD 卡升级。

如图 5.10，生成的升级文件存放在上位机同级目录下，即 ZDP1440_SDK 文件夹。Flash 为 NOR Flash 时对应固件名为 ui_nor.bin，Flash 为 NAND Flash 时对应固件名为 ui_nand.bin。

图 5.9 一键打包生成 UI 升级文件

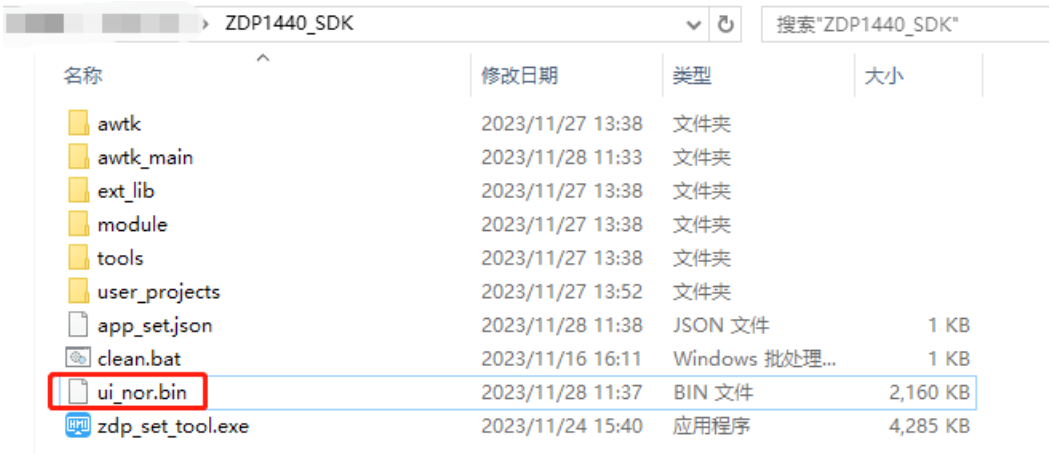

图 5.10　生成的 UI 升级文件

5.3.2 运行测试

将 UI 固件拷贝到 SD 卡，升级后主板运行，运行结果如图 5.11、图 5.12 所示，按下按钮，串口接收到 5A A5 01 0D 0A，向串口发送 A5 5A 10 0D 0A/A5 5A 20 0D 0A，进度条与发送的数据一致。

图 5.11　运行结果串口数据

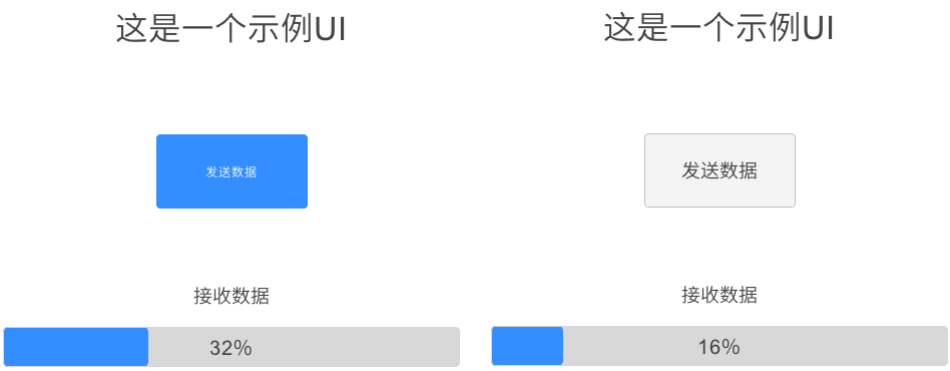

图 5.12　运行结果 UI 界面

6. 免责声明

本着为用户提供更好服务的原则，广州立功科技股份有限公司（下称“立功科技”）在本手册中将尽可能地为用户呈现详实、准确的产品信息。但介于本手册的内容具有一定的时效性，立功科技不能完全保证该文档在任何时段的时效性与适用性。立功科技有权在没有通知的情况下对本手册上的内容进行更新，恕不另行通知。为了得到最新版本的信息，请尊敬的用户定时访问立功科技官方网站或者与立功科技工作人员联系。感谢您的包容与支持！

专业 . 专注成就梦想

Dreams come true with professionalism and dedication.