乐升半导体
LEVETOP

LEVETOP

UI_Editor-Lite 二代串口屏

使用说明书

V1.0

www.levetop.cn
Levetop Semiconductor Co.，Ltd.

版本记录

版 本	发 布 日 期	改 版 说 明
V1.0	2025/07/18	版本发布

版权说明

目　录

UI_Editor-Lite 二代串口屏 1
版本记录 2
版权说明 2
目　录 3
图 目 录 10
表 目 录 17
1. 概述 19
1.1. TFT 串口屏的软硬件架构 19
1.2. UI_Editor-Lite 软件设置与显示功能开发的基本流程 19
2. UI_Editor-Lite 软件的安装及附带资料说明 20
2.1. 下载压缩包 20
2.2. 压缩包解压缩 21
2.3. UI_Editor-Lite 压缩包内附带资料说明 22
2.4. 启动软件 23
2.5. 检查软件 24
2.6. 创建快捷方式 25
3. 软件介绍 26
3.1. 软件界面介绍 26
3.2. 软件功能栏 27
3.2.1. File 功能 27
3.2.2. Tool 功能 29
3.2.3. Help 功能 30
4. 新建工程 32
4.1. 工程文件夹内容说明 32
4.1.1. 素材格式要求 32
4.1.2. UI_projectname.h 文件 35
4.2. 工程设置 37
4.3. 新建工程步骤 40
5. 页面操作说明 42
5.1. 页面设置及参数 42
5.2. 页面跳转 43

5.3. 新增页面 45
5.4. 复制页面 45
5.5. 清空页面 46
5.6. 删除页面 47
5.7. 多余页处理说明 47
5.8. 基础操作说明 47
5.8.1. 添加控件 47
5.8.2. 选中控件操作 48
5.8.3. 删除控件（Delete） 48
5.8.4. 克隆控件（Widget_Clone） 48
5.8.5. 拷贝粘贴功能（Widget Copy and Paste） 49
5.8.6. 微调控件位置功能 50
5.8.7. 上一步和下一步说明 51
5.8.8. 解锁和锁定说明 51
5.9. 快捷键操作汇总 51
6. 控件使用 52
6.1. 按键（Button） 52
6.2. 弹窗（Popupbox） 53
6.3. 变量调节（Variable Button） 55
6.4. 多变量操作（Multi-Variable Button） 56
6.5. 环形触摸（Circular Touch） 57
6.5.1. 用 Icon 控件控制一组图片的方式制作环形触摸进度条举例： 58
6.6. 滑条（Slider Bar） 60
6.6.1. 用小图标的方式制作滑条效果 61
6.7. 键盘系统的使用说明 63
6.7.1. 键盘使用教程 63
6.7.2. 键盘键值（SingleKey） 66
6.7.3. 数字键盘（Numeric Keypad） 67
6.7.4. 键盘键值设置说明 69
6.8. 字符串显示（String_Label） 70
6.9. 静态文本显示（Static_Text） 71
6.9.1. 静态文本使用说明 72
6.10. 字库数字（Text Number Display） 74
6.11. 图片数字（Graphics Number Display） 75
6.12. 数字时钟（Digital Clock） 76
6.12.1. 修改时间说明 77

6.13. 计时器（Timer） 78
6.14. 动图（Gif） 80
6.15. 音频播放（Audio Play） 82
6.16. 位元状态（Bit Status） 83
6.17. 小图标（Icon） 84
6.18. 编码器（Encoder） 86
6.18.1. 编码器使用说明 86
6.18.2. 编码器旋转切换页面功能说明 89
6.18.3. 编码器 item 设置 90
6.19. 自动变量（Automatic variable） 93
6.20. 条形进度条 94
6.21. 环形进度条 95
6.22. 页面布局控件说明 97
6.22.1. 控件左对齐（Left_Align） 98
6.22.2. 控件右对齐（Right_Align） 98
6.22.3. 控件上对齐（Top_Align） 99
6.22.4. 控件下对齐（Bottom_Align） 99
6.22.5. 控件等宽（Width_Align） 100
6.22.6. 控件等高（Height_Align） 100
6.22.7. 控件等宽高（Shape Consistent） 101
6.22.8. 水平等距（Horizontal Equidistance） 101
6.22.9. 垂直等距（Vertical Equidistance） 102
6.22.10. 放大缩小功能（Zoom in or out） 102
7. 存储空间与变量地址说明 104
7.1. RAM 的储存空间 104
7.2. 变量地址（writeAddr） 104
7.3. 参数地址（parameterAddr） 104
7.4. 特殊寄存器地址说明 104
8. 多国语言功能说明 107
8.1. 通过图片切换实现多国语言切换 107
8.1.1. 多国语言 Icon 文件夹设置 107
8.1.2. 其他语言的文件夹素材说明 108
8.1.3. 多国语言所支持的控件 108
8.1.4. 多国语言切换过程 108
8.2. 通过字符编码切换实现多国语言切换 109
8.2.1. 采集 Unicode 字库 109
8.2.2. 设置多国语言 110

8.2.3. 多国语言切换过程 110
9. 配套工具使用说明 111
9.1. 二代串口屏模拟器（UI_Emulator-II）使用说明 112
9.1.1. 模拟器入口及主界面说明 112
9.1.2. 模拟器的变量操作 114
9.1.3. 模拟器写数据举例 117
9.1.4. 模拟器模拟编码器的使用 117
9.1.5. 模拟器显示带旋转角度的工程的注意事项 118
9.1.6. 模拟器的限制说明 118
9.1.7. 模拟器写数据参考 119
9.2. 串口调试工具（UI_Debugger-II）使用说明 120
9.2.1. 软件主界面 120
9.2.2. 指令发送使用教程 123
9.2.3. 指令文档说明 125
9.2.4. 指令信息文档说明 126
9.3. 字库取模工具使用说明 127
9.4. 图片编号工具使用说明 133
9.5. 音频转换工具使用说明 136
9.5.1. 制作 Wav 文件 136
9.5.2. Wav 转 bin 文件 139
10. 串口通信指令 142
10.1. 串口写数据指令 143
10.1.1. 发送串口指令控制控件举例 144
10.1.2. 发送串口指令控制寄存器举例 148
10.2. 串口读数据指令 154
10.3. 触摸反馈指令 155
10.4. CRC 计算代码 157
10.4.1. 读写数据指令的 CRC 计算 159
10.4.2. 读返回指令的 CRC 计算 159
10.4.3. 触摸反馈指令的 CRC 计算 160
10.5. 串口发送直接修改控件参数 160
10.6. 键码触发说明 161
11. ModBus 通信功能 162
11.1. 创建 ModBus 主机指令文件 162
11.2. ModBus 主机指令编辑界面 163
11.3. ModBus 主机指令结构 164

11.4. ModBus 主机指令说明.....167
11.4.1. 主机读保持寄存器（指令码 0x03）.....167
11.4.2. 主机读输入寄存器（指令码 0x04）.....168
11.4.3. 主机写单个保持寄存器（指令码 0x06）.....169
11.4.4. 主机写多个保持寄存器（指令码 0x10）.....170
11.4.5. 主机读线圈状态（指令码 0x01）.....171
11.4.6. 主机读离散输入（指令码 0x02）.....172
11.4.7. 主机写单个线圈（指令码 0x05）.....173
11.4.8. 主机写多个线圈（指令码 0x0F）.....174
11.5. Modbus 通信指令的 CRC 计算.....174
11.6. Modbus 工程及程序设置例程.....175
11.6.1. Modbus 从机工程设置.....175
11.6.2. Modbus 主机工程设置.....175
11.7. Modbus 指令执行模式具体使用教程.....177
11.7.1. Modbus 指令执行模式 Operation 0x00.....177
11.7.2. Modbus 指令执行模式 Operation 0x01.....177
11.7.3. Modbus 指令执行模式 Operation 0x02.....178
11.7.4. Modbus 指令执行模式 Operation 0x03.....179
12. 读写外部 SPI Flash 和 MCU Flash 说明.....181
12.1. 指令说明.....181
12.2. 通信过程说明.....182
12.3. CRC 计算及代码.....183
12.3.1. CRC16 的产生.....183
12.3.2. CRC32 的产生.....183
12.4. Flash_RW-II 软件使用说明.....184
12.5. 通过串口读写.....186
13. 横竖屏 UI 工程说明.....189
13.1. 横竖屏 UI 工程要求.....189
13.1.1. 页面内容和页码要求.....189
13.1.2. 控件要求.....190
13.2. 横竖屏工程的合并及烧录过程.....191
13.2.1. 合并工程文件.....191
13.2.2. 更改 mcu_code.....195
13.3. 发指令切换 UI 工程.....195
14. 控件叠加显示说明.....196
14.1. 叠加限制说明.....196
14.2. 叠加使用说明.....196

15.注意事项说明.....197
15.1. 第二代串口屏使用概括及常用文件说明.....197
15.1.1. 第二代串口屏使用的概括.....197
15.1.2. 常用文件说明.....197
15.2. 以旧工程素材创建新工程说明.....198
15.3. 工程备份说明.....199
15.4. 画面旋转操作设置说明.....200
15.5. UartTFT-II_Flash.bin 大小与素材关系说明.....201
15.5.1. 图片素材转 bin 文件说明.....201
15.5.2. UartTFT-II_Flash.bin 组成.....201
15.6. 数据类型说明.....202
15.7. 整数位和小数位设置.....203
15.8. 同组的 Icon 宽高说明.....203
15.9. 删除控件底图说明.....203
15.10. 控件初值设置说明.....205
15.11. 控件堆叠的说明.....205
15.12. 控件超出屏幕范围的说明.....205
15.13. 电脑屏幕分辨率导致软件字体显示不全的说明.....206
15.14. UI_Editor-Lite 使用环境说明.....209
15.15. 各类素材和文件的命名说明.....209
15.16. 素材库说明.....209
15.17. dataFormat 说明.....210
15.17.1. 不同数据格式的结构说明.....210
15.17.2. Icon 和 Gif 的 dataFormat 选择说明.....211
16.附录.....212
16.1. 附录 1 LT165 烧录说明.....212
16.1.1. TFT 串口屏的升级更新概述.....212
16.1.2. LT165 的烧录说明.....213
16.2. 附录 2 UI_Editor-Lite 支持的 IC 及其参数限制.....218
16.2.1. UI_Editor-Lite 支持的 IC 列举.....218
16.2.2. 不同 IC 的参数限制.....218
16.2.3. 不同 IC 的单页控件数量限制.....220
16.2.4. 不同 IC 的变量地址范围及寄存器地址说明.....221
16.3. 附录 3 TFT 串口屏的演示板（Demo Kit）.....222
16.4. 附录 4 串口屏开发流程.....223

16.5. 附录 5 一代与二代串口通信的差异.....226
16.6. 附录 6 二代串口屏可实现效果.....227
16.7. 附录 7 OTA 差分升级说明.....228
16.7.1. 概述.....228
16.7.2. OTA 差分升级流程说明.....228
16.7.3. 软件使用说明.....228

图目录

图 1-1：主控端 MCU 的 Uart 串口对 TFT 串口屏进行通信 19
图 2-1：下载专区 20
图 2-2：选择版本 20
图 2-3：UI_Editor-Lite 压缩包 20
图 2-4：解压软件压缩包 21
图 2-5：解压缩完成 21
图 2-6：UI_Editor-Lite 压缩包内容（1） 22
图 2-7：UI_Editor-Lite 压缩包内容（2） 23
图 2-8：运行软件 23
图 2-9：进入设置界面 24
图 2-10：错误显示 24
图 2-11：正常显示 24
图 2-12：创建快捷方式 25
图 2-13：快捷方式创建成功 25
图 2-14：剪切或复制快捷方式 25
图 2-15：粘贴快捷方式 25
图 3-1：软件界面 26
图 3-2：控件栏 26
图 3-3：软件功能栏 27
图 3-4：file 功能 27
图 3-5：工具列表 29
图 3-6：变量表说明 29
图 3-7：Help 功能 30
图 3-8：writeAddr 设置 30
图 3-9：File Timestamp 设置 30
图 4-1：工程文件夹说明 32
图 4-2：底图命名 32
图 4-3：Icon 命名 33
图 4-4：字库命名 33
图 4-5：Gif 命名 34
图 4-6：音频命名 34
图 4-7：editorConfig.ini 配置文件 35
图 4-8：工程界面设置 37
图 4-9：新建工程入口 40
图 4-10：新工程命名 40
图 4-11：添加素材 41
图 4-12：进入新工程界面 41
图 5-1：查看页面参数 42
图 5-2：页面参数列表 42

图 5-3：无滑动效果的滑动跳页参数.....44
图 5-4：添加新页.....45
图 5-5：复制页面.....45
图 5-6：清除页面内容.....46
图 5-7：删除页面.....47
图 5-8：添加控件.....47
图 5-9：生成控件.....47
图 5-10：框选控件.....48
图 5-11：控件复制.....49
图 5-12：拷贝控件.....49
图 5-13：粘贴控件.....50
图 5-14：微调控件位置.....50
图 6-1：按键参数.....52
图 6-2：弹窗参数.....53
图 6-3：变量调节参数.....55
图 6-4：多变量操作参数.....56
图 6-5：环型触控举例.....57
图 6-6：环型触控参数.....57
图 6-7：小图标控件设置要求.....58
图 6-8：环形触摸控件设置要求.....59
图 6-9：滑条参数.....60
图 6-10：滑条举例.....60
图 6-11：Icon 设置要求.....61
图 6-12：滑条设置要求.....62
图 6-13：键盘页素材.....63
图 6-14：添加无按压效果底图.....63
图 6-15：添加有按压效果底图.....64
图 6-16：设置键盘按键.....64
图 6-17：键盘控件设置.....64
图 6-18：键盘按键控件参数.....66
图 6-19：数字键盘参数.....67
图 6-20：字符串参数.....70
图 6-21：静态文本参数.....71
图 6-22：静态文本设置入口.....72
图 6-23：静态文本设置主界面.....72
图 6-24：工程设置多国语言.....73
图 6-25：对应的编号选择.....73
图 6-26：字库数字参数.....74
图 6-27：图片数字参数.....75
图 6-28：数字时钟参数.....76
图 6-29：数字时钟样式.....76

图 6-30：计时器参数 78
图 6-31：Gif 参数 80
图 6-32：音频参数 82
图 6-33：位元状态参数 83
图 6-34：Icon 参数 84
图 6-35：编码器参数 86
图 6-36：编码器模式一 Icon 设置 87
图 6-37：模式一编码器设置 87
图 6-38：编码器模式二 Icon 设置（左图）与编码器设置（右图） 88
图 6-39：设置页面滑动参数 89
图 6-40：编码器 Mode 设置界面 90
图 6-41：编码器子功能一 90
图 6-42：编码器子功能二 91
图 6-43：编码器子功能三 91
图 6-44：编码器子功能四 92
图 6-45：自动变量控件参数 93
图 6-46：条形进度条参数 94
图 6-47：环形进度条参数 95
图 6-48：布局控件 97
图 6-49：左对齐效果图 98
图 6-50：右对齐效果图 98
图 6-51：上对齐效果图 99
图 6-52：下对齐效果图 99
图 6-53：等宽效果图 100
图 6-54：等高效果图 100
图 6-55：等宽高效果图 101
图 6-56：水平等距效果图 101
图 6-57：垂直等距效果图 102
图 6-58：编辑界面放大 103
图 6-59：编辑界面缩小 103
图 7-1：变量地址内数据的储存方式 104
图 8-1：多国语言 Icon 文件夹设置 107
图 8-2：其他语言的文件夹素材说明 108
图 8-3：多国语言切换过程 109
图 8-4：输入多国语言 110
图 8-5：多国语言预览 110
图 9-1：配套处理工具 111
图 9-2：模拟器入口一 112
图 9-3：模拟器入口二 112
图 9-4：视频解码器安装提示 112
图 9-5：UI_Emulator-II 主界面 113

图 9-6：模拟器的变量操作栏.....114
图 9-7：手动输入框.....114
图 9-8：基本数据类型选择.....115
图 9-9：String 数据类型选择及编码类型选择.....115
图 9-10：工程地址列表.....116
图 9-11：模拟器写变量.....117
图 9-12：模拟器写字符串数据.....117
图 9-13：模拟器旋转角度设置.....118
图 9-14：串口调试工具入口.....120
图 9-15：软件主界面.....120
图 9-16：用户自定义波特率.....121
图 9-17：指令全选功能.....122
图 9-18：软件参数配置.....123
图 9-19：发送指令.....123
图 9-20：指令信息区.....123
图 9-21：一次发送多条指令.....124
图 9-22：延时指令.....124
图 9-23：设置延时时间.....125
图 9-24：导入导出普通指令.....125
图 9-25：指令文档内容.....125
图 9-26：指令信息记录文档.....126
图 9-27：字库取模工具入口.....127
图 9-28：字库取模工具主界面.....127
图 9-29：编码类型选择.....128
图 9-30：字体字号选择.....128
图 9-31：字符示例.....128
图 9-32：字库制作过程 1.....130
图 9-33：字库制作过程 2.....130
图 9-34：自定义字库制作过程 1.....131
图 9-35：自定义字库制作过程 2.....132
图 9-36：自定义字库制作过程 3.....132
图 9-37：图片编号工具入口.....133
图 9-38：图片工具主界面.....133
图 9-39：选择图片.....134
图 9-40：顺序调整.....135
图 9-41：编号完成.....135
图 9-42：自动编号效果展示.....135
图 9-43：格式工厂主界面.....136
图 9-44：添加音频文件.....137
图 9-45：输出配置.....137
图 9-46：截取音频.....138

图 9-47：开始转换 138
图 9-48：转换完成 138
图 9-49：打开音频转换工具 139
图 9-50：音频转换工具主界面 139
图 9-51：选择 Wav 文件 140
图 9-52：设置音频参数 141
图 9-53：命名保存 141
图 10-1：写数据指令举例（1） 143
图 10-2：写数据指令举例（2） 143
图 10-3：文字控件参数 144
图 10-4：写字符串指令举例 144
图 10-5：串口发送文字 145
图 10-6：数字控件参数对比 145
图 10-7：写数字指令举例 146
图 10-8：串口发送数字 146
图 10-9：位元状态参数设置 147
图 10-10：写位元状态指令举例 147
图 10-11：串口控制位元状态显示 147
图 10-12：Icon 参数设置 148
图 10-13：写小图标切换指令举例 148
图 10-14：串口发送切换 Icon 148
图 10-15：写页面跳转指令举例 149
图 10-16：写控制屏幕亮度指令举例 149
图 10-17：写修改时间指令举例 149
图 10-18：写控制音频播放指令举例（1） 150
图 10-19：写控制音频播放指令举例（2） 150
图 10-20：写控制音频播放指令举例（3） 150
图 10-21：写音量调节指令举例 150
图 10-22：写电阻屏校准指令 151
图 10-23：开启自动背光指令 151
图 10-24：关闭自动背光指令 151
图 10-25：写休眠背光亮度指令举例 151
图 10-26：写休眠背光时间指令举例 152
图 10-27：写串口升级指令举例 152
图 10-28：写屏幕检测指令举例（1） 152
图 10-29：写屏幕检测指令举例（2） 152
图 10-30：写屏幕检测指令举例（3） 153
图 10-31：退出屏幕检测指令 153
图 10-32：读数据指令举例（1） 154
图 10-33：读数据指令举例（2） 155
图 10-34：开启触摸反馈 155

图 11-1：ModBus 指令文件.....162
图 11-2：ModBus 指令编辑界面入口.....163
图 11-3：ModBus 指令编辑界面.....163
图 11-4：设置从机地址.....175
图 11-5：LT165 选择 modbus 从机模式.....175
图 11-6：设置主机模式.....176
图 11-7：LT165 选择 modbus 从机模式.....176
图 11-8：Operation0x01 模式通信指令.....177
图 11-9：示例图页面.....177
图 11-10：Operation0x02 模式通信指令.....178
图 11-11：多变量操作控件对地址赋值.....178
图 11-12：Operation0x03 模式通信指令.....179
图 11-13：指令发送函数所在位置.....179
图 11-14：标志位数组.....179
图 11-15：触控判断函数所在位置.....180
图 11-16：对应元素置 1.....180
图 11-17：添加串口返回值.....180
图 12-1：Flash_RW-II 软件主界面.....184
图 12-2：地址输入框.....185
图 12-3：选择串口通信模式.....186
图 12-4：串口通信模式设置.....186
图 12-5：USB 转 TTL 模块.....187
图 12-6：选择串口.....187
图 12-7：Flash_RW-II 软件更新 UI 工程.....188
图 13-1：横竖屏 UI 页面页码要求.....189
图 13-2：竖屏 UI.....190
图 13-3：横屏 UI.....190
图 13-4：下载软件包.....191
图 13-5：打开软件工具.....191
图 13-6：UI_BinFiles_Merge 软件界面.....192
图 13-7：添加 file1.....192
图 13-8：添加 file2.....193
图 13-9：合成文件.....193
图 13-10：合成成功.....194
图 13-11：文件说明.....194
图 13-12：合并文件的结构说明.....194
图 13-13：填入 second_UI_add.....195
图 14-1：控件叠加.....196
图 15-1：删除旧工程原有配置文件.....198
图 15-2：Plugin 文件夹.....199
图 15-3：Plugin 文件夹内容.....199

图 15-4：旋转参数设置.....200
图 15-5：顺时针旋转 270°.....200
图 15-6：Gif 转 bin.....201
图 15-7：取消选择.....203
图 15-8：删除参数内容.....204
图 15-9：底图删除成功.....204
图 15-10：软件内字体显示不全.....206
图 15-11：正常显示.....206
图 15-12：进入属性界面.....207
图 15-13：更改 DPI 设置.....207
图 15-14：更改属性.....208
图 15-15：确认修改属性.....208
图 15-16：素材库.....209
图 16-1：导入要烧录的 Bootloder 文件.....213
图 16-2：连接烧录板.....214
图 16-3：进行烧录与烧录完成.....214
图 16-4：LT165 开发板示意图.....215
图 16-5：USB 转 TTL 模块.....215
图 16-6：串口升级软件界面说明.....216
图 16-7：串口升级软件界面说明.....216
图 16-8：点击进入主程序.....217
图 16-9：添加 Flash ID.....217
图 16-10：IC 选择.....218
图 16-11：LT165A 串口屏的开发演示板（Demo Kit）.....222
图 16-12：开发流程（1）.....223
图 16-13：开发流程（2）.....224
图 16-14：开发流程（3）.....225
图 16-15：在 UI 工程找到 OTA 升级地址管理.....228
图 16-16：勾选 OTA Update Addr.....229
图 16-17：设置升级预留空间.....229
图 16-18：差分文件总结构.....230
图 16-19：差分升级文件的 32byte 数据结构.....230
图 16-20：每块 block 的 16byte 数据结构.....231
图 16-21：在差分对比软件中导入 bin 文件.....231
图 16-22：差分对比软件比较出的差异信息.....232
图 16-23：生成后缀名为.diff 的数据文件.....233
图 16-24：导入.diff 文件.....233
图 16-25：导入.diff 文件.....234
图 16-26：点击 Update Flash 进行烧录.....234

表目录

表 4-1：素材存放位置表 35
表 6-1：数字键盘输入限制举例说明表 68
表 6-2：键盘数据位数设置建议表 68
表 6-3：数字键盘键值键码对应表 69
表 7-1：时间寄存器表 105
表 7-2：修改时间确认值对应表 105
表 9-1：模拟器发数据参考表 119
表 10-1：指令格式总表 142
表 10-2：串口写数据指令及其反馈指令格式表 143
表 10-3：串口读数据指令与返回指令及其反馈指令 154
表 10-4：触摸反馈指令格式表 155
表 10-5：触摸反馈控件表 156
表 10-6：读写数据指令的 CRC 计算 159
表 10-7：读返回指令的 CRC 计算 159
表 10-8：控件串口返回指令的 CRC 计算 160
表 10-9：控件是否支持键码触发 161
表 11-1：ModBus 指令结构 164
表 11-2：指令功能码 165
表 11-3：指令执行模式参数对应表 165
表 11-4：主机读保存寄存器指令 167
表 11-5：主机读保存寄存器实际通信指令 167
表 11-6：主机读输入寄存器指令 168
表 11-7：主机读输入寄存器实际通信指令 168
表 11-8：主机写单个保存寄存器指令 169
表 11-9：主机写单个保存寄存器实际通信指令 169
表 11-10：主机写多个保持寄存器指令 170
表 11-11：主机写多个保持寄存器实际通信指令 170
表 11-12：主机读线圈状态指令 171
表 11-13：主机读线圈状态实际通信指令 171
表 11-14：主机读离散输入指令 172
表 11-15：主机读离散输入实际通信指令 172
表 11-16：主机写单个线圈指令 173
表 11-17：主机写单个线圈实际通信指令 173
表 11-18：主机写多个线圈指令 174
表 11-19：主机写多个线圈实际通信指令 174
表 12-1：读写 Flash 指令表 181
表 12-2：CRC 校验反馈表 182
表 12-3：通信指令表 183
表 15-1：数据类型说明表 202

表 15-2：命名时禁止输入的符号 209
表 15-3：RGB888 数据结构 210
表 15-4：RGB565 数据结构 210
表 15-5：αRGB4444 数据结构 210
表 15-6：αRGB8565 数据结构 210
表 16-1：各芯片升级方式 212
表 16-2：IC 类型及其参数限制表 218
表 16-3：串口屏单页相应控件最大支持数量表 220
表 16-4：不同 IC 的变量地址范围及寄存器地址说明表 221
表 16-5：串口通信与指令差异 226
表 16-6：第二代串口屏可实现的效果 227

1. 概述

UI_Editor-Lite 是乐升半导体推出的第二代串口屏开发软件，搭配 LT165 系列的二代串口屏程序实现串口屏显示功能，本手册是描述此开发软件的使用说明。

1.1. TFT 串口屏的软硬件架构

TFT 串口屏就是在 TFT 显示模块上增加 MCU（有的还需要 TFT 控制器），其中 MCU 负责接收主控端送来的串口指令（串口指令详情查看串口通信指令）。主控端依据这些定义好的通讯协议，发送指令给串口屏，串口屏显示出由 UI_Editor-Lite 做好的工程，这样**主控端不需要为了繁琐的图片显示去编写和运行复杂的程序。**
TFT 串口屏与主控端主要是通过 Uart（SCI）接口来通信，以 LT165 为例。如果要达到较远距离的通信效果，通常需要加上 RS232 或是 RS485 的专用驱动芯片。

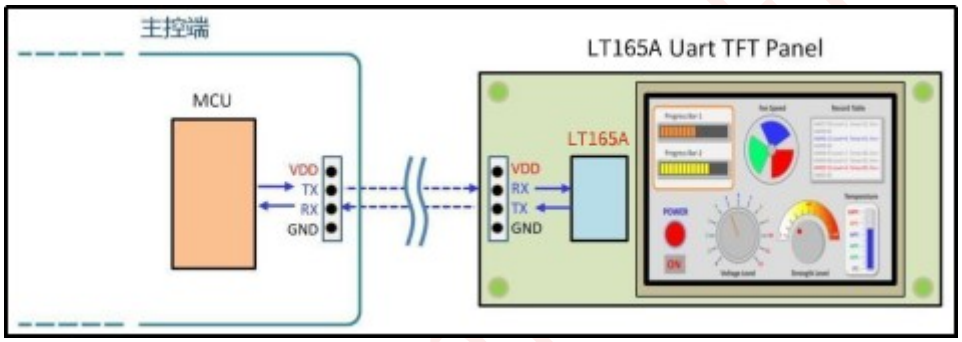


图 1-1：主控端 MCU 的 Uart 串口对 TFT 串口屏进行通信

在使用乐升半导体二代的串口屏方案之前必须用 UI_Editor-Lite 软件对 TFT 串口屏进行设置及显示功能的开发，UI_Editor-Lite 最终会将使用到图片、文字、动画及显示流程等关联信息生成 Bin 文件，是专用的 SPI Flash 烧录器将这些 Bin 文件烧录到串口屏上外置的 SPI Flash 内，然后主控端的 MCU 通过 Uart 串口对 TFT 串口屏进行通信，或是通过 PC 由 USB 转串口接口对串口屏进行通信，做串口屏显示画面的前期验证。

1.2. UI_Editor-Lite 软件设置与显示功能开发的基本流程

使用 UI_Editor-Lite 来开发一个新的工程，一般来说包含以下五个步骤：

1、准备素材，详情可参考素材格式要求；

2、新建工程，详情可参考新建工程步骤；

3、设计 UI 页面，UI 页面涉及到的控件的使用详情可参考控件使用；

4、编译模拟，UI_Editor-Lite 配套的模拟器（UI_Emulator-II）使用详情可参考二代串口屏模拟器（UI_Emulator-II）使用说明；

5、烧录，详情可参考附录 1 烧录。

注： UI_Editor-Lite 软件界面中英切换功能请查看 Help 章节的 **Language** 参数说明。

2. UI_Editor-Lite 软件的安装及附带资料说明

2.1. 下载压缩包

UI_Editor-Lite 软件资料包可在乐升半导体官网上下载，具体下载步骤如下：点击以下乐升半导体公司官网，在首页导航栏中点击“下载专区”。

图 2-1：下载专区

点击选择“串口屏开发软件/教学视频”，点击“UI 编辑/模拟软件”，点击下载软件安装包即可，下图的软件版本仅供参考。

图 2-2：选择版本

下载完成会得到一个压缩包，UI_Editor-Lite 软件及附带资料存放于此压缩包中，如下图。V1.00 表示 1.00 版本，下图仅供参考，具体版本信息以用户手里的软件安装包为准，有可能会存在安装包压缩包后缀与下图不相同的情况，**以名称和网站上的安装包版本为准。**

图 2-3：UI_Editor-Lite 压缩包

2.2. 压缩包解压缩

将压缩包存放至用户自己定义的文件夹中，选中压缩包单击右键，如下图，选择解压到当前文件夹或者是解压到与压缩包同名的文件夹中。解压软件取决于用户电脑内安装的软件，压缩包无密码。

图 2-4：解压软件压缩包

压缩包解压缩完成后，出现一个与压缩包同名的文件夹。

图 2-5：解压缩完成

2.3. UI_Editor-Lite 压缩包内附带资料说明

点击进入文件夹后，内部资料如下图所示：

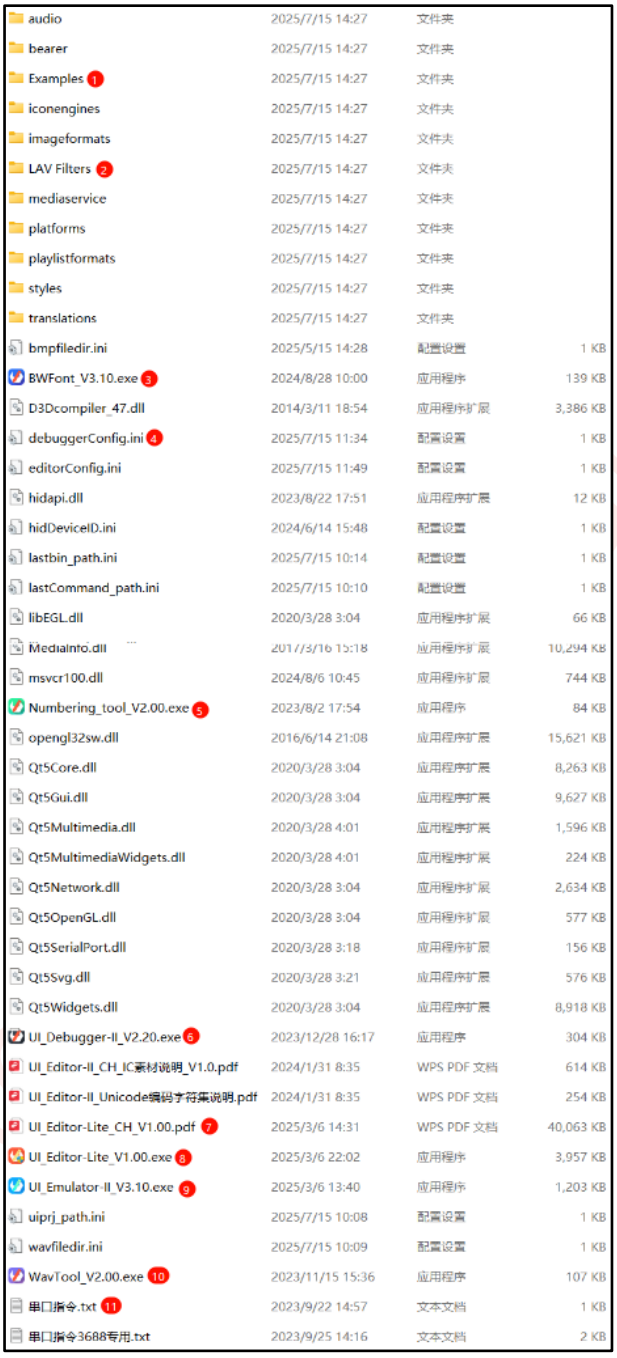


图 2-6：UI_Editor-Lite 压缩包内容（1）

1、**工程例子：**Examples 文件夹，内部包含三个 UI_Editor-Lite 工程，一个是正常横屏的 800*480 的全功能演示工程，另一个是顺时针旋转 270°的 480*800 的标准工程，还有一个是 800*480 的微波炉_多国语言工程。

2、**LAV Filters：**视频解码器文件夹，存放模拟器所需要的视频解码器的安装包。

3、**字库取模软件：**BWFont_Vx.x.exe，用于创造不同字号不同编码类型的字库，详细用法请参考字库取模工具使用说明。

4、配置文件：UI-Editor 的配置文件，非必要不要进行修改。

5、图片素材编号工具：Numbering_Vx.x.exe，用于给图片素材快速编号，详细用法请参考图片编号工具使用说明。

6、串口通信调试工具：UI_Debugger-II_Vx.xx.exe，用于电脑与串口屏的指令通信，详细用法请参考串口调试工具（UI_Debugger-II）使用说明和串口通信指令参考。

7、软件说明书：UI_Editor-Lite_CH_Vx.xx.pdf，供用户查阅参考。版本以用户手里的为准。

8、UI_Editor-Lite 主程序：本软件主程序，使用时请右键选择“以管理员身份运行”运行。

9、UI_Emulator-II：UI_Editor-II_Vx.xx 的配套模拟器，可在电脑上模拟工程的运行环境，详情可查看二代串口屏模拟器（UI_Emulator-II）使用说明。

10、音频转换工具：WavTool_Vxx.exe，可将 Wav 文件转换成 bin 文件，详细用法请参考音频转换工具使用说明。

11、串口指令列表：串口指令文档.txt，内部含有串口通信的指令，可以在串口通信调试工具里导入该指令文件实现快速编写指令。

注：

1、所有工具的版本以安装包内的为准。

2、进行字库制作、模拟器运行工程时请使用与 UI_Editor-Lite 同一压缩包下的工具。

2.4. 启动软件

点击进入 UI_Editor-Lite_Vx.xx 文件夹。

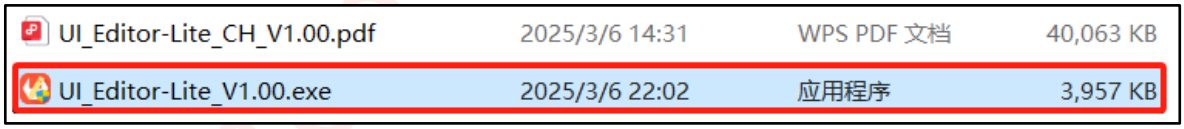


图 2-7：UI_Editor-Lite 压缩包内容（2）

选中 **UI_Editor-Lite_Vx.xx.exe** 文件点击右键，选择以“管理员身份运行”，打开软件。建议运行环境为 windows10 及以上。

图 2-8：运行软件

2.5. 检查软件

打开软件后，在主界面点出 File 界面再点击 Project Setting，进入设置界面。

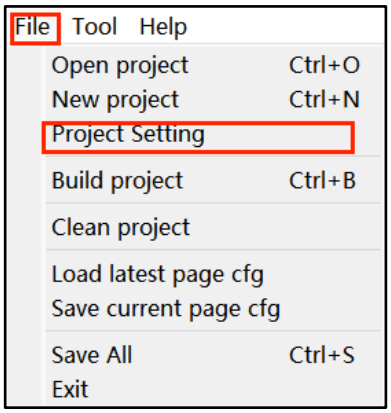


图 2-9：进入设置界面

检查文字显示是否出现下图类似的显示错误问题，如果显示出现错误，请前往电脑屏幕分辨率导致软件字体显示不全的说明的说明查看解决方法，如果显示是正常的，请跟着步骤继续往下看。

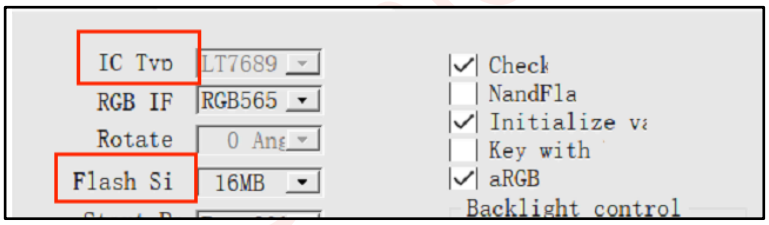


图 2-10：错误显示

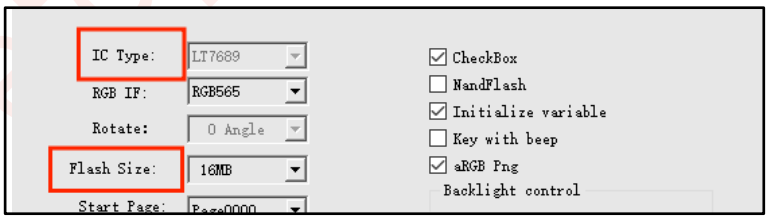


图 2-11：正常显示

2.6. 创建快捷方式

退出软件后，选中软件单击右键，如下图，选择创建快捷方式。

图 2-12：创建快捷方式

快捷方式创建成功。

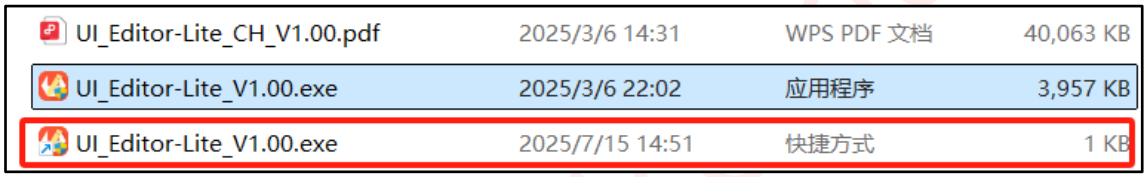


图 2-13：快捷方式创建成功

选中快捷方式单击右键，选择剪切或者复制。

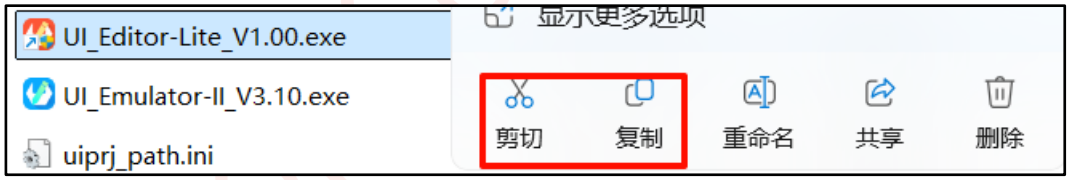


图 2-14：剪切或复制快捷方式

然后回到桌面，鼠标指针在桌面空白位置点击右键，选择粘贴。快捷方式就粘贴到了桌面上，左键双击图标即可快捷进入软件，无需去文件夹内部寻找软件本体。

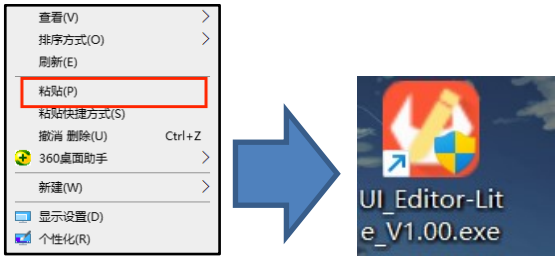


图 2-15：粘贴快捷方式

3. 软件介绍

3.1. 软件界面介绍

执行 UI_Editor-Lite 的主程序后会进入到以下软件界面，为了方便说明，这里展示一个已导入工程的软件界面。如下图，软件界面大体上分为六个部分，包括标注 1 区域的控件栏、标注 2 区域的页面栏、标注 3 区域的工程编辑设计视窗、标注 4 区域的状态栏、标注 5 区域的控件列表以及标注 6 区域的控件参数设置区域。

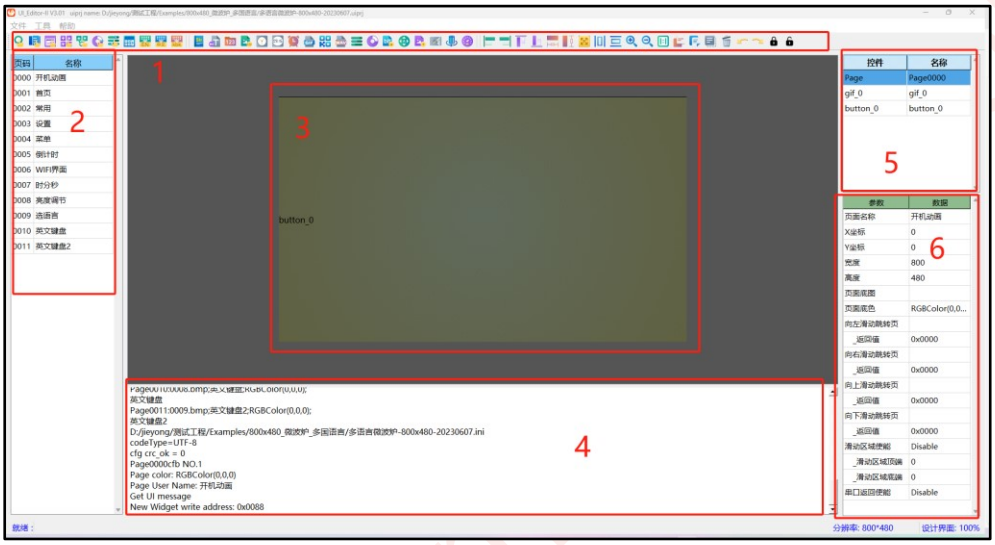


图 3-1：软件界面

1、**控件栏：**如下图所示：首先是带触摸属性的控件，接下来是显示控件、页面快捷布局控件区域。各个控件的功能详情请查看控件使用。

图 3-2：控件栏

2、页面栏：列表左栏是页 ID 号（不可改变），右栏是页名称（可自定义）。页面操作请查看页面操作说明。

3、工程编辑设计视窗：视窗内的页面底图部分为可编辑设计区域，可在底图上添加控件，灰色区域不可编辑。

4、状态栏：在此区域可查看操作记录，编译工程时可查看 UartTFT-II_Flash.bin 文件是否完成。

5、控件列表：此列表显示当前页面的所有的控件名称及其编号。用户可以在列表里点击控件名称快速地在设计区域内找到对应控件。

6、控件参数设置区域：在此区域可以设置控件的参数，包括但不限于名称、地址、在底图上的坐标位置等参数。

3.2. 软件功能栏

如下图，软件功能栏主要包括 File、Tool 和 Help 三个部分。

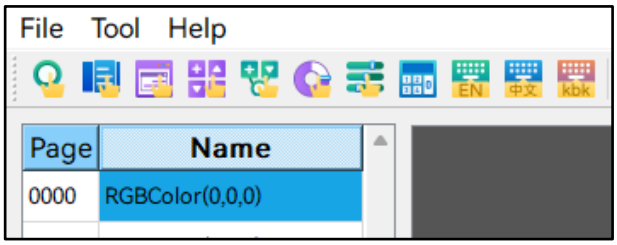


图 3-3：软件功能栏

3.2.1. File 功能

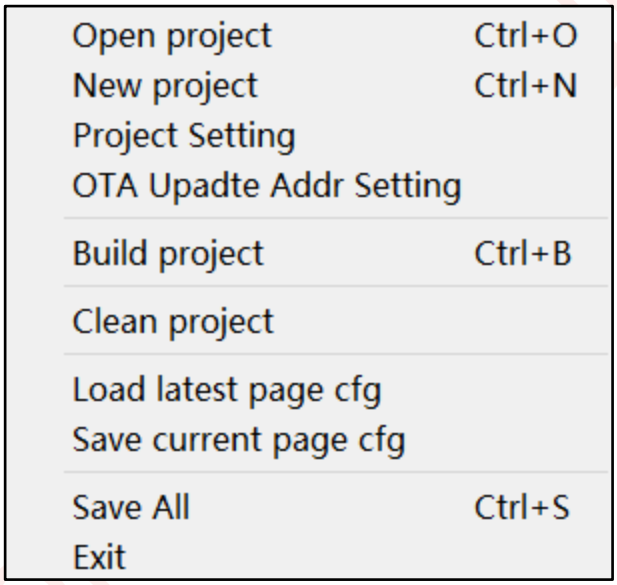


图 3-4：file 功能

1. **Open project**：打开已有工程。
2. **New project**：新建工程入口。
3. **Project Setting**：工程设置页，具体说明可查看工程设置。
4. **OTA Update Addr Setting**：OTA 升级地址管理，具体可查看附录 7 OTA 差分升级说明。
5. **Build project**：编译工程，产生 UartTFT-II_Flash.bin 文件。
6. **Clean project**：删除除字库和音频的 bin 文件之外的所有 bin 文件。
7. **Load latest page cfg**：打开最近的 cfg 文件。
8. **Save current page cfg**：保存当前页控件参数到 cfg 文件。
9. **Save all**：保存所有文件的所有改动。

10. Exit：退出程序。

注：cfg 文件是每一页的最终配置文件，每一页都只有一个。**cfg 文件触发保存的条件有：**

1、编译工程（全体页）。

2、保存此页控件参数至 cfg 文件（当前页）。

3、保存工程。

4、离开工程时会弹出选项选择是否保存（全体页），包括新建工程、打开其他工程、关闭当前工程。

3.2.2. Tool 功能

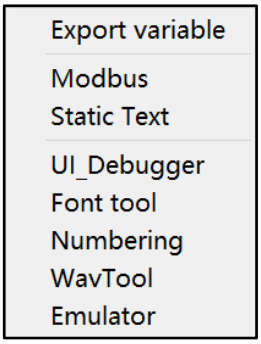


图 3-5：工具列表

1. **Export variable**：导出变量表，变量表存放于工程文件夹，与素材文件夹同级，包含 DisplayWidget.csv 和 touchWidget.csv。

   (1) DisplayWidget.csv：显示控件的参数列表。包含显示控件的地址、长度、所在页 ID 和名称等。

   (2) touchWidget.csv：触摸控件的参数列表。包含触摸控件的地址、长度、所在页 ID 和名称和部分关键参数。

图 3-6：变量表说明

2. **Modbus**：点击弹出 Modbus 主机指令编辑窗口，具体说明可查看 ModBus 通信功能。
3. **UI_Debugger**：串口调试工具，点击后打开串口调试软件，详情参考串口调试工具（UI_Debugger-II）使用说明。
4. **Font tool**：字库制作，点击后打开字库制作软件，详情参考字库取模工具使用说明。
5. **Numbering**：图片素材编号，点击后打开图片素材编号软件，详情参考图片编号工具使用说明。
6. **WavTool**：音频 bin 文件制作，点击后打开音频 bin 文件制作软件，可以将 Wav 文件转成 bin 文件，详情参考音频转换工具使用说明。
7. **Emulator**：开启模拟器，点击后开启模拟器并自动把当前工程导入模拟器运行，详情参考二代串口屏模拟器（UI_Emulator-II）使用说明。
8. **Static Text**：静态文本，使用静态文本控件的配置，详情参考静态文本使用说明。

3.2.3. Help 功能

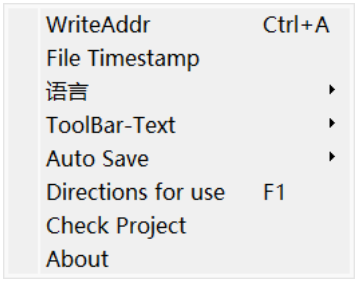


图 3-7：Help 功能

1. **WriteAddr**：包含以下功能：

(1) Paste Auto Address：勾选后，拷贝粘贴控件时，新生成的复制体控件的 writeAddr 会自动偏移；若不勾选，则新生成的复制体控件与原控件地址一致。

(2) New WriteAddr：设置下个控件的起始地址，后续增加的控件都会以此偏移。

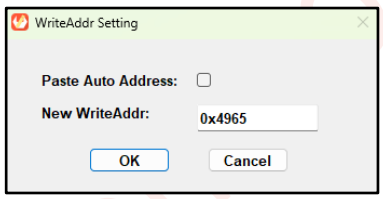


图 3-8：writeAddr 设置

2. **File Timestamp**：

(1) 未勾选，编译工程时，将当前时间编译至 UartTFT-II_Flash.bin 文件中。

(2) 勾选后，编译工程时，将显示框显示的时间编译至 UartTFT-II_Flash.bin 文件中。

注：显示框显示时间为上一次编译进 UartTFT-II_Flash.bin 文件中的时间。

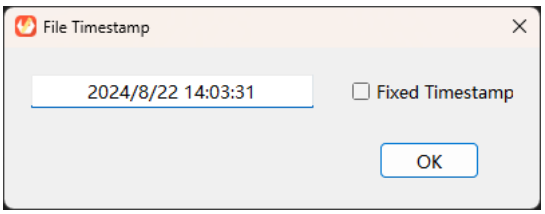


图 3-9：File Timestamp 设置

3. **语言**：语言切换功能，可选择将软件切换中文或英文模式。

4. **ToolBar-Text**：打开后控件栏的图标会在下方显示控件名称。

5. **Auto Save**：自动保存功能，打开后，每 5 秒保存一次工作状态。
6. **Directions for use** ：打开软件说明书。
7. **Check Project**：检查工程功能，检查素材文件命名是否规范、是否存在重复，检查控件是否超出可显示范围。
8. **About**：包含软件信息。

4. 新建工程

4.1. 工程文件夹内容说明

工程文件夹内的素材文件夹如下图所示，**点击新建工程时，下图的文件夹会自动新建**；如果用户想要自己新建工程素材文件夹，需要严格按照图中内容去设置文件夹名称，或者是直接复制 Examples 文件夹内基础工程的 7 个子文件夹，然后把其中的内容都清理掉、替换成用户自己的素材；

如果是以旧工程素材新建新工程，可查看以旧工程素材创建新工程说明。

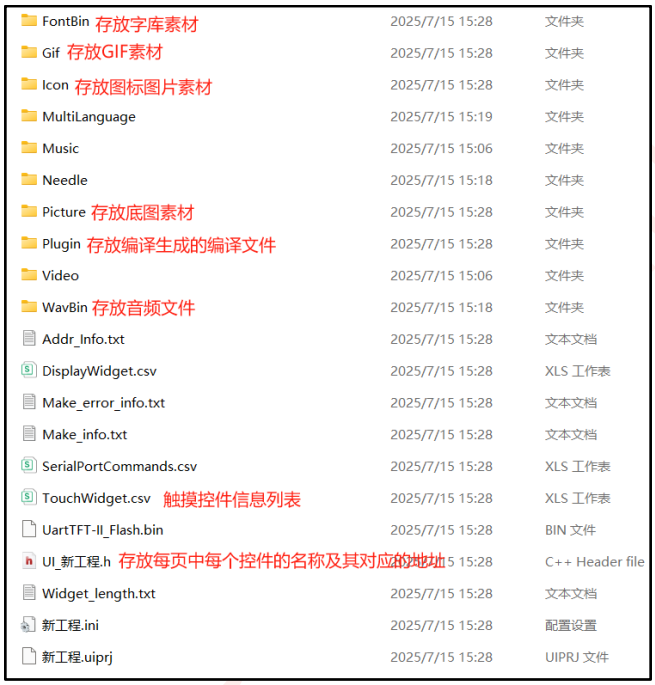


图 4-1：工程文件夹说明

4.1.1. 素材格式要求

4.1.1.1. Picture 文件夹

存放内容：页面底图、弹窗底图、键盘底图。

素材格式：BMP、JPG

素材命名：以 4 位数字给图片编排序号，按 0000 ~ 9999 排列，固定命名为“**xxxx**”或者是“**xxxx_注释**”。

图 4-2：底图命名

注:

1、序号不可以相同。

2、序号可以留空，留空的位置在新建工程时也会自动生成页，不过该页没有底图，需要手动添加。

3、新建工程时生成多少页取决于图片的最大序号，后续想增加页可以点击页列表右键并单击 newpage。

4、**PNG 格式的图无法作为底图，需要 jpg 或者 bmp 格式。**

5、若图片素材较多，可通过图片编号工具进行快速编号，详情请查看图片编号工具使用说明。

4.1.1.2. Icon 文件夹

存放内容：小图标素材、图片数字素材、滑动菜单素材、滑条和进度条素材、模拟时钟素材等

素材格式：BMP、JPG、（只有数字图片控件的素材推荐用 PNG 格式的图片，其他控件的素材不推荐 PNG 格式素材）。

素材命名：以 4 位数字给图片编排序号，按 0000 ~ 9999 排列，固定命名为“**xxxx**”或者是“**xxxx_注释**”。

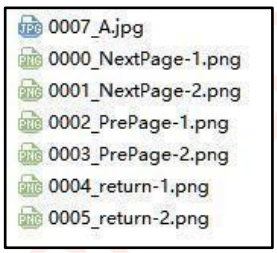


图 4-3：Icon 命名

注:

1、关于同组 Icon 宽高的说明，详情可查看同组的 Icon 宽高说明。

4.1.1.3. FontBin 文件夹

存放内容：字库 bin 文件

素材格式：bin 文件

素材命名:以 2 位数字给字库文件编排序号,按 00 ~ 35 排列,最多放置 36 个字库文件。固定格式为“**xx_Font-注释** ”。

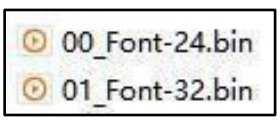


图 4-4：字库命名

注: 用户需要自行生成字库时务必使用软件安装包内自带的字库生成软件 **BWFont_Vx.xx.exe**。详情请参考字库取模工具使用说明。

4.1.1.4. Gif 文件夹

存放内容：Gif 文件

素材格式：Gif（推荐 bmp 格式）

素材命名：以 4 位数字给动图文件编排序号，按 0000 ~ 9999 排列，固定格式为“**xxxx**”或者是“**xxxx_注释**”。文件后缀需设为 .gif ，不能含有大写英文字母。

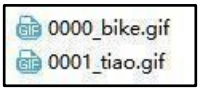


图 4-5：Gif 命名

4.1.1.5. WavBin 文件夹

存放内容：音频 bin 文件

素材格式：bin 文件

素材命名：以 4 位数字给音频 bin 文件编排序号，按 0000 ~ 0099 排列，最多放置 99 个音频文件。固定格式为“**00xx**_Wav”或者“**00xx_Wav_注释**”。

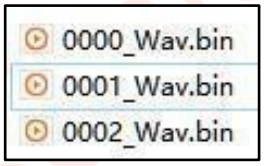


图 4-6：音频命名

4.1.1.6. Needle 文件夹

存放内容：needle 控件生成的指针旋转文件

注：文件内容由软件自动生成，不需要人为添加素材。

4.1.1.7. 其他

1、Plugin 文件夹存放编译信息文件，不存放用户文件。

2、对于同一种素材，其编号可以留空但不能出现重复，0000.png 和 0000.bmp 虽然在电脑的同一个文件夹里是合法的，但是在命名编号时却是违法的，会导致图片调用错乱。

3、所有的素材、控件、页名称和工程名称等的命名都不能存在下列符号，小数点（英文句号）只能作为素材和格式后缀的分隔符存在一个，不能存在两个及以上。若素材命名不正确，则无法导入该素材。

\ / : * ? " < > | . ,（英文输入法的逗号）

5、乐升半导体为用户提供一个素材库，详情参考素材库说明。

4.1.1.8. 素材存放位置

素材的存放位置如下表所示，注意的是除了音频文件，工程内没有调用到的素材均不会被打包到 UartTFT-II_Flash.bin。而音频文件无论是否被调用均会打包。关于 UartTFT-II_Flash.bin 与素材的关系说明详情，可参考 UartTFT-II_Flash.bin 大小与素材关系说明。

表 4-1：素材存放位置表

素材存放结构图							
FontBin	Gif	Icon	Picture	Plugin	WavBin	Video	Needle
字库素材	动图素材	小图标	主图素材	软件生成	音频素材	视频素材	软件生成
		数字图片	键盘素材				
			弹窗素材				

4.1.2. UI_projectname.h 文件

存放内容：每一页每一个控件的名称及其对应的地址生成的宏。

注：

1、projectname 指的是每个 UI 工程的工程名。

2、UI_Editor 默认生成的宏名组成是由控件类型+PageID+控件默认名称组成，如果用户需要在生成的宏名中加上自定义的控件名称让宏名变成控件类型+PageID+控件默认名称+用户自定义名称的组成形式，则可以在**关掉 UI_Editor 软件后**，在 UI_Editor 的软件目录（可参考 UI_Editor-Lite 软件的安装及附带资料说明）下打开 **editorConfig.ini（此文件非必要不要修改）**这个文件，然后将 Header_file_index 这个参数的值改成 1 如图所示。

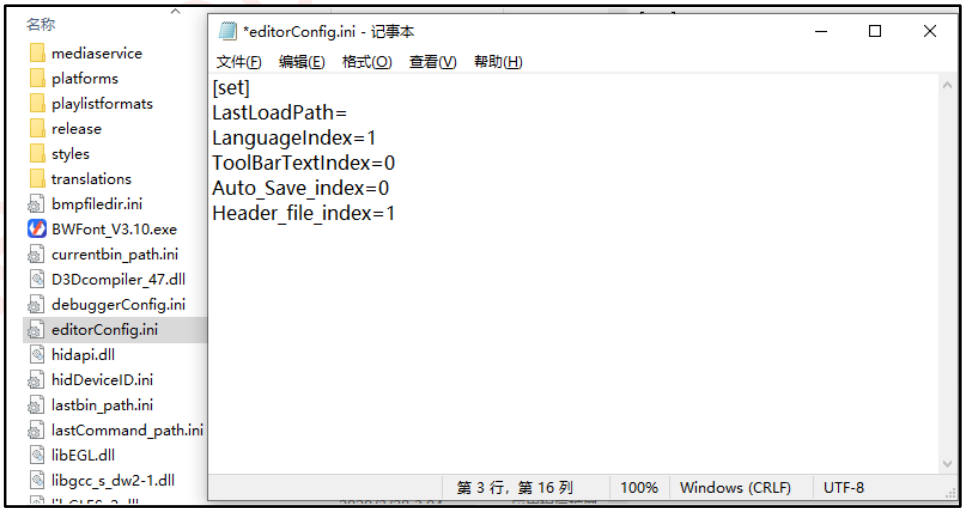


图 4-7：editorConfig.ini 配置文件

3、UI_projectname.h 文件不仅清晰展示了 UI 工程中每一页每一个控件的名称及其对应的地址，还可以将 UI_projectname.h 文件添加到代码的 user 文件夹中，**需要注意的是如果工程名是中文则需要将 UI_project.h**

这个文件手动命名成全英文才能放入 user 文件夹中调用。然后包含 UI_projectname.h 这个头文件，这样就可以在代码中通过控件名称相关的宏来使用对应的地址，以保持控件和地址的一致性。

4、**需要注意如果已经将参数 Header_file_index 改为 1 且要用到 UI_projectname.h 这个文件时控件的自定义命名不能命名成中文**，因为 UI_projectname.h 这个文件生成的宏名包含了用户自定义的控件名称，如果有中文就不能在程序中调用。如果没有修改配置文件中的 Header_file_index 参数，则它的值默认是 0，这样生成的 UI_projectname.h 文件的宏名就不包括用户给控件的自定义命名部分，所以即使此时控件的自定义命名为中文，也不影响 UI_projectname.h 文件的宏在程序中调用。

4.2. 工程设置

新建工程前，需要设置好屏参以及各项参数，点击 Project Setting 即可来到设置界面，如下图所示。

图 4-8：工程界面设置

工程设置参数说明如下：

1、General：

(1) MCU Type：芯片型号选择，需与实际板子对应。不同的 IC 型号具有不同的限制，详情参考附录 2 UI_Editor-Lite 支持的 IC 及其参数限制。

(2) Flash Type：外部 SPI Flash 类型选择，根据 Flash 芯片实际类型选择，若为灰色则不需要选择。

(3) Flash Size：外部 SPI Flash 的容量大小选择，根据 Flash 芯片实际大小选择。

(4) Rotate：旋转角度设置。详情参考画面旋转操作设置说明。

(5) Num of Language：使用多国语言功能时可设置语言种类，默认为 1。多国语言功能请查看多国语言功能说明。

2、Backlight Control：背光控制。

(1) Auto Dimming：勾选后启用自动背光设置，未勾选时则屏幕常亮。

(2) **Normal（10 ~ 63）**：常亮时的亮度等级，硬件支持 0 ~ 63，这里限制输入 10 ~ 63 内的亮度等级。

(3) **Hold time (s)**：熄屏时间，屏幕无操作时自动熄屏所需的时间，单位秒，可设置的范围为 1 ~ 65535。

(4) **Sleep（0 ~ 63）**：熄屏后屏幕的亮度等级。熄屏后再次点击屏幕即可唤醒。

3、Application：

(1) **RGB Format**：图片数据格式，根据 IC 型号支持与否以及设计要求选择。

(2) **Startup Page**：开机页面选择。

(3) **Needle Data Type**：选择是否压缩 Needle 文件夹内容，注意 LT165 不支持该功能。

(4) **Gesture Data**：滑动手势判断的距离，单位是像素。手指滑动距离超过 50 像素时，该操作才会被判定为滑动，主要用于判断滑动跳页。

(5) **Volume（0~16）**：初始音量级别设置，16 表示最大音量，0 为静音。

(6) **Initialize Variable**：初始化使能，勾选后 UI_Editor-Lite 内控件设置的默认值会在串口屏上电时展示；若不勾选，则 Flash.bin 烧录到串口屏后，上位机控件设置的默认值不会在上电时展示。默认值包括控件参数 defaultText，defaultValue 等参数。

(7) **With GBKCode**：是否把 GBK 编码添加到 UartTFT-II_Flash.bin 中，使用 GBK 输入法时需要勾选。

(8) **Key with beep**：蜂鸣器使能，勾选后启用蜂鸣器，触控时会响应。

(9) **aRGB Png**：勾选为硬件 PNG，数据为 aRGB4444-16bits，不勾选为软件 PNG。

(10) **Page Image Zip**：页面图片压缩功能，LT165 不支持该功能。

4、Communication：通信参数设置。

(1) **Baudrate**：波特率选择。

(2) **Parity**：选择校验方式，默认 None。Odd：奇校验，Even：偶校验。

(3) **No reply**：勾选后串口通信不会返回 CRC 校验成功或失败的信息。

(4) **No CRC padding**：勾选后串口通信不需要 CRC 校验，同时也不会返回 CRC 校验信息。注意 No CRC 与串口调试工具的 Crc Enable 需要同步勾选或不勾选。

(5) **User defined CMD header**：用户自定义串口通信的帧头。

5、Modbus

(6) **Master Mode：勾选后该工程作为 modbus 通信的主机，不勾选则作为从机。主从机都需要相应的 Mcu_code 支持，详情参考 Modbus 工程及程序设置例程。**

(7) **Device Addr：该工程作从机时可由此设置从机地址,可设范围 0x01~0xFF，不可设置为 0x00。**

(8) **Device Num：设备数量，暂时默认即可。**

6、User Information：用户信息，无需修改。

User ID：用户 ID 号。

Version：软件的版本号。

7、TFT panel

TFT Horizontal:

X-Pixel：屏幕横向分辨率。

HBPD：根据 TFT 屏参数设置。

HFPD：根据 TFT 屏参数设置。

HSPW：根据 TFT 屏参数设置。

TFT Vertical:

Y-Pixel：屏幕纵向分辨率。

VBPD：根据 TFT 屏参数设置。

VFPD：根据 TFT 屏参数设置。

VSPW：根据 TFT 屏参数设置。

Signal Polarity:

PCLK_falling：根据 TFT 屏参数设置。

HSYNC_High：根据 TFT 屏参数设置。

VSYNC_High：根据 TFT 屏参数设置。

DE_Low：根据 TFT 屏参数设置。

BGR：勾选后遵循 BGR 模式，不勾选则是 RGB。

4.3. 新建工程步骤

如下图所示，软软件主界面 File 功能目录下的红色框为新建工程入口。这里建议读者在新建工程前，将工程文件和软件放在不同的文件夹内，这样既方便管理，也方便后续软件更新时直接删除旧版本的软件。

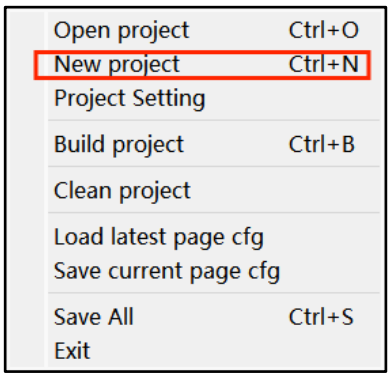


图 4-9：新建工程入口

下面介绍新建工程的步骤：

1、新建工程文件夹。按照新建工程前的素材准备及注意事项的规则设置好工程文件夹，准备好素材，如果不需要准备素材可以直接跳到步骤 2。

2、打开 UI_Editor-Lite 软件，进入设置界面，设置好芯片类型、SPI FLASH，分辨率等参数，参数详情参考工程设置，然后点击 New Project。

3、点击后，会弹出以下界面，先找前面创建的工程文件夹，点击进入工程文件夹后不进入子文件夹，然后在下方输入新工程的名字，点击保存。

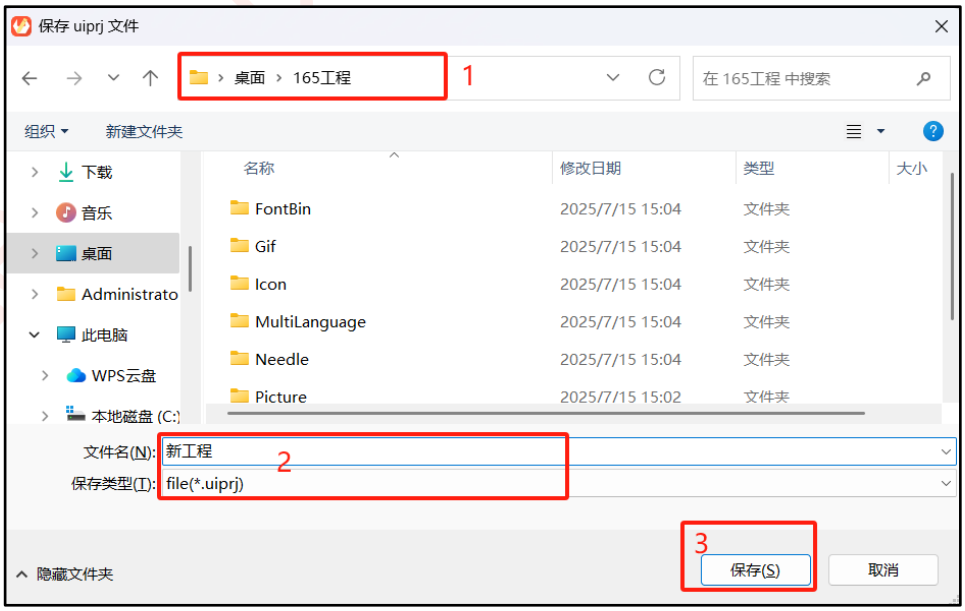


图 4-10：新工程命名

4、点击保存后，会自动跳到下图所示页面。随便选中一张素材图片，然后点击打开。若没有任何底图或者不想添加底图，可以点击取消。

图 4-11：添加素材

5、进入新工程，如果出现以下界面，说明新建工程成功。

图 4-12：进入新工程界面

5. 页面操作说明

5.1. 页面设置及参数

页面参数如下图标注 2，调出某一页的页面参数有两种方法，第一种是通过点击页列表对应页，如下图标注 1；第二种是点击当前页的底图（无控件的地方），如下图标注 3。

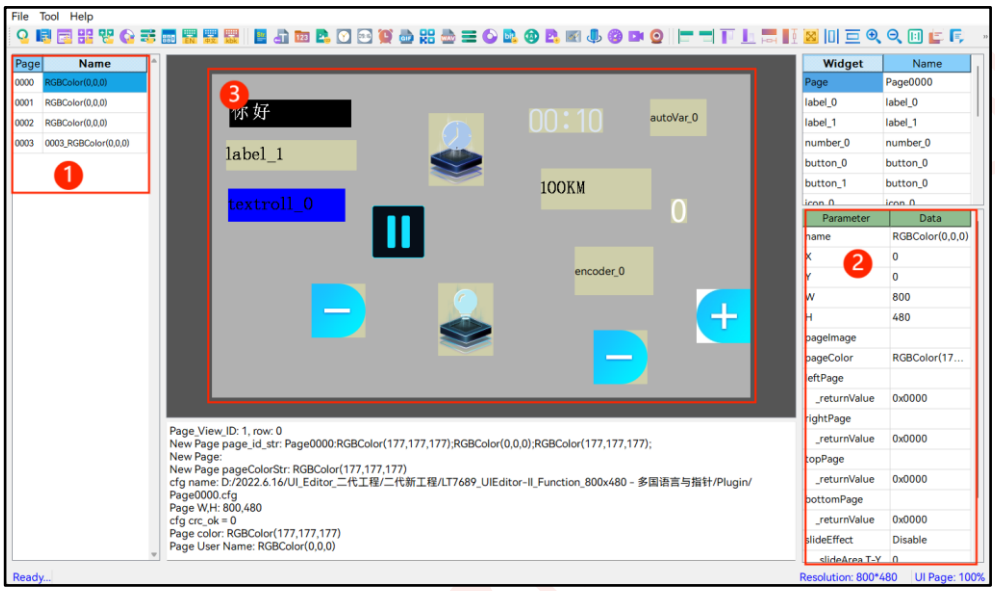


图 5-1：查看页面参数

调出页面参数后，下面来介绍各个参数的使用。

Parameter	Data
name	键盘1
X	0
Y	0
W	398
H	302
pageImage	0020.bmp
pageColor	RGBColor(0,0,0)
leftPage	
_returnValue	0x0000
rightPage	
_returnValue	0x0000
topPage	
_returnValue	0x0000
bottomPage	
_returnValue	0x0000
slideEffect	Disable
_slideArea T-Y	0
_slideArea B-Y	0
reportToHost	Disable

图 5-2：页面参数列表

1、name：页的名称，可自定义。新建工程时，默认为同编号的底图名字。

2、X 和 Y：均默认为 0，无需修改。

3、W 和 H：该页底图的宽高，无需修改。更换底图时，这两个参数会自适应。

4、pageImage：双击可更换成素材库内的其它底图。

5、pageColor：无底图时该页的颜色，仅在 PageImage 为空时，该参数生效，双击可选择颜色。若 PageImage 不为空，则需要清空

6、leftPage：在屏幕上向左滑会跳转到的页。（left 向左，right 向右，top 向上，bottom 向下）

7、_returnValue：向左（left 向左，right 向右，top 向上，bottom 向下）滑时会返回的数据。

8、slideEffect：滑动区域使能，LT165 系列不支持开启滑动使能。

9、_slideArea T-Y：滑动区域上边沿的 Y 坐标，基准点是页的左上角坐标（0，0）。

10、_slideArea B-Y：滑动区域下边沿的 Y 坐标，基准点是页的左上角坐标（0，0）。这两个参数在后面会配图说明。

11、reportToHost：串口返回使能，跳页后返回，返回固定地址 0xFFFF 及对应方向的 returnValue。详情可查看触摸反馈指令。

注：0 <= _slideArea T-Y < _slideArea B-Y <= 屏幕的 Y 分辨率。页面的参数只会对该页生效，只有屏幕处于该页上时，对应的滑动手势才能触发跳页。

5.2. 页面跳转

在 UI_Editor-Lite 里，跳转页面总共有 3 类共 5 种方法。

第一类：通过控件跳页

1、通过按键（Button）控件跳页，详情参考按键（Button）。

2、通过多变量操作（Multi-Variable Button）控件跳页，详情参考多变量操作（Multi-Variable Button）。

第二类：滑动手势跳页

1、无滑动效果的滑动跳页。

2、有滑动效果的滑动跳页。

第三类：串口通信跳页

1、向寄存器 0x7000 发送对应页码跳页，详情参考串口发送控制页面跳转。

第四类：无滑动效果的滑动跳页

slideEffect 设置为 Disable 即为无滑动效果的滑动跳页，支持**上下左右四个方向**的滑动手势。无滑动效果指的是页面不会随手指移动而移动，滑动手势触发跳页后，新页会直接覆盖旧页，无动态的显示效果。

如下图的参数设置，在该页向左滑动时，页面跳转到第 1 页，同时返回 0x0001 给主控；在该页向右滑动时，页面跳转到第 2 页，同时返回 0x0002 给主控。

leftPage	Page0001
_returnValue	0x0001
rightPage	Page0002
_returnValue	0x0002
topPage	Page0003
_returnValue	0x0003
bottomPage	Page0004
_returnValue	0x0004
slideEffect	Disable
_slideArea T-Y	0
_slideArea B-Y	0

图 5-3：无滑动效果的滑动跳页参数

5.3. 新增页面

点击页列表，点击右键弹出下图所示的弹窗，选择 NewPage。新增页不含任何内容，且只能在列表最后新增，不能自定义在某两页之间新增。

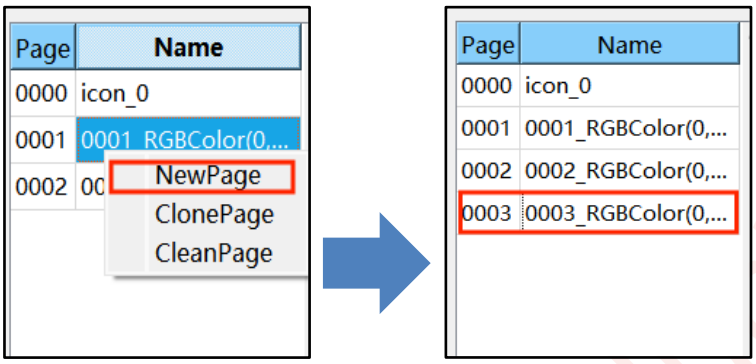


图 5-4：添加新页

5.4. 复制页面

点击页列表选中需要复制的页，再点击右键，选择 ClonePage，复制出来的页面自动添加在列表最后一位，复制页的内容与原页一模一样，使用时要注意控件地址是否冲突。

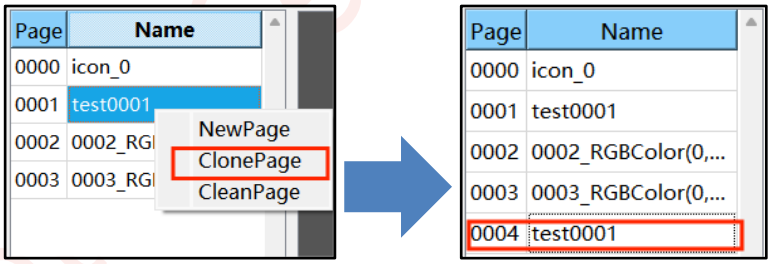


图 5-5：复制页面

5.5. 清空页面

点击页列表选中需要清空的页面，再点击右键，选择 CleanPage，会弹出一个确认弹窗，确认后该页的所有内容都会被清除，点击确认后，需要切换一下页面刷新被删除的页面，编译工程时，清空的页不会增加 UartTFT-II_Flash.bin 的大小。

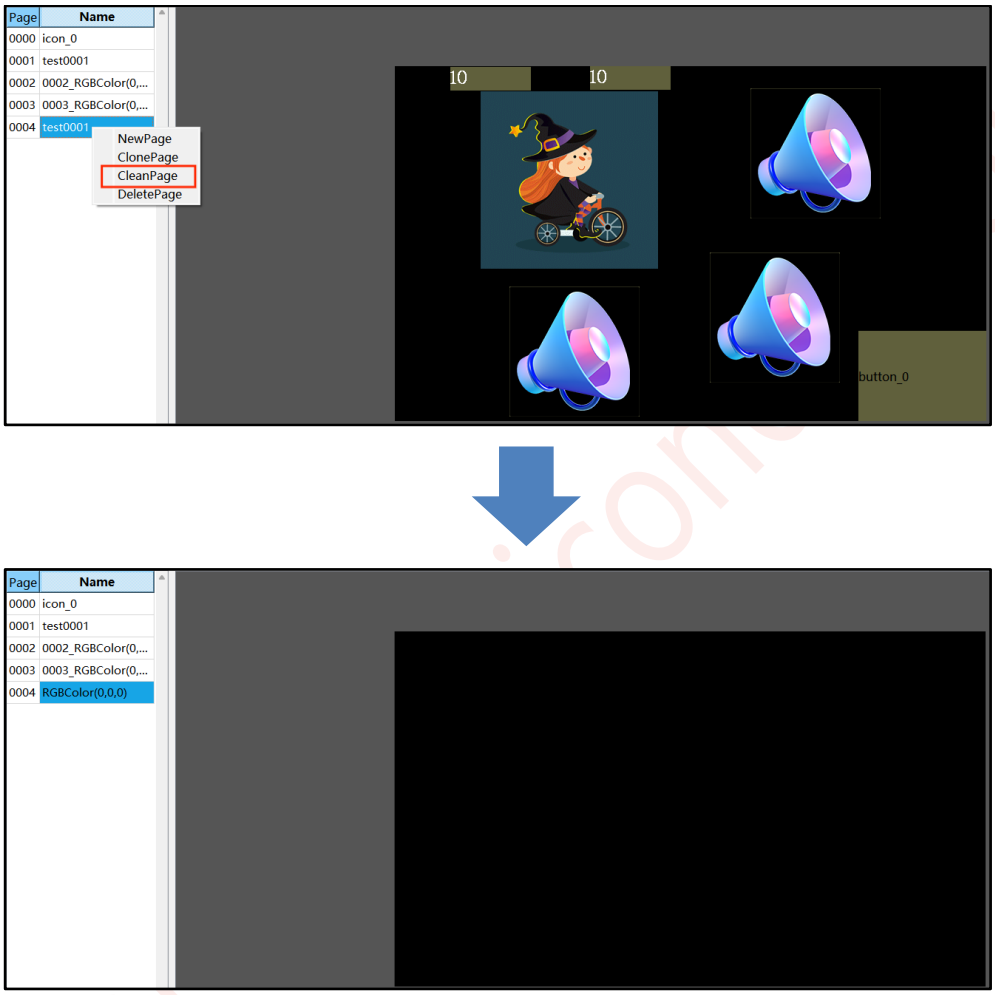


图 5-6：清除页面内容

5.6. 删除页面

对于页面列表的最后一页，可以点击 删除页面 进行删除，删除后，页面 ID 从列表中移除，但是其对应页的 cfg 文件没有被删除，新建同 ID 的页面时，已经删除页面内容会重新加载至新建页面内。

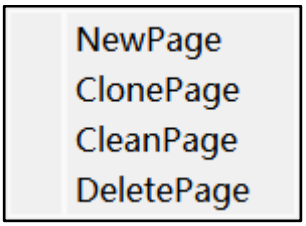


图 5-7：删除页面

5.7. 多余页处理说明

在编辑工程的过程中，可能会产生一些多余的页（没有被使用到且不在列最后的页）。由于软件无法删除页，所以对于这些多余页，用户可以直接选中该页清空页面内容，多余的页清空内容后不会占用 Flash 的空间。

5.8. 基础操作说明

5.8.1. 添加控件

1、鼠标点击需要添加的控件图标，鼠标光标由箭头变成十字光标，进入控件添加模式。

图 5-8：添加控件

2、鼠标操作十字光标在工程编辑区拖动生成控件，

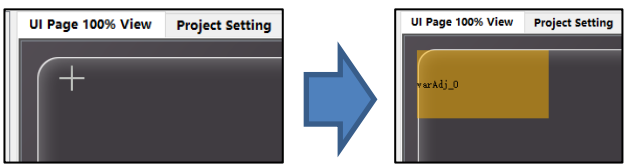


图 5-9：生成控件

3、将鼠标的十字光标停靠在工程编辑界面，然后**单击右键退出控件添加模式**，鼠标光标由十字光标转换回箭头。

5.8.2. 选中控件操作

选中控件有两种方式，一种是单击某一控件选择；另一种是批量框选控件，框选时必须把整个控件框进去。框选控件后，我们可以对选中的控件进行整体移动，或者是复制，然后粘贴到其它页。

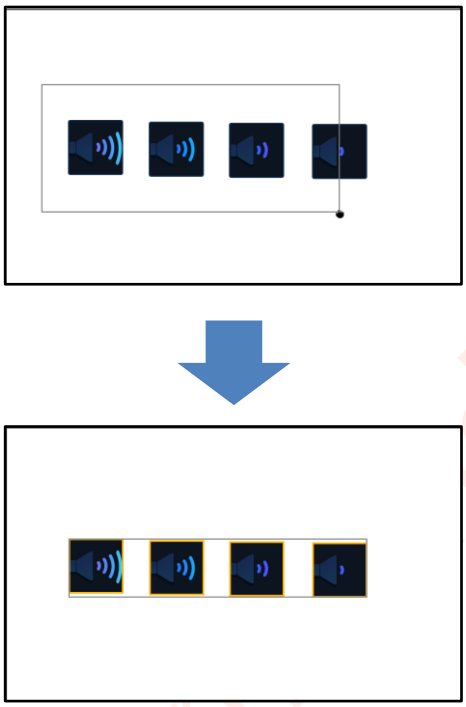


图 5-10：框选控件

5.8.3. 删除控件（Delete）

图标： 

功能：点选或者框选需要删除的控件，再点击软件上的删除图标，或者是键盘的 Delete 按键，选中的控件就会被删除。

5.8.4. 克隆控件（Widget_Clone）

图标： 

功能：该复制功能是针对同一页内单个控件的快捷复制，点击选中需要复制的控件，再点击克隆，新控件就会在旧控件旁边生成。除坐标外的所有的参数一致。

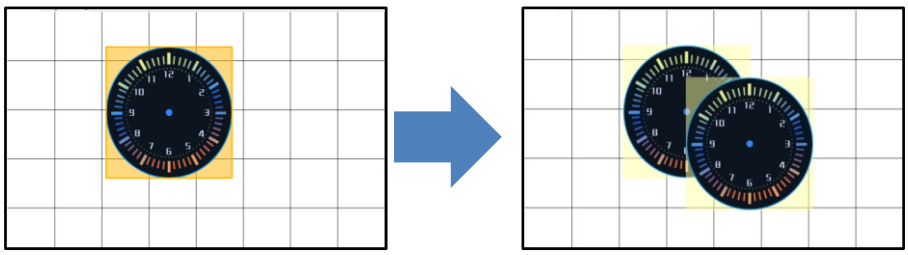


图 5-11：控件复制

5.8.5. 拷贝粘贴功能（Widget Copy and Paste）

图标： 

功能：该复制功能主要是针对多控件的拷贝粘贴，可跨页拷贝粘贴，也可以页内拷贝粘贴，生成的控件参数与原控件除了地址外其它参数一模一样（包括 XY 坐标）。此操作支持快捷键操作，Ctrl + C =复制至剪切板，Ctrl + V =粘贴至当前页的固定位置。

注：在进行跨页拷贝粘贴时，需要注意被复制控件的位置是否超出目标页的范围。

操作步骤：

1、框选需要拷贝的控件，然后快捷键 Ctrl + C 或者是点击控件拷贝图标，把框选内容加入剪切板。

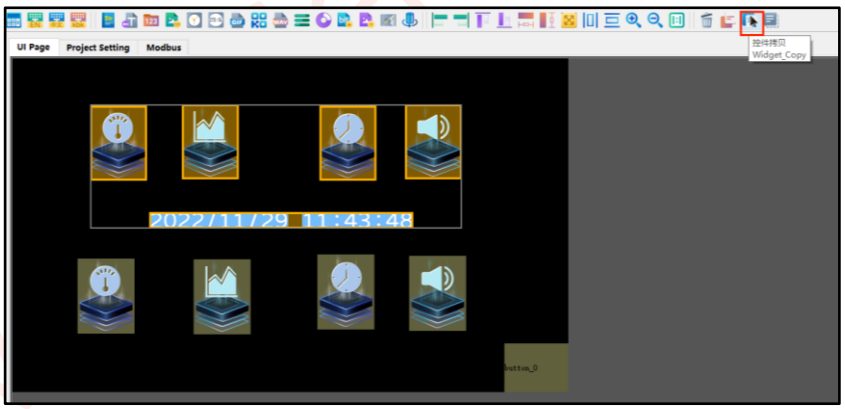


图 5-12：拷贝控件

2、前往目标页，然后快捷键 Ctrl + V 粘贴至固定位置，或者是点击控件粘贴至固定位置。

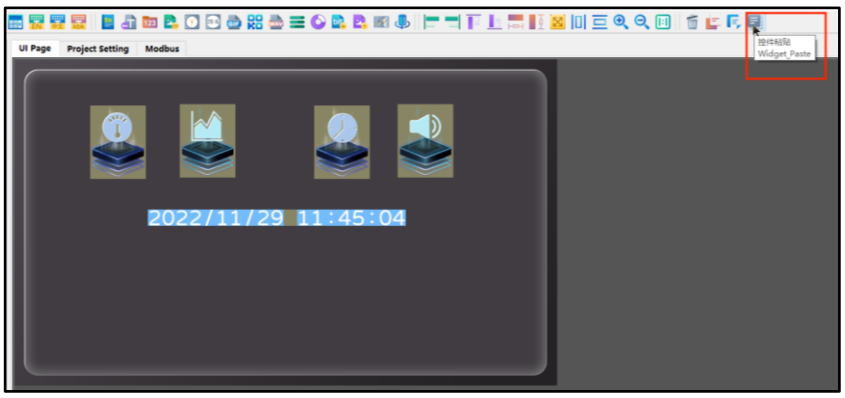


图 5-13：粘贴控件

5.8.6. 微调控件位置功能

功能：选中一个或多个控件，通过键盘上的方向键进行微调，每点击一次方向键，移动一个像素。

底图放大超过编辑视窗的大小时，此时编辑视窗会进入可移动状态。由于方向键控制编辑视窗移动的优先级比方向键控制控件移动的优先级高，所以方向键会先调整编辑视窗的位置，到达编辑视窗末端后才可以微调控件。

如下图所示，通过视窗右方和下方的滑条位置可以得知视窗此时在底图的左上角，想要微调控件位置，需要事先把视窗移动右下角的位置。

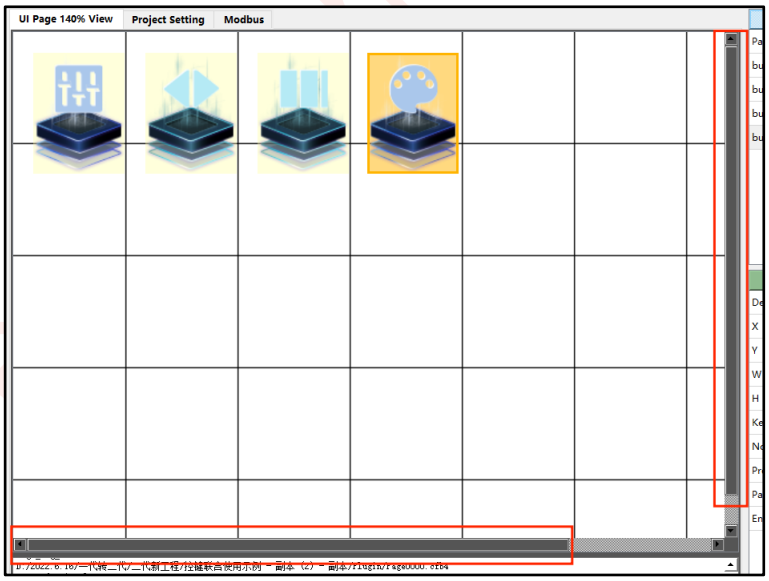
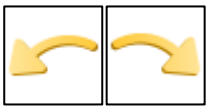


图 5-14：微调控件位置

5.8.7. 上一步和下一步说明

图标： 

功能：该功能用于快速恢复到修改前的某一状态，例如删除或者移动了某个控件，可以通过 Ctrl+Z 快速恢复修改前的状态。这个功能对每一页单独生效，每一页至多可以恢复 50 步操作（前提是该页被更改过这么多次）。只有操作被记录了，用户才能使用 Ctrl+Z 和 Ctrl+Y。需要注意的是，该功能只能保存当次编辑的操作记录，退出工程时会把所有的操作记录删除，只保留最终文件，所以刚打开工程时是无法使用 Ctrl+Z 和 Ctrl+Y。

操作被记录遵循以下规则：

1、每次移动控件坐标都会被记录；每次删除或者添加控件都会被记录。

2、修改控件参数时，只有**切页**或者**是编译保存工程**参数才会被记录。

5.8.8. 解锁和锁定说明

图标： 

功能：该功能用于锁定控件位置和解锁控件位置，防止操作过程中失误改变控件的位置。

5.9. 快捷键操作汇总

1. **编译产生 UartTFT-II_Flash.bin 文件**：Ctrl + B。
2. **微调控件位置**：选中控件后点击上下左右这四个方向键。
3. **新建工程**：Ctrl + N。
4. **打开工程**：Ctrl + O。
5. **复制**：Ctrl + C。
6. **粘贴**：Ctrl + V。
7. **删除**：Delete。
8. **放大界面**：Ctrl + I。
9. **缩小界面**：Ctrl + U。
10. **原始大小**：Ctrl + Q。
11. **设置 writeAddr**：Ctrl + A。
12. **上一步（undo）**：Ctrl + Z。
13. **下一步（redo）**：Ctrl + Y。
14. **保存（Save All）**：Ctrl+S。
15. **打开说明书**：F1 。

6. 控件使用

6.1. 按键（Button）

图标： 

用途：用于执行页面跳转。控件参数列表如右图所示。

Parameter	Data
name	button_0
X	128
Y	108
W	168
H	119
returnValue	0x0020
unpressedIcon	
pressedIcon	
pageGoto	
reportToHost	Disable
hostControl	Disable
_triggerValue	0x0000

图 6-1：按键参数

name：自定义控件名称。

X 和 Y：控件左上角在底图上的位置。

W 和 H：控件的宽和高，作虚拟按键时可以设置宽和高，添加图标时则不必设置，控件会自适应图标的宽和高。

returnValue：串口返回值设置。

unpressedIcon：设置按键图标（未按下状态）。

pressedIcon：设置按键图标（按下状态）。unpressedIcon 和 pressedIcon 两个图片宽高必须一致。

pageGoto：设置按键要跳转的页面。

reportToHost：串口返回使能，按键触发时串口返回 returnValue，详情请查看触摸反馈指令。

hostControl：键码触发使能，开启后该控件的触控失效，详情请查看键码触发说明。

_triggerValue：键码触发值设置。

6.2. 弹窗（Popupbox）

图标： 

用途：把另一页的内容搬到当前页显示操作，需要注意的是弹窗触发之后，只有弹窗的内容才能操作。控件参数列表如右图所示。

Parameter	Data
name	popbox_0
X	215
Y	305
W	460
H	99
returnValue	0x0000
unpressedIcon	
pressedIcon	
pageGoto	
box_X	0
box_Y	0
dimming	Disable
reportToHost	Disable
clearLastPopupBox	Disable
hostControl	Disable
_triggerValue	0x0000
backgroundPage	

图 6-2：弹窗参数

name：自定义控件名称。

X 和 Y：控件左上角在底图上的位置。

W 和 H：控件的宽和高。添加图标后会自适应图标宽高。

returnValue：串口返回值设置。

unpressedIcon：设置按键图标（未按下状态）。

pressedIcon：设置按键图标（按下状态）。unpressedIcon 和 pressedIcon 两个图片宽高必须一致。

pageGoto：设置弹窗弹出的页面的来源（**注意此页底图只能用 bmp 格式**）。

box_X 和 box_Y：弹窗页在当前页弹出时的左上角坐标。

dimming：背景变暗使能，指弹窗弹出后当前页的变化模式。

（1）Disable：即无变化。

（2）Enable：指当前页在弹窗弹出后亮度降低，而弹窗部分的亮度处于正常亮度。

reportToHost：串口返回使能，控件触发时返回 returnValue，详情请查看触摸反馈指令。

clearLastPopupBox：退出当前弹窗后，是否清空弹窗内容。

hostControl：键码触发使能，开启后该控件的触控失效，详情请查看键码触发说明。

_triggerValue：键码触发值设置。

backgroundPage：弹窗弹出背景页选择，使用选中页作为背景页预览。

注：

1、我们在设计工程时，有时需要使用弹窗里再次调用弹窗的操作。这个时候我们如果需要把弹窗的背景变黑，那么我们只需要把第一个弹窗控件的 dimming 设置成 Enable 模式，其它的弹窗的依旧设成 Disable 模式。

2、在使用弹窗里调用弹窗的操作时，会碰到上一弹窗与当前弹窗弹出位置和大小不统一的问题，如果想让上一弹窗消失，那么当前弹窗的触发控件就可以把 clearLastPopupBox 设置成 enable；如果说两个弹窗的弹出位置和大小都是一致的，那么 clearLastPopupBox 直接设置成 disable 即可。

3、弹窗没有记录路径功能，无法从当前弹窗原路返回上一个弹窗。

4、**弹窗弹出页面的底图只能用 bmp 格式的图片。**

6.3. 变量调节（Variable Button）

图标： 

用途：给单一变量多次连续地赋值。控件参数列表如右图所示。

Parameter	Data
name	varAdj_0
X	297
Y	85
W	180
H	121
writeAddr	0x0001
adjStep	1
minValue	0
maxValue	100
dataType	uchar
gradation	+
cyclicalCounting	Loop
longPress	Once
unpressedIcon	
pressedIcon	
reportToHost	Disable
hostControl	Disable
_triggerValue	0x0000

name：自定义控件名称。

X 和 Y：控件左上角在底图上的位置。

W 和 H：控件的宽和高，添加图片后会自适应。

writeAddr：数据存在变量储存空间的起始地址。

adjStep：步进，每次点击后变量的变化值。

maxValue/minValue：最大最小值，两者可以相等，相等时相当于赋这个值到目的地址，范围为-32767~32767（10 进制）。

dataType：数据类型选择，共 5 种可用（uchar，char，ushort，short，位控制）。

gradation：设置点击时变量变化的方向，递增或递减。

cyclicalCounting：是否开启循环。Loop 表示增加（减小）到最大（最小）值时，下一次触摸变量会被调节成最小值（最大值）形成一个循环。位控制不受限制。

longPress：Once 表示按下松开会触发一次按键，长按也只能触发一次；Repeat 表示长按有效，长按期间变量会一直增或减。

图 6-3：变量调节参数

unpressedIcon：设置按键图标（未按下状态）。

pressedIcon：设置按键图标（按下状态）。unpressedIcon 和 pressedIcon 两个图片宽高必须一致。

reportToHost：串口返回使能。返回 writeAddr 和数值，详情请查看触摸反馈指令。

hostControl：键码触发使能，开启后该控件的触控失效，详情请查看键码触发说明。

_triggerValue：键码触发值设置。

注：

1、使用数据类型为 char 和 uchar 的变量调节给一个地址赋值时，不会改变该变量的高 8 位。

2、使用位控制时，maxValue 和 minValue 只能是 1 或 0。

3、给数字控件赋值时需要注意两者的数据类型要一致。

6.4. 多变量操作（Multi-Variable Button）

图标： 

用途：实现一键操作多个变量，最多可同时操作 8 个变量，但是只能单次单程操作。例如可以一键关闭音乐、切换图片、跳转页面等。控件参数列表如右图所示。

name：自定义控件名称。

X 和 Y：控件左上角在底图上的位置。

W 和 H：控件的宽和高，添加图片后会自适应。

unpressedIcon：设置按键图标（未按下状态）。

pressedIcon：设置按键图标（按下状态）。unpressedIcon 和 pressedIcon 两个图片宽高必须一致。

pageGoto：设置按键将要跳转的页面，无需跳转时不设置。

writeAddr0~7：操作变量地址。

_value0~7：给操作变量赋值（16 进制）。

reportToHost：串口返回使能，详情请查看触摸反馈指令。

hostControl：键码触发使能，开启后该控件的触控失效，详情请查看键码触发说明。

_triggerValue：键码触发值设置。

Parameter	Data
name	batVar_0
X	471
Y	113
W	165
H	196
unpressedIcon	
pressedIcon	
pageGoto	
writeAddr0	0xFFFF
_value	0xFFFF
writeAddr1	0xFFFF
_value	0xFFFF
writeAddr2	0xFFFF
_value	0xFFFF
writeAddr3	0xFFFF
_value	0xFFFF
writeAddr4	0xFFFF
_value	0xFFFF
writeAddr5	0xFFFF
_value	0xFFFF
writeAddr6	0xFFFF
_value	0xFFFF
writeAddr7	0xFFFF
_value	0xFFFF
reportToHost	Disable
hostControl	Disable
_triggerValue	0x0000

图 6-4：多变量操作参数

6.5. 环形触摸（Circular Touch）

图标： 

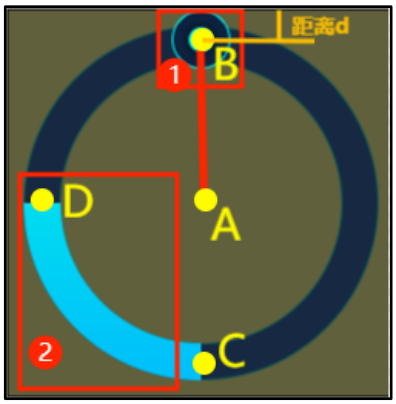
用途：以环形触控的方式控制一个变量。控件参数列表如右图所示。

name：自定义控件名称。

writeAddr：数据存在变量储存空间的起始地址。

X 和 Y：控件左上角在底图上的位置。

W 和 H：控件的宽和高。



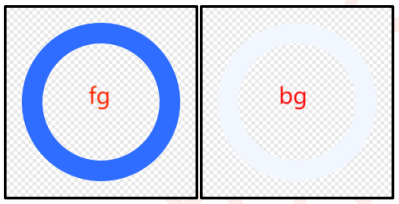


图 6-5：环型触控举例

Parameter	Data
name	rtouch_0
writeAddr	0x0001
X	47
Y	31
W	218
H	189
slide_R	50
touch_R	50
minValue	0
maxValue	100
defaultValue	0
startAngle	0
finalAngle	359
reportToHost	Disable
Touch Ctrl	Enable

图 6-6：环型触控参数

slide_R：环形滑条圆心到滑动图标圆心的距离，如图 6-5 中 A 点（圆心)到 B 点的距离（一般设置为控件高度的一半）。

touch_R：触摸区域的半径，如图 6-5，以 B 为中心点，向内向外同时扩展 touch_R 距离，作为触控区域，只有在触控区域滑动滑条才有效。

minValue 和 maxValue：环形滑条可操作的范围，-32768 ~ 32767，两个参数共同决定了数值的范围。

defaultValue：默认值，取最大最小值之间的数值。

startAngle：滑动的起始角度，和 finalAngle 共同决定了滑动的范围。

finalAngle：滑动的终止角度。如上图 6-5 所示，0°（360°）在 C 点，滑动的小图标顺时针旋转角度增加。

起始角度不能大于结束角度，角度设置只能顺时针方向（0°<=起始角度<终止角度<=360°），不能逆时针方向设置。

reportToHost: 串口返回使能。返回 writeAddr 和数值，需要注意是否设置了小数，如数值 60.00 和 6000 均返回 0x1770。详情请查看触摸反馈指令。

Touch Ctrl: 触摸控制使能。开启后可滑动触摸，关闭后即触摸失效。

注:

1、环形触摸控件相当于一个虚拟控件，只有触控功能没有显示功能。

2、环形触摸控件没有显示功能，如果需要控制环形进度条的百分比显示，则需要小图标（Icon）控件的配合。首先需要准备好环形进度条各个百分比的图片，并且按照小图标控件控制一组图片显示的方式将图片导入小图标控件中，具体可查看小图标控件说明。

3、环形触摸控件与小图标控件共地址，通过触摸环形触摸控件可以改变该变量地址的值，进而改变小图标控件的当前映射值，从而显示环形进度条对应的百分比图片。

6.5.1. 用 Icon 控件控制一组图片的方式制作环形触摸进度条举例:

小图标控件需要做到以下设置，如下图所示:

1. 变量地址需要与环形触摸控件保持一致。
2. 将各个进度的百分比素材，以 Icon 控件控制一组素材显示的方式导入。
3. 设置好每张图片的映射值（与滑条的最大最小值对应上）。

图 6-7：小图标控件设置要求

环形触摸控件控件需要做到以下设置，如下图所示:

1. 变量地址与 Icon 控件保持一致。
2. 设置好环形触摸控件的坐标位置以及宽高（一般与 Icon 控件重叠即可，但也可根据需求调整）。
3. 设置好环形触摸控件的半径以及触摸半径。
4. 设置好环形触摸控件的最大最小值，需要与图片映射值相对应。
5. 设置好环形触摸控件的起始和终止角度。

图 6-8：环形触摸控件设置要求

6.6. 滑条（Slider Bar）

图标： 

用途：以滑动的方式修改一个变量。控件参数列表如右图所示。

Parameter	Data
name	slider_0
writeAddr	0x0003
X	328
Y	161
W	155
H	99
touch_X	5
touch_Y	5
touch_W	145
touch_H	89
minValue	0
maxValue	100
defaultValue	0
bar_X	0
bar_Y	0
barIcon	
slideButton	
background	
direction	L_to_R
reportToHost	Disable

图 6-9：滑条参数

name：自定义控件名称。

writeAddr：数据存在变量储存空间的起始地址。

X 和 Y：滑条的左上角坐标。

W 和 H：滑条的宽高，添加背景图之后会自适应；如果没有添加背景图，则需要手动设置（**LT165 系列限制滑条的宽高不超过 80×80，如果超过**[**请用小图标的方式制作滑条效果**]）。

touch_X 和 Y：滑条触控区域的左上角坐标，基准点是滑条的左上角坐标。

touch_W 和 H：滑条触控区域的宽高，设置不能超出背景图范围。

minVlaue 和 maxVlaue：滑条的滑动时可控制的数值范围，-32768 ~ 32767。

defaultValue：滑条初始默认位置。

bar_X 和 bar_Y：滑条前景图的位置，基准点是滑条左上角坐标。

图 6-10：滑条举例

barIcon：滑条前景图（上图绿色长条区域）。

slideButton：滑条的滑动按钮（上图中间菱形）。滑动按钮图片的高度（宽度）必须大于或等于滑条图片的高度（宽度）。

background：滑条的背景图（上图黄橙区域）。可以不添加背景图，也可以在大底图上直接绘制滑条的背景图。

direction：滑条数值从小到大滑动的方向。

reportToHost：串口返回使能。返回 writeAddr 和数值，详情请查看触摸反馈指令。

注:

1、LT165 系列限制滑条控件的宽×高不能超过 80×80，如果超过请用小图标的方式制作滑条效果。

2、在准备素材时，滑条的背景图要完全把滑条包括在里面。

3、如果滑动按钮特别小时，可以把背景图的高度（纵向滑动）或者是宽度（横向滑动）适当加大，这样触控区域就可以设置得更大，滑动体验效果就会更好。如果是不加背景图的情况，则是将控件宽高调整。

6.6.1. 用小图标的方式制作滑条效果

1、当滑条控件的宽×高超过 80×80 时，由于超过缓存最大范围，滑条控件此时只支持触摸功能，不再支持显示功能。

2、当滑条控件不支持显示功能时，如果需要通过触摸的方式控制进度条的百分比显示，则需要小图标（Icon）控件的配合。首先需要准备好进度条各个百分比的图片，并且按照小图标控件控制一组图片显示的方式将图片导入小图标控件中，具体可查看小图标控件说明。

3、滑条控件与小图标控件共地址，通过触摸滑条控件可以改变该变量地址的值，进而改变小图标控件的当前映射值，从而显示进度条对应的百分比图片。

4、用小图标方式制作滑条效果举例

Icon 需要做到以下设置，如下图所示：

1. 变量地址与滑条控件的变量地址保持一致。
2. 将各个进度的百分比素材，以 Icon 控件控制一组素材显示的方式导入。
3. 设置好每张图片的映射值（与滑条的最大最小值对应上）。

图 6-11：Icon 设置要求

滑条控件需要做到以下设置，如下图所示：

6. 变量地址与 Icon 控件保持一致
7. 设置好滑条控件的坐标位置以及宽高（一般与 Icon 控件重叠即可，但也可根据需求调整）。
8. 设置好滑条控件的触摸区域
9. 设置好滑条控件的最大最小值，需要与图片映射值相对应。

图 6-12：滑条设置要求

6.7. 键盘系统的使用说明

6.7.1. 键盘使用教程

对于 165 系列芯片，只支持数字键盘，不支持英文键盘和中文键盘，添加一个键盘通常包括以下步骤：

1、如下图所示，使用键盘控件需要准备两张键盘图片素材作为底图，如下图所示。两张底图除按压键盘部分颜色不同，大小和样式均相同。

图 6-13：键盘页素材

2、将两张键盘底图加入 page 列表。

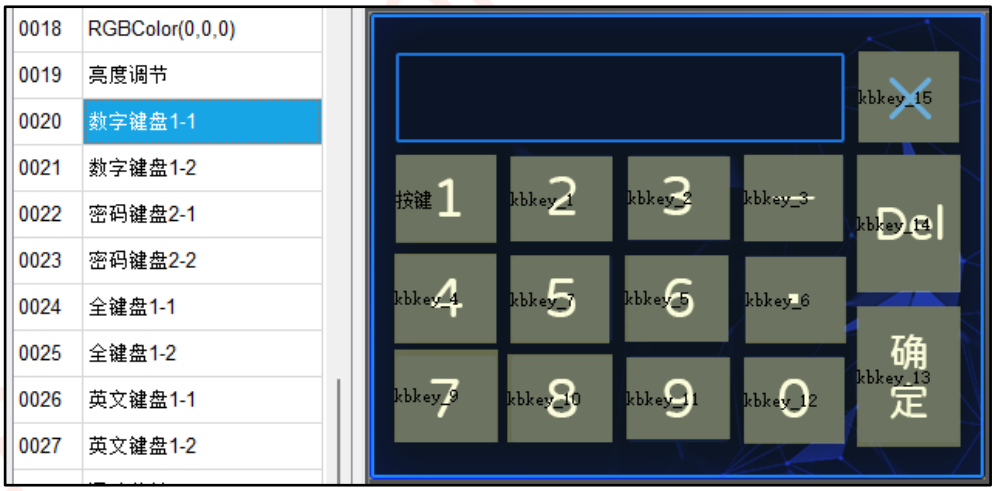


图 6-14：添加无按压效果底图

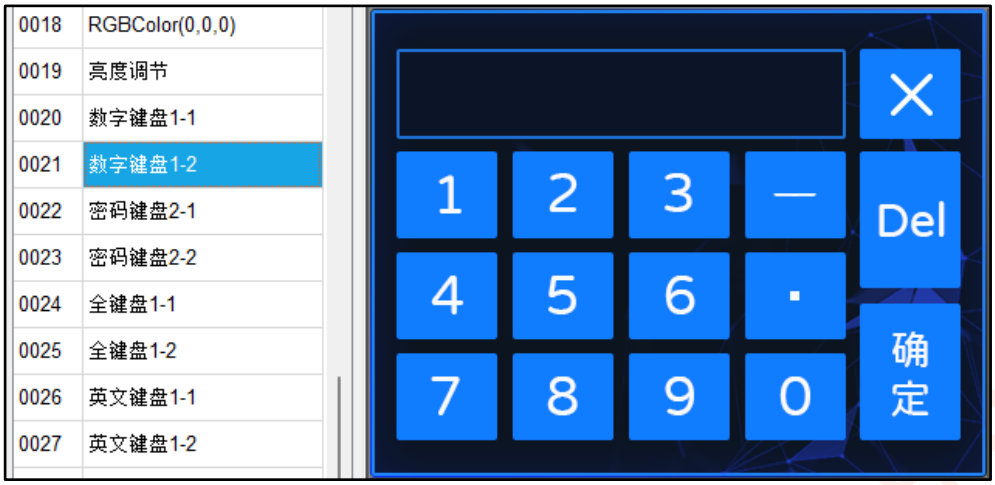


图 6-15：添加有按压效果底图

3、在键盘按键无按压效果页添加键盘按键（SingleKey）控件，**注意该页只能放置键盘按键控件，不能放置其他控件。**键盘按键控件参数设置如下图，keyCode 设置键码，0x0031 是数字 1 的 ASCII 码，需要对照键值键码表填写，具体键值详情，请参考键盘键值设置说明；pressPage 设置为键盘按键有按压效果页，如图是 Page0021。

Parameter	Data
name	kbkey_0
X	20
Y	94
W	64
H	52
keyCode	0x0031
pressPage	Page0021

图 6-16：设置键盘按键

4、设置完键盘页后，开始设置键盘控件，将数字键盘控件的 pageID 设置为键盘按键无按压效果页（即添加了键盘按键控件的页），其他参数按需设置。

dataType	short
pageID	Page0020
fontColor	0x000000

图 6-17：键盘控件设置

注：

1、键盘页不能放置 SingleKey 之外的控件。

2、按压状态键盘页不放置任何控件。

6.7.2. 键盘键值（SingleKey）

图标： 

用途：给按键赋上键码，用于确定键值。控件参数列表如右图所示。

name：自定义控件名称。

X 和 Y：控件左上角在底图上的位置。

W 和 H：键盘按键触发区的宽和高，根据底图键盘按键位置大致适配一下就行。

keyCode：设置按键的键码。

Parameter	Data
name	kbkey_0
X	357
Y	132
W	140
H	121
keyCode	0x0020
pressPage	

图 6-18：键盘按键控件参数

pressPage：按键按下时的状态，按键按下时，会显示所选页对应位置的图片。

注：具体键值说明请参考键盘键值设置说明。

6.7.3. 数字键盘（Numeric Keypad）

图标： 

用途：向指定地址写入一个数字，编码采用 ASCII 编码。控件参数列表如右图所示。

Parameter	Data
name	keypad_0
writeAddr	0xFFFF
byteLength	8
X	262
Y	99
W	383
H	182
kpad_X	0
kpad_Y	0
input_X	20
input_Y	10
input_Max	0
input_Min	0
integerDigit	20
decimalDigit	0
fontWidth	16
inputLimit	false
alignment	Left
cursorColor	Black
fontID	
dataType	short
pageID	
fontColor	0x000000
reportToHost	Disable
backgroundPage	
hostControl	Disable
_triggerValue	0x0000

name：自定义控件名称。

writeAddr：数据存在变量储存空间的起始地址。

byteLength：数据长度（占用变量储存空间的长度），无需修改，根据数据类型自适应。

X 和 Y：键盘控件的左上角坐标。

W 和 H：键盘控件的宽和高。

kpad_X 和 kpad_Y：键盘弹出时的的左上角坐标，基准点是当前页的左上角坐标。

input_X 和 input_Y：输入数字的左上角坐标，基准点是键盘页的左上角坐标（0，0）。

input_Max 和 input_Min：分别设置键盘输入的最大值和最小值，需开启 inputLimit 后生效。可设范围为[-2147483648, 2147483647] 。

integerDigit：键盘输入数字的整数部分的位数。

decimalDigit：键盘输入数字的小数部分的位数。

fontWidth：数字的字体宽度，无需修改，根据字库自适应。

inputLimit：输入限制开关，开启后限制输入数值不能超出 input_Max 和 input_Min 的范围。

alignment：对齐方式。输入数字时，效果如下。

（1）Left： 

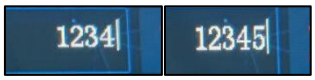
（2）Right： 

图 6-19：数字键盘参数

cursorColor：光标颜色选择。

fontID：字库选择。

dataType：数据类型选择。

pageID：调用的键盘所在的页，即放置键盘按键（SingleKey）的页面。该页必须有键盘底图**且底图格式必须为 bmp**，该参数不能留空。

fontColor：输入数字的字体颜色选择。

reportToHost：串口返回使能，返回 writeAddr 和输入的数值，enter 后返回。详情请查看触摸反馈指令。

backgroundPage：键盘弹出背景页选择，用于预览键盘弹出后的画面。

hostControl：键码触发使能，开启后该控件不能通过触控触发，详情请查看键码触发说明。

_triggerValue：键码触发值设置。

注：

1、用数字键盘给字库数字或者是图片数字赋值时，控件之间的变量地址、数据类型、整数位数和小数位数这 4 个参数必须相同，否则赋值时会出现数据错乱。

2、input_Max、input_Min、integerDigit 和 inputLimit 四者关系说明：input_Max&input_Min 表示输入的值的最大值和最小值；inputNum 表示输入的整数的位数，比如 inputNum=2，输入的值最大为 99；inputLimit 表示是否开启限制开关（具体参考上面的参数说明）。

参考下表，当 inputLimit 启用，则 input_Max 和 input_Min 值有效，**输入值的限制优先受 input_Max 和 input_Min 控制**，如下表第一行；如果 input_Max 和 input_Min 的范围超出了 integerDigit 的极限范围，则输入限制受到 integerDigit 的限制，如下表第二行，integerDigit 设 2 位（极限值 99），Max 值设置 200，开启限制时，因为 200 比 99 大，所以键盘只允许输入小于等于 99 的值。

表 6-1：数字键盘输入限制举例说明表

integerDigit	inputLimit	input_Max	input_Min	输入范围
2	TRUE	60	30	30 — 60
		200	-200	-99 — 99
	FALSE	60	30	00 — 99
		200	-200	00 — 99

如果数字键盘设置了小数位且要开启输入限制，则 input_Max 应当遵循下文设置说明：假设数字键盘设置了 2 位整数及 1 位小数、输入限制需要设置为 67.8，则 inputLimit 设置为开启且 input_Max 设置为 678；假设数字键盘设置了 2 位整数及 2 位小数而输入限制需要设置为 67.89，则 input_Max 应设置为 6789 其他依次类推。

3、不同数据类型的范围在下表给出，使用者可根据实际情况设置。注意这里位数和指的是整数位数和小数位数的和。

表 6-2：键盘数据位数设置建议表

数据类型	范围	最大位数和	键盘建议位数和
char	-128—127	2	2
short	-32768—32767	4	4
int	-2^31—2^31-1	9	9
longlong	-2^63—2^63-1	18	18

6.7.4. 键盘键值设置说明

数字键盘：

ASCII 输入功能码：

0x00F0：取消

0x00F1：确认

0x00F2：退格

表 6-3：数字键盘键值键码对应表

数字键盘键码对应表			
键值	键码	键值	键码
0	0x0030	7	0x0037
1	0x0031	8	0x0038
2	0x0032	9	0x0039
3	0x0033	-	0x002D
4	0x0034	.	0x002E
5	0x0035	取消	0x00F0
6	0x0036	确认	0x00F1
		退格	0x00F2

6.8. 字符串显示（String_Label）

图标： 

用途：可以显示中文、英文、数字和特殊字符。控件参数列表如右图所示。

Parameter	Data
name	label_0
parameterAddr	0xFFFF
writeAddr	0x5E09
wordLength	20
X	140
Y	28
W	176
H	52
fontWidth	32
fontHeight	32
fontID	00_Font_1bit-...
encoding	GB2312
alignment	Left
backgroundColor	Disable
_color	0xD3D3D3
fontColor	0xFFFFFF
defaultText	label_0
passwordMode	Disable
multiLanguage	Disable

图 6-20：字符串参数

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

writeAddr：字符串的存放的起始地址。

wordLength：占用变量储存空间长度，单位 Word。后续的非关联控件在分配储存空间时不能占用这部分变量储存空间。

X 和 Y：控件左上角坐标。

W 和 H：控件的宽和高，高度必须大于或等于字库字体高度。

fontWidth、fontHeight：字库字体宽高，会根据字库自适应，无需设置。

fontID：字库选择。

encoding：字库的编码类型，会根据字库自适应，无需设置。

alignment：对齐方式，共有 9 种对齐方式可选，Y 轴居上三种（T-Left，T-Middle，T-Right），Y 轴居中三种（Left，Middle，Right），Y 轴居下三种（B-Left，B-Middle，B-Right）。

backgroundColor：使能背景色。

_color：设置背景色。

fontColor：选择字体颜色。

defaultText：初始显示，上电时显示的字符串。

passwordMode：开启后所有内容显示成星号，但是不会改变内部数据。

multiLanguage：多国语言使能，开启后改字符串控件可以使用多国语言功能，详情请查看通过字符编码切换实现多国语言切换

注：1、可以通过直接给字符串地址赋汉字的字符编码的形式或者通过中文或者英文键盘输入改变显示内容。串口发送控件数据请查看发送串口指令控制控件举例。

2、输入的字符串很长时，控件需要设置足够的高度，字符串控件才能完整显示文字。

3、字符串控件与键盘关联时，需要使用与键盘字库同一种编码的字库。

4、需要使用 Unicode 字库时，建议使用 WIN10 及以上系统的电脑。

6.9. 静态文本显示（Static_Text）

图标： 

用途：可以显示固定的中文、英文、数字和特殊字符，且不占用变量地址。控件参数列表如右图所示。

Parameter	Data
name	sText_0
parameterA...	0xFFFF
writeAddr	0x0000
wordLength	20
X	327
Y	189
W	160
H	66
fontWidth	20
fontHeight	24
fontID	01_Font-...
encoding	unicode
alignment	Middle
background...	Disable
_color	0xD3D3D3
fontColor	0x000000
defaultText	Statictext_7
passwordM...	Disable
multiLangu...	Disable
Constant_N...	7

图 6-21：静态文本参数

name：自定义控件名称。

parameterAddr：参数地址，此控件为固定显示内容，**此处无需设置更改。**

writeAddr：静态文本不占用存放的变量地址，**此处无需设置更改。**

wordLength：占用变量储存空间长度，单位 Word。根据设置好的内容进行自动读取自行计算，**此处无需设置更改。**

X 和 Y：控件左上角坐标。

W 和 H：控件的宽和高，高度必须大于或等于字库字体高度。

fontWidth、fontHeight：字库字体宽高，会根据字库自适应，**此处无需设置。**

fontID：字库选择，根据设置好的编号进行自动读取，**此处无需设置。**

encoding：字库的编码类型，会根据字库自适应，**此处无需设置。**

alignment：对齐方式，共有 9 种对齐方式可选，Y 轴居上三种（T-Left，T-Middle，T-Right），Y 轴居中三种（Left，Middle，Right），Y 轴居下三种（B-Left，B-Middle，B-Right）。

backgroundColor：使能背景色。

_color：设置背景色。

fontColor：选择字体颜色，根据设置好的编号进行自动读取，**此处无需设置。**

defaultText：显示的内容，根据设置好的编号进行自动读取，**此处无需设置。**

passwordMode：开启后所有内容显示成星号，但是不会改变内部数据。

multiLanguage：多国语言使能，根据工程设置多国语言，**此处无需设置**，静态文本的多国语言功能详情请查看静态文本使用说明。

Constant_Num：编号，与静态文本设置编号对应。

注：1、静态文本内容为固定内容，**不能通过串口发送指令改变数据。**

2、需要使用 Unicode 字库时，建议使用 WIN10 及以上系统的电脑。

6.9.1. 静态文本使用说明

添加一个静态文本通常包括以下步骤：

1、首先在点击 Tool 工具栏中选择 Static Text 一项。

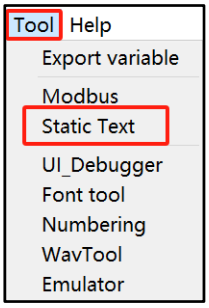


图 6-22：静态文本设置入口

2、点击打开静态文本工具后，弹出下图界面。之后在根据所需要显示的内容填写 3 处，并选好 4 处的颜色和 5 处的字体，有多国语言功能需求则往后面的 language1 继续相同步骤填写。若需要显示多个文本，则点击 6 处的“ADD”按钮进行设置添加。

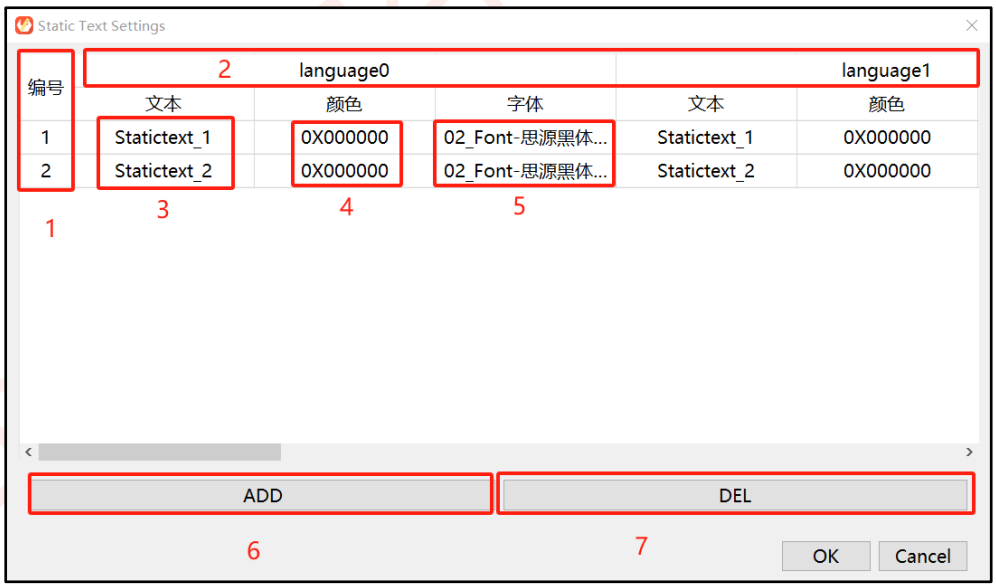


图 6-23：静态文本设置主界面

(1) 编号：与静态文本控件参数 Constant_Num 对应。

(2) language0、language1：多国语言的序号，在工程设置里设置所需要的多国语言的数量后，对内容进行编写。

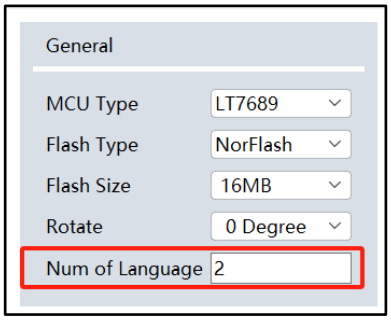


图 6-24：工程设置多国语言

(3) 文本：填写静态文本内容。

(4) 文本：显示静态文本颜色。

(5) 文本：静态文本字库的选择。

(6) ADD：添加多个静态文本。

(7) DEL：删除静态文本。**注**：选中后连续删除是按顺序**往上删除。**

3、拉出静态文本控件，选择对应编号即可显示已配置的文本。

图 6-25：对应的编号选择

注：若是配置多国语言，可以通过串口指令切换多国语言，往 0x703F 写 0x0000 表示切换为 Language0，写 0x0001 表示切换为 Language1。

例如切换 Language1 的完整指令：5A A5 07 10 70 3F 00 01 0E CF

6.10. 字库数字（Text Number Display）

图标： 

用途：显示一个数字。控件参数列表如右图所示。

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

writeAddr：数据存在变量储存空间的起始地址。

byteLength：数据占用变量储存空间的长度，根据选择的数据类型自适应。

X 和 Y：控件左上角坐标。

W 和 H：控件的宽和高，高度必须大于或等于字库字体高度。建议设置时等于字库的高度。

fontWidth：字库字体宽高，会根据字库自适应，无需设置。

fontID：字库选择。

encoding：字库的编码类型，会根据字库自适应，无需设置。

alignment：对齐方式，有左、右和居中对齐三种方式。

integerDigit：设置整数位数。

decimalDigit：设置小数位数。关于小数位数设置可前往整数位和小数位设置查看说明。

Parameter	Data
name	number_0
parameterAddr	0xFFFF
writeAddr	0x0038
byteLength	2
X	274
Y	122
W	253
H	202
fontWidth	65535
fontID	
encoding	
alignment	Left
integerDigit	4
decimalDigit	0
dataType	short
unitSymbol	
_length	0
fontColor	0x000000
defaultNumber	0
leadingZero	Disable

图 6-26：字库数字参数

dataType：数据类型，共 9 种。char，uchar，char_H8，uchar_H8，Short，ushort，int，uint，longlong。关于数据类型的解答可前往数据类型说明查看说明。

uniSymbol：输入的单位，只支持 ASCII 码的符号，不支持中文。

_length：输入的单位所占的字节，一个 ASCII 字符占一个字节。

fontColor：选择字体颜色。

defaultNumber：初始显示，上电时显示的字符串。

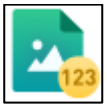
_leadingZero：是否在有效数字之前加 0，Enable 则加 0，Disable 则不加。

注：

1、输入多个小数点时，第二个小数点及其之后的数据会被舍弃。

2、串口发送控件数据请查看发送串口指令控制控件举例。

6.11. 图片数字（Graphics Number Display）

图标： 

用途：以 0 ~ 9、小数点和负号的 png 小图片排列组成数字。控件参数列表如右图所示。

Parameter	Data
name	pngNumber_0
parameterAddr	0xFFFF
writeAddr	0x003C
byteLength	2
X	315
Y	116
W	234
H	230
integerDigit	4
decimalDigit	0
dataType	short
alignment	Left
firstIcon	
lastIcon	
defaultNumber	100
leadingZero	Disable

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

writeAddr：数据存在变量储存空间的起始地址。

byteLength：数据占用变量储存空间的长度，根据选择的数据类型自适应。

X 和 Y：控件左上角坐标。

W 和 H：控件的宽和高，高度会基于导入的图片自适应。

integerDigit：设置整数位数。

decimalDigit：设置小数位数。关于小数位数设置可前往整数位和小数位设置查看说明。

dataType：数据类型，共 9 种。Char，uchar，char_H8，uchar_H8，Short，ushort，int，uint，longlong。关于数据类型的解释可前往可前往数据类型说明查看说明。

图 6-27：图片数字参数

alignment：对齐方式，有左、右和居中三种。

firstIcon：选择”0“这个数字图片。

lastIcon：根据需要选择结束的图片，如果需要小数点或者负号的可以选择“.（小数点）”或者选择“-（负号）”这个数字图片，如果不需要小数点或者负号，可以选择“9”这个数字图片。

defaultNumber：初始值设置，上电时显示的数字。

leadingZero：是否在有效数字之前加 0，Enable 则加 0，Disable 则不加。

注：

1、图片数字的排序是【0 ~ 9、小数点、负号】。

2、输入多个小数点时，第二个小数点及其之后的数据会被舍弃。

3、串口发送控件数据请查看发送串口指令控制控件举例。

6.12. 数字时钟（Digital Clock）

图标： 

Parameter	Data
name	RTC_0
parameterAddr	0xFFFF
X	278
Y	88
W	233
H	156
firstIcon	
lastIcon	
displayFormat	YY/MM/DD ...

用途：显示一个数字形式的时钟。控件参数列表如右图所示。

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

X 和 Y：控件左上角坐标。

W 和 H：控件的宽和高。

firstIcon：数字时钟调用的图片组里的‘0’。

图 6-28：数字时钟参数

lastIcon：数字时钟调用的图片组里的‘星期六’，不适用星期显示则添加到“/(日)”添加完整素材之后会出现预览效果。

displayFormat：显示模式，有以下选择。YY/MM/DD 与 YY/MM/DD/的区别在于前者显示时最后一道斜杠不显示，而后者会显示最后一道斜杠，其它以此类推。

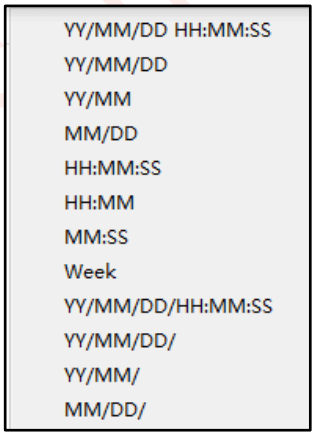


图 6-29：数字时钟样式

注：

1、PNG 图片的排序为【0、1、2、3、4、5、6、7、8、9、：、/（年）、/（月）、/（日）、Sun、Mon、Tues、Wed、Thur、Fri、Sat】，同组 Icon 宽高说明详情请参考同组的 Icon 宽高说明。

2、如果不需要显示星期，则素材只需要准备【0、1、2、3、4、5、6、7、8、9、：、/（年）、/（月）、/（日）】。

3、如果不需要素材“/（日）”，则该序号需要留空，不能被其他素材占用。

4、**该控件需要 RTC 电路支持，若无 RTC 电路，则显示异常。**

6.12.1.修改时间说明

如果要修改时间，总共分为两步。

1、向对应的寄存器写入与时间对应的值。**寄存器对应关系：**年 0x7002 、月 0x7003 、日 0x7004、时 0x7005、分 0x7006、 秒 0x7007。

2、确认修改：向 7008 写入以下值中的一个，包括**【0：年月日时分秒、1：年月日、2：年月、3：月日、4：时分秒、5：时分、6：分秒】**。在实际应用中，需要根据修改时间的哪些组成部分，如果修改年月日，则可以往 0x7008 写入 1 确认修改；如果修改时分秒，则写入 4 确认修改。

3、使用串口修改时间时，直接发送 0x7002~0x7007 的对应值即可，不需要想 0x7008 发确认值。

4、串口修改时间请查看串口发送修改时间 。

6.13. 计时器（Timer）

图标： 

用途：执行计时功能，计时结束后执行预设操作。控件参数列表如右图所示。

Parameter	Data
name	uitimer_0
parameterAddr	0xFFFF
X	283
Y	126
W	261
H	203
presetAddr	0x0040
_value	120
countAddr	0x0041
_value	0
controlAddr	0x0042
_value	1:Start the timer
firstIcon	
lastIcon	
displayFormat	MM:SS
countMode	+
globalCounting	Disable
reportToHost	Disable
writeAddr0	0xFFFF
_value	0xFFFF
writeAddr1	0xFFFF
_value	0xFFFF
writeAddr2	0xFFFF
_value	0xFFFF
writeAddr3	0xFFFF
_value	0xFFFF

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

X 和 Y：控件的左上角坐标。

W 和 H：控件的宽高，添加图片素材后会自适应。

presetAddr：储存正计时停止时间的变量地址。修改其变量更改正计时停止时间。

_value：正计时的停止时间，10 进制，范围 1~65535，单位秒。

countAddr：储存计时器当前计时值的变量地址。修改其变量更改当前计时时间。

_value：计时器当前计时值，10 进制，范围 1~65535，单位秒。

controlAddr：计时器控制地址，修改其变量可以控制计时器。

_value：计时器初始状态设置。

```
0:Pause the timer
1:Start the timer
2:Cancle the timer
3:Show timer at pause state
```

（1）0 = 暂停计时，

（2）1 = 执行计时，

（3）2 = 取消计时，计时器暂停计时且从屏幕上消失

（4）其他值 = 显示计时器（仍处于暂停状态）

图 6-30：计时器参数

firstIcon：添加数字时钟的起始素材”0“。

lastIcon：添加数字时钟的终止素材”/（日）“。

displayFormat：计时器显示模式设置。NULL 表示隐藏计时器，使用时不会显示计时器，不需要素材即可运行。

countMode：选择计时模式，”+” = 正计时，“-” = 倒计时。

globalCounting：选择仅在当前页计时还是全局计时。

（1）Disable：仅在当前页计时，离开当前页面就会暂停计时。

（2）Enable：离开当前页后计时仍继续。

reportToHost：串口返回使能，计时结束时返回。返回 8 个 WriteAddr 和及其对应的数值。

writeAddr0~7：计时结束后执行下一步操作的操作地址。

_value~7：计时结束后执行下一步操作时要往操作地址写入的值。

注：

2、正向计时时，由 Preset_value 和 count_value 共同控制计时时间，**计时时间 = Preset_value - count_value**，所以 Preset_value 必须要大于 count_value，计时从 count_value 开始到 Preset_value 结束。例如 count_value 为 60 即 1 分 0 秒，Preset_value 为 180 即 3 分 0 秒，正向计时 2 分钟，串口屏显示为 01:00 正向计时到 03:00。

3、倒数计时时，计时时间由 count_value 单独控制，**计时时间 = count_value**，倒计时从 count_value 开始到 0 结束，例如 count_value 为 60 即 1 分 0 秒，串口屏显示为 01:00 倒计时到 00:00。

4、计时器 **displayFormat** 选择 SS 时，计时器显示为秒的 10 进制显示，显示位数由 Preset_value 控制，即 Preset_value 为 4 位数时计时器就显示 4 位数。

5、**该控件需要 RTC 电路支持，若无 RTC 电路，则显示异常。**

6.14. 动图（Gif）

图标： 

用途：显示动图，播放开机动画。控件参数列表如右图所示。

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

writeAddr：数据存在变量储存空间的起始地址，用于控制 Gif 播放。

X 和 Y：动图的左上角坐标。

W 和 H：动图的宽高，添加动图后会自适应。

playOnceCode：播放一次 Gif 时操作地址内的值，往变量地址 writeAddr 写入该值，Gif 仅播放一次。

startCode：Gif 启动值，往 writeAddr 写入该值，Gif 启动。

stopCode：Gif 停止值，往 writeAddr 写入该值，Gif 暂停。

playAtStart：暂停 Gif 播放后，再次启动是从第一帧开始还是从当前帧开始。Enable 表示从第一帧开始，Disable 表示从当前帧继续播放。

interval(10ms)：动图每相邻两帧之间的刷新间隔，最大值可设置 255，单位是 10ms，比如设置为 2，那就是两帧之间的间隔是 20ms。

gifName：添加动图。

dataFormat：图片取模格式选择，不选择则默认跟随工程设置，关于 dataFormat 详情请参考 dataFormat 说明。

Parameter	Data
name	gif_0
parameterAddr	0xFFFF
writeAddr	0x0043
X	342
Y	136
W	196
H	180
playOnceCode	11
startCode	10
stopCode	100
playAtStart	Disable
interval(10ms)	5
gifName	
dataFormat	
defaultStatus	Run
effects	Disable
writeAddr0	0xFFFF
_value	0xFFFF
writeAddr1	0xFFFF
_value	0xFFFF
writeAddr2	0xFFFF
_value	0xFFFF
writeAddr3	0xFFFF
_value	0xFFFF
writeAddr4	0xFFFF
_value	0xFFFF

图 6-31：Gif 参数

defaultStatus：初始值状态设置，包括四种状态。

（1）Run：上电后循环播放动图。

（2）Stop：上电后停止在第一帧。

（3）Disappear：上电后不显示。

（4）RunOnce：上电播放一次后停止并执行下一步操作。

effects：Gif 与图片数字（aPNG）、Icon（aPNG）、文本显示(字库带灰阶)这三个控件叠加时需要选择 enable，否则默认 disable，开启时这三个控件显示在 Gif 上面。

writeAddr（0~7）：播放一次后停止并执行下一步操作的变量地址。

_value（0~7）：播放一次后停止并执行下一步操作时要往操作地址写入变量的值。

注:

1、往 Gif 的控制地址写 0xFFFF，可以使得 Gif 消失，除了 startCode、stopCode、playOnceCode 和消失这 4 个值之外，其他值对于 Gif 来说是非法的，控制值设置范围为［0，65534］，4 个值要互不相同。

2、在使用变量调节控制 Gif 时，变量调节控件的数据类型要设置为 ushort 型。

3、播放一次停止后要跳转页面，则 writeAddrN 写 0x7000，其对应的_valueN 填入对应页码的 16 进制数。

4、Gif 仅支持与图片数字、Icon（aPNG）、文本显示(字库带灰阶,即 2 或 4bits)这三个显示控件叠加，控件叠加时，三个控件必须完全处于 Gif 控件范围内，否则会出现显示错乱问题。

5、Gif 的播放速度仅与每帧的大小、**interval** 值二者有关，与 Gif 素材的播放速度无关。

6、Gif 编译后转化为 bin 文件得大小计算请查看图片素材转 bin 文件说明。

6.15. 音频播放（Audio Play）

图标： 

用途：播放音频，最多支持播放 99 个音频文件。控件参数列表如右图所示。

Parameter	Data
name	wav_0
X	393
Y	48
W	181
H	192
wavID	
repeat	Disable

图 6-32：音频参数

name：自定义控件名称。

X，Y，W，H：坐标和宽高，音频是声音，这些参数无效。

Wav ID：点击添加上电时播放的音频文件。

repeat：播放音频时是否循环。

（1）Disable：不循环，播放完一首音乐后就停止，也不会播放列表的下一首。

（2）Enable：循环播放当前首音乐，并非循环播放列表里的音乐。

注：

1、同一页只能放置一个音频控件，不能同时放置多个。

2、切换音频方法：往 WAV 控制寄存器写入对应值，寄存器地址为 0x700A。

写入 0x0000：停止播放，

写入 0x0001：播放第 1 首，即编号为 0000 的音频

写入 0x8001：循环播放第 1 首。

3、可以往 0x700A 寄存器写入对应数值直接播放对应音频，无需放置音频控件。放置音频控件是为了进入该页面时就播放一次音频，以达到换页响应的效果。

6.16. 位元状态（Bit Status）

图标： 

用途：通过判断变量地址内的 16 位 2 进制数据的对应位是否为 1，从而改变位元状态的显示。控件参数列表如右图所示。

Parameter	Data
name	bitIcon_0
parameterAddr	0xFFFF
writeAddr	0x00AA
bitIndex	bit0
X	451
Y	111
W	144
H	253
offStateIcon	
onStateIcon	
overlap	Disable

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

writeAddr：数据存在变量储存空间的起始地址。

bitIndex：选择位元状态的控制位，选择范围 0 ~ 15。

（1）对应位为 1 时（不受其它位影响），显示 onStateIcon；

（2）对应位为 0 时，显示 offStateIcon。

（3）变量初值默认为 0x0000。

图 6-33：位元状态参数

X 和 Y：控件的左上角坐标，基准点是当前页的左上角坐标

W 和 H：宽高，添加素材后会自适应。

offStateIcon：对应变量的对应位为 0 时显示，使用时必须添加图片。

onStateIcon：对应变量的对应位为 1 时显示，使用时必须添加图片。

overlap：叠加使能，主要是用于处理 Icon 控件、位元状态控件之间互相叠加的问题，当控件选择带透明度的 PNG 图显示时，如果控件之间需要互相叠加，则选择 Enable；若不需要则选择 Disable。

（1）Disable：从底图截取对应区域的内容与 PNG 图叠加显示；

（2）Enable：从显存截取对应区域的内容与之叠加显示。

注：串口发送控件数据请查看发送串口指令控制控件举例。

6.17. 小图标（Icon）

图标： 

用途：显示一个或者一组小图片。控件参数列表如右图所示。

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x00AC
byteLength	2
X	535
Y	161
W	160
H	236
firstIcon	
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	0
maxDisplayID	0
overlap	Disable

图 6-34：Icon 参数

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

writeAddr：数据存在变量储存空间的起始地址。

byteLength：数据占用变量储存空间的长度。

X 和 Y：控件的左上角坐标，基准点是当前页的左上角坐标。

W 和 H：宽高，添加素材后会自适应。

firstIcon：选择起始图片编号，该参数不能为空。

lastIcon：选择结束图片编号。如果想要显示一组小图片，这一组小图片编号必须连续，宽高必须一致。

dataFormat：图片取模格式，不选择则默认跟随工程设置，详情请参考 dataFormat 说明。

defaultDisplayID：变量初值设置，该参数输入为空时，则该地址内值默认为 0。

minDisplayID 和 maxDisplayID：这两者必须要和[firstIcon，lastIcon]数值上对应（映射关系）。比如说一组编号连续小图片有 10 张，编号可能是 0100 ~ 0109，那么 minDisplayID 和 maxDisplayID 可以设置成 [0，9]、 [10，19]等，符合 **max - min + 1 = 图片数量** 即可。如果数值上不对应，会导致在切换图片时出现错误。可设置范围为 [0 , 65535] 。

overlap：叠加使能，主要是用于处理 Icon 控件、位元状态控件之间互相叠加的问题，当控件选择带透明度的 PNG 图显示时，如果控件之间需要互相叠加，则选择 Enable；若不需要则选择 Disable。

（1）Disable：从底图截取对应区域的内容与 PNG 图叠加显示；

（2）Enable：从显存截取对应区域的内容与之叠加显示。

注：

1、minDisplayID 为首张图片映射值、maxDisplayID 为末张图片映射值，变量地址内的当前值可以称为当前映射值。

2、Icon 的显示与否和对应地址内的当前映射值有关。

（1）显示一个 Icon 时，minDisplayID、maxDisplayID 和当前映射值三者一致时才会显示；

（2）显示一组 Icon 时，如果当前映射值在 minDisplayID 和 maxDisplayID 之间时，则 Icon 会显

示，并且显示的当前图片编号是：当前映射值-首张图片映射值+首张图片编号。

3、在多组 Icon 共用同一个变量地址时，只需要设置其中一组的 defaultDisplayID。

6.18. 编码器（Encoder）

图标： 

用途：用编码器替代部分触控操作（需要旋钮硬件支持）。编码器的参数如右图所示。在选定一个 item 执行之前，编码器旋转可以控制其变量地址（writeAddr）内的变量增减，此时的变量与 item 是相对应的，即旋转旋钮改变变量的值为 2 时，表示当前编码器来到了 item2，此时单击编码器表示选中并执行当前的 item2。

Parameter	Data
name	encoder_0
writeAddr	0x00AD
X	521
Y	126
W	146
H	179
item0	NULL
item1	NULL
item2	NULL
item3	NULL
item4	NULL
item5	NULL
item6	NULL
item7	NULL
item8	NULL
item9	NULL
item10	NULL
item11	NULL
item12	NULL

长度：无。

name：自定义控件名称。

writeAddr：编码器变量地址，与控制的 Icon 共地址。该地址不能设置为 0x7000 及其往后地址。

X 和 Y：控件的左上角坐标。

W 和 H：对于该控件宽高无意义。

item0 ~ 15：用于定义不同操作的功能，共 4 种功能可以选择。由于每一页只能放置一个编码器控件，所以每一页的操作最多有 16 个。

注：编码器控件每一页只能放置 1 个。

图 6-35：编码器参数

6.18.1.编码器使用说明

编码器旋转可以控制其 writeAddr 内的变量增减，编码器与 Icon 控件共用地址时，可以控制 Icon 显示与消失。编码器控制 Icon 有两种模式，**模式一**是 1 个编码器控制多个 Icon 控件显示与消失；**模式二**是 1 个编码器控制 1 个 Icon 控件切换图片。

模式一：1 个编码器控制多个 Icon 控件显示与消失，即旋钮控制变量变化从而控制多个 Icon 控件中的一个单独显示一张图片，最后达成旋钮选中功能的效果。

1、Icon 需要做到以下设置：

(1) Icon 与编码器共地址。

(2) Icon 设置参数时，min 和 max 值需要相等。因为当变量当前值与 min（max）相等时，该 Icon 就会显示；当变量当前值变化时，对应 Icon 就会显示或者消失。

(3) 不同 Icon 的 min 和 max 值不相同且从 0 递增。

(4) Icon 个数与编码器的 item 设置个数相等。

2、编码器需要做到以下设置：

(1) 编码器与 Icon 共地址。

(2) 编码器的 item 设置个数与共地址的 Icon 个数相等。

(3) item 设置不能留空，例如不能设置了 item1 而没有设置 item0。

3、模式一举例如下：

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x2801
byteLength	2 与编码器共地址
X	397
Y	69
W	82
H	108
firstIcon	0000.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	0 两个值相同
maxDisplayID	0
overlap	Disable

Parameter	Data
name	icon_1
parameterAddr	0xFFFF
writeAddr	0x2801
byteLength	2
X	502
Y	83
W	82
H	108
firstIcon	0001.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	1
maxDisplayID	1
overlap	Disable

Parameter	Data
name	icon_2
parameterAddr	0xFFFF
writeAddr	0x2801
byteLength	2
X	640
Y	88
W	82
H	108
firstIcon	0002.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	2
maxDisplayID	2
overlap	Disable

图 6-36：编码器模式一 Icon 设置

Parameter	Data
name	encoder_0
writeAddr	0x2801
X	477
Y	243
W	168
H	193
item0	1,,0xFFFF,... 三个Icon对应三个item
item1	1,,0xFFFF,...
item2	4,,Disable,0xFF...
item3	NULL
item4	NULL
item5	NULL
item6	NULL
item7	NULL
item8	NULL
item9	NULL

图 6-37：模式一编码器设置

模式二：1 个编码器控制 1 个 Icon 控件切换图片，即旋钮控制变量变化从而控制一个 Icon 切换一组图片，最后也达成旋钮选中功能的效果。

1、Icon 需要做到以下设置：

(1) Icon 与编码器共地址。

(2) 这 1 个 Icon 放置了一组连续的 N 张图片（N <= 15）

(3) Icon 的 min 和 max 值设为为[0 ~ N]

2、编码器需要做到以下设置：

(1) 编码器与 Icon 共地址。

(2) 编码器的 item 设置个数与 Icon 的导入的图片数 N 相等。

(3) item 设置不能留空，例如不能设置了 item1 而没有设置 item0。

3、模式二举例如下：

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x2901
byteLength	2
X	396
Y	70
W	82
H	108
firstIcon	0000.png
lastIcon	0002.png
dataFormat	
defaultDisplayID	0
minDisplayID	0
maxDisplayID	2
overlap	Disable

与编码器共地址

二者数量对应

Parameter	Data
name	encoder_0
writeAddr	0x2901
X	477
Y	243
W	168
H	193
item0	1,,0xFFFF,...
item1	1,,0xFFFF,...
item2	4,,Disable,0xFF...
item3	NULL
item4	NULL
item5	NULL
item6	NULL
item7	NULL

item与图片数量对应

图 6-38：编码器模式二 Icon 设置（左图）与编码器设置（右图）

编码器主要有 3 种操作，旋钮旋转、单击、双击和长按：编码器单击表示选中并执行当前操作，编码器单击后程序就会执行用户预设的操作（item）；双击、长按和单击三者的效果一致，不同的是触发条件发生了改变。编码器每个 item 有 4 个子功能（Mode1~4）可选，其中 Mode1~3 都是单击触发操作，Mode4 可以设置单击、双击和长按，四个子功能会在后文介绍。

注：编码器 item 控制的地址不能与编码器的 writeAddr 相同。

6.18.2.编码器旋转切换页面功能说明

需要实现编码器左右旋转切换页面时，就需要使用编码器旋转切换页面功能，该功能需要通过设置页面参数内的 **向左滑动跳转页和向右滑动跳转页** （**不支持上下滑动跳转页和带滑动效果的滑动翻页**）两个参数开启，如下图，如何设置**左右滑动跳转页**可参考页面跳转第四类无滑动效果的滑动跳页。

参数	数据
页面名称	热疗1
X坐标	0
Y坐标	0
宽度	800
高度	480
页面底图	0002.bmp
页面底色	RGBColor(0,0...
向左滑动跳转页	Page0004
_返回值	0x0000
向右滑动跳转页	Page0003
_返回值	0x0000
向上滑动跳转页	
_返回值	0x0000
向下滑动跳转页	
_返回值	0x0000

图 6-39：设置页面滑动参数

当某一页面设置了**左右滑动跳转页** 且 当前页面存在编码器控件，则显示该页面时程序会将**编码器旋转切换页面寄存器** 0x7043 的变量值修改为 0x0001，当**编码器旋转切换页面寄存器** 的变量值为 0x0001 时，编码器旋转的动作将用来操作页面跳转，旋转方向对应关系如下：

编码器**顺时针旋转**：**向右滑动跳转页；**

编码器**逆时针旋转**：**向左滑动跳转页；**

在该页面按压一次编码器，则程序将**编码器旋转切换页面寄存器**的变量值修改为 0x0000，当**编码器旋转切换页面寄存器** 的变量值为 0x0000 时，编码器旋转的动作回归正常修改编码器控件关联的变量地址的变量值

按压编码器将**编码器旋转切换页面寄存器**的变量值修改为 0x0000 后，切换至别的页面再回来当前页面，程序会重新将**编码器旋转切换页面寄存器**的变量值修改为 0x0001，也可以通过串口发送或者是编码器的子功能将**编码器旋转切换页面寄存器 0x7043** 的变量值修改为 0x0001，以回到旋转切换页面模式。

6.18.3.编码器 item 设置

编码器在使用时，通过旋钮旋转改变其对应地址内的数值，数值的改变带动 Icon 的显示与消失，旋转选择到某个 Icon 时，按下旋钮，该 Icon 对应的 item 就会被执行。关于编码器内 item 的 4 个子功能（Mode1~4）设置如下：

双击 item0，弹出右图界面：

Mode：共 4 种选择，详情请看下文。

New：生成选择的 Mode。想要更改不同 Mode，只需要选择对应 Mode，然后点击 New。

Clear：清除当前 Mode。

OK：确认该 item 的设置。想把 item 清空时，点击 Clear 再点击 OK，该 item 就变成了 NULL。

Cancel：直接退出，维持原样，不保存任何修改

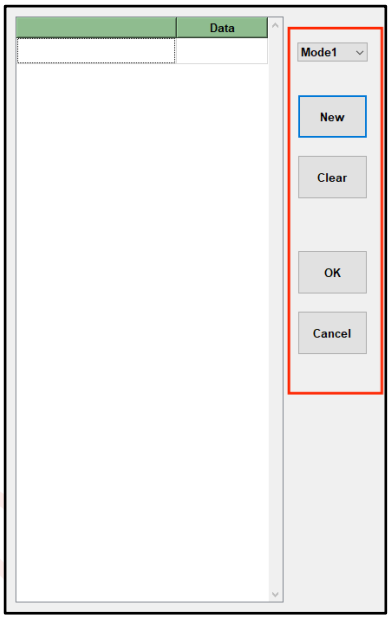


图 6-40：编码器 Mode 设置界面

6.18.3.1.编码器子功能一

功能说明：编码器单击实现跳页和赋值功能。

controlMode：功能名称，可自定义。

reportToHost：串口返回使能。返回 writeAddr 和输入的数据，详情可查看触摸反馈指令。

pageGoto：页面跳转。

writeAddr-（0 ~ 7）：操作地址。

_value（0 ~ 7）：给对应地址赋的值。

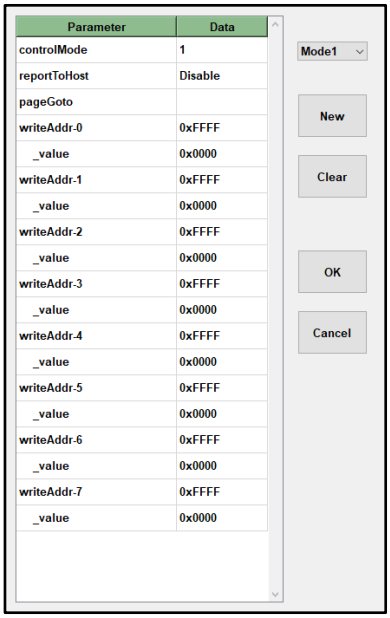
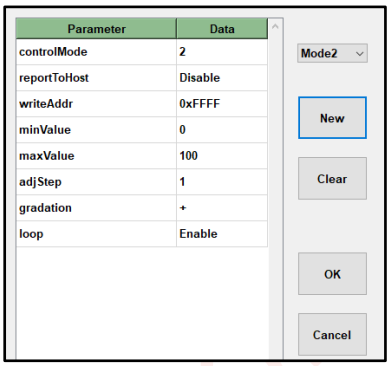


图 6-41：编码器子功能一

6.18.3.2.编码器子功能二



功能说明：编码器单击实现控制变量增减。

controlMode：功能名称，可自定义。

reportToHost：串口返回使能。返回 writeAddr 和输入的数据，详情可查看触摸反馈指令。

writeAddr：操作地址。

minValue 和 maxValue：变量调节的范围，可设范围为-32768 ~ 32767。

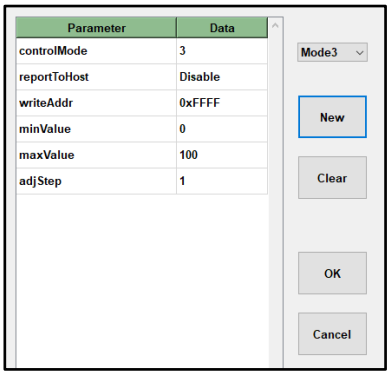
图 6-42：编码器子功能二

adjStep：步长，点击一次旋钮，变量变化的值。

gradation：设置点击时变量变化的方向，增加或者减少。

loop：是否开启循环。开启时，变量变化到极限值时，再点击一次会赋另一个极限值，形成循环。

6.18.3.3.编码器子功能三



功能说明：编码器旋钮旋转控制变量增减，单击进入和退出该模式。

controlMode：功能名称，可自定义。

reportToHost：串口返回使能。返回 writeAddr 和输入的数据，详情可查看触摸反馈指令。

writeAddr：操作地址。

minValue 和 maxValue：变量调节的范围，可设范围为-32768 ~ 32767。

adjStep：步进，旋钮旋转一定角度，变量变化的值。

图 6-43：编码器子功能三

6.18.3.4.编码器子功能四

功能说明:单击,双击和长按编码器旋钮实现控制跳页和变量赋值,一次只能触发一种操作。

controlMode: 功能名称，可自定义。

singleClickPageGoto: 单击跳转目标页。

singleClickReport: 单击串口返回使能,详情可查看触摸反馈指令。

singleClick_Addr（0~7）：变量地址。

_value（0~7）：给对应变量地址赋值。

doubleClickPageGoto: 双击跳转目标页。

doubleClickTimeGap(50ms)：设置双击触发的有效时间，50*N(1<=N<=255)毫秒。双击的触发时间可以适当延长。

doubleClickReport: 双击串口返回使能，详情可查看触摸反馈指令。

doubleClick_Addr（0~7）：变量地址。

_value（0~7）：给对应变量地址赋值。

longPressPageGoto: 长按跳转目标页。

Parameter	Data
controlMode	4
singleClickPageGoto	
singleClickReport	Disable
singleClick_Addr0	0xFFFF
_value	0x0000
doubleClickPageGoto	
doubleClickTimeGap(50ms)	2
doubleClickReport	Disable
doubleClick_Addr0	0xFFFF
_value	0x0000
longPressPageGoto	
longPressDuration(50ms)	20
longPressReport	Disable
longPress_Addr0	0xFFFF
_value	0x0000
longPress_Addr1	0xFFFF

图 6-44: 编码器子功能四

longPressDuration(50ms): 设置长按触发的有效时间，50*N(1<=N<=255)毫秒。

longPressReport: 长按触发时串口返回使能，详情可查看触摸反馈指令。

longPress_Addr（0~7）：变量地址。

_value（0~7）：给对应变量地址赋值。

6.19. 自动变量（Automatic variable）

图标： 

用途：对指定变量进行自动增减操作。控件参数列表如右图所示。

Parameter	Data
name	autoVar_0
parameterAddr	0xFFFF
X	542
Y	102
W	93
H	144
presetAddr	0x00AF
_value	0
loopCode	0
onceCode	1
stopCode	2
stepValue	1
interval(10ms)	1
targetAddr	0xFFFF
dataType	short
minValue	0
maxValue	100
gradation	+
reportToHost	Disable
writeAddr0	0xFFFF
_value	0xFFFF
writeAddr1	0xFFFF
_value	0xFFFF
writeAddr2	0xFFFF
_value	0xFFFF
writeAddr3	0xFFFF

name：自定义控件名称。

parameterAddr：参数地址，LT165 系列不支持通过参数地址修改控件信息。

X 和 Y：控件左上角坐标。

W 和 H：控件宽高。该控件属于虚拟控件，宽高属于无效数据。

presetAddr：控件的控制地址。

_value：控件控制地址的初始化值。

loopCode：设置循环的控制值。presetAddr 内容为该值时，控件循环执行。

onceCode：设置执行一次后停止的控制值。presetAddr 内容为该值时，控件执行一次。

stopCode：设置停止控制值。presetAddr 内容为该值时，控件停止执行。

stepValue：设置变量每次变化的幅度值。

interval(10ms)：设置变量每次变化的间隔时间，单位 10ms，范围为 1~65535。

targetAddr：目标变量的变量地址。

dataType：目标变量的数据类型。

minValue 和 maxValue：自动变化的范围，不能超出所选数据类型的数值范围。

图 6-45：自动变量控件参数

gradation：自动变化的变化方向，增加或者减少。

reportToHost：串口反馈使能，计数结束后反馈 targetAddr 及该地址当前值，详情参考触摸反馈指令。

writeAddr0~7：执行一次后对该地址进行赋值。

_value0~7：给对应地址赋值。

注：第一次对指定变量进行增减操作时，若 targetAddr 的初始值在最小最大值范围内，自动变化起始值为 targetAddr 的初始值。若 targetAddr 的初始值超出最小最大值范围，则自动变化起始值为 minValue（+）或者 maxValue（-）。

6.20. 条形进度条

图标： 

用途：显示进程进度。控件参数列表如右图所示。

name：自定义控件名称。

parameterAddr：参数地址，可通过该参数与串口配合实现实时修改部分控件参数，详情请参考修改进度条参数。

writeAddr：数据存在变量储存空间的起始地址。

X 和 Y：进度条的左上角坐标，基准点是当前页的左上角坐标。

W 和 H：宽高，添加素材后会自适应。

bar_X 和 bar_Y：进度条相对于背景图的坐标。如果不添加背景图，则 bar_X 和 bar_Y 必须都为 0。

direction：进度条由小到大的方向。

barIcon：进度条样式图片。

Parameter	Data
name	progress_0
parameterAddr	0xFFFF
writeAddr	0x00A8
X	435
Y	119
W	183
H	190
bar_X	0
bar_Y	0
direction	L_to_R
barIcon	
minValue	0
maxValue	100
defaultValue	0
background	

图 6-46：条形进度条参数

minValue 和 maxValue：最小最大值，两者组成进度条的受控范围。受控范围为 [-32768 , 32767]。

defaultValue：初始值设置。

background：进度条背景图。

6.21. 环形进度条

图标： 

用途：以环形的样式显示进程进度。控件参数列表如右图所示。

Parameter	Data
name	rProgress_0
parameterAddr	0xFFFF
writeAddr	0x00A9
X	467
Y	215
W	130
H	109
foreground	
background	
minValue	0
maxValue	100
defaultValue	0
startAngle	0
finalAngle	359
promptNum_X	65
promptNum_Y	54
integerDigit	3
decimalDigit	0
alignment	Left
fontID	
fontColor	0x000000
firstIcon	
lastIcon	
digitDisplayMode	NULL

name：自定义控件名称。

parameterAddr：参数地址，可通过该参数与串口配合实现实时修改部分控件参数，详情请参考修改环形进度条参数。

writeAddr：数据存在变量储存空间的起始地址。

X 和 Y：进度条的左上角坐标，基准点是当前页的左上角坐标。

W 和 H：宽高，添加素材后会自适应。

foreground：前景图片。

background：背景图片。

minValue 和 maxValue：最小最大值，两者组成进度条的受控范围。受控范围为 [-32768 , 32767]。

defaultValue：初始值。

startAngle：开始角度。

finalAngle：结束角度。

promptNum_X，promptNum_Y：数字的坐标。基准点是控件左上角坐标。优先设置对齐方式再设置数字坐标。

integerDigit：整数位数设置。

decimalDigit：小数位数设置。

alignment：数字显示对齐方式，与环形滑条一致，详情请参考环形触摸（Circular touch）。

图 6-47：环形进度条参数

fontID：数字显示的字库选择。

fontColor：数字显示的字体颜色。

firstIcon：使用图片数字显示时，图片的第一张，也就是“0”。

lastIcon：使用图片数字显示时，图片的“小数点”。

digitDisplayMode：数字显示显示的样式选择，包含 Null、FontNum 和 IconNum（图片数字）三种。

（1）Null：不显示。

（2）FontNum：选择字库文字。

（3）IconNum：选择图片数字

注：

1、在准备素材时，必须准备 **foreground** 和 **background** 两张图，两者参数不能留空。

2、**foreground** 和 **background** 两张图除了需要变化的地方允许不一样，其它部分要全部保持一致。

3、此控件的参数设置与环形触控的同名参数设置相同，详情请参考环形触摸（Circular touch）。

6.22. 页面布局控件说明

页面布局所包含的控件如下图所示，控件从左到右分别为左对齐、右对齐、上对齐、下对齐、等宽、等高、等宽高、水平等距、垂直等距、放大、放小和原始大小（100%），详情请看下文。

图 6-48：布局控件

页面布局控件的使用如下：

1、首先在编辑页面中选择一个已经放置好的控件作为基准控件；

2、其次根据需求点击左对齐、右对齐、上对齐、下对齐、等宽和等高这六个布局控件中的一个，点击布局控件后，鼠标光标由  切换成  ，表明进入选择模式；

3、长按左键拖动鼠标，在编辑界面上选择需要快捷布局的控件，鼠标左键释放后执行快捷布局操作；

4、在编辑区域内单击右键退出选择模式，完成快速布局。

注：

1、水平等距和垂直等距布局不需要选择基准控件，因此对这两个布局进行操作时，只需要从上文的步骤二开始即可

2、所有控件在使用时，选择区域需要将被改动的控件**左上角包括进去**，该控件才会被选中参与接下来的布局，如果只是被选中区域包括其它部分，该控件不会参与快捷布局；基准控件可以不被包括在选择区中，被选中的控件依旧会以它为基准。

3、如果被操作的控件加入了图片，该控件的宽高就会自适应成图片的宽高，那么等高、等宽、等宽高这三个控件对其使用就失去了意义。其他的对齐和调整间距的操作依旧可以生效。等宽高这几个控件更多的是运用于数字、字符串这类控件和虚拟控件（如键值设置控件）的大小调节过程中。

6.22.1.控件左对齐（Left_Align）

图标： 

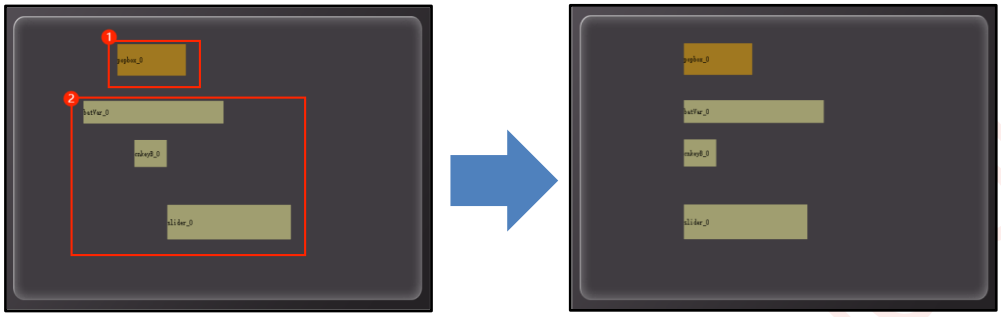


图 6-49：左对齐效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件对齐方式，随后框选需要被对齐的控件（如上左图红框 2 内控件），释放鼠标，被对齐控件向基准控件的左边界对齐。

6.22.2.控件右对齐（Right_Align）

图标： 

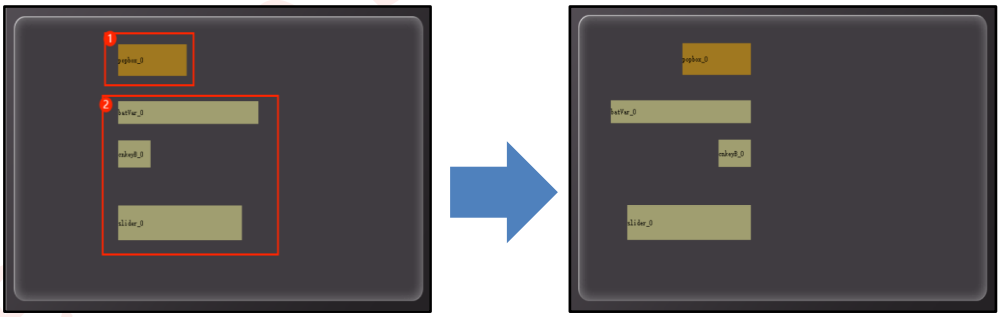


图 6-50：右对齐效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件对齐方式，随后框选需要被对齐的控件（如上左图红框 2 内控件），释放鼠标，被对齐控件向基准控件的右边界对齐。

6.22.3.控件上对齐（Top_Align）

图标： 

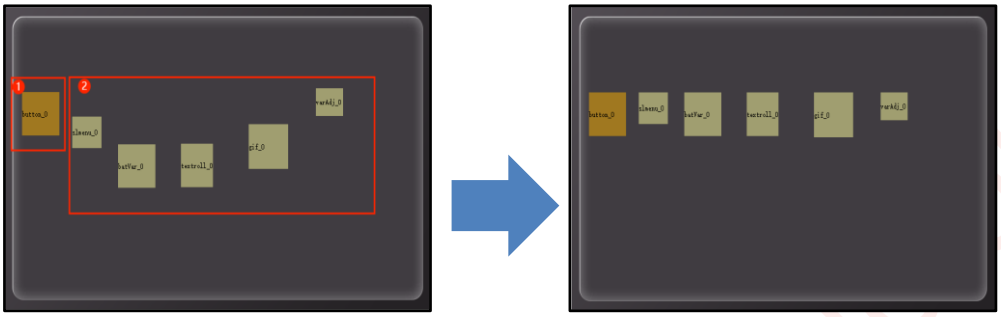


图 6-51：上对齐效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件对齐方式，随后框选需要被对齐的控件（如上左图红框 2 内控件），释放鼠标，被对齐控件向基准控件的上边界对齐。

6.22.4.控件下对齐（Bottom_Align）

图标： 

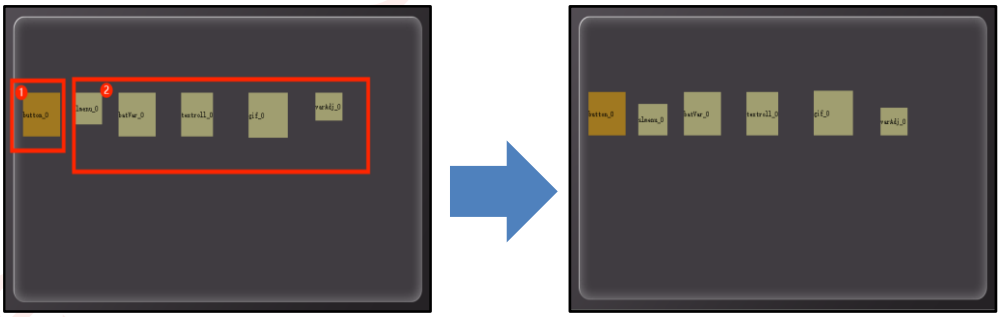


图 6-52：下对齐效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件对齐方式，随后框选需要被对齐的控件（如上左图红框 2 内控件），释放鼠标，被对齐控件向基准控件的下边界对齐。

6.22.5.控件等宽（Width_Align）

图标： 

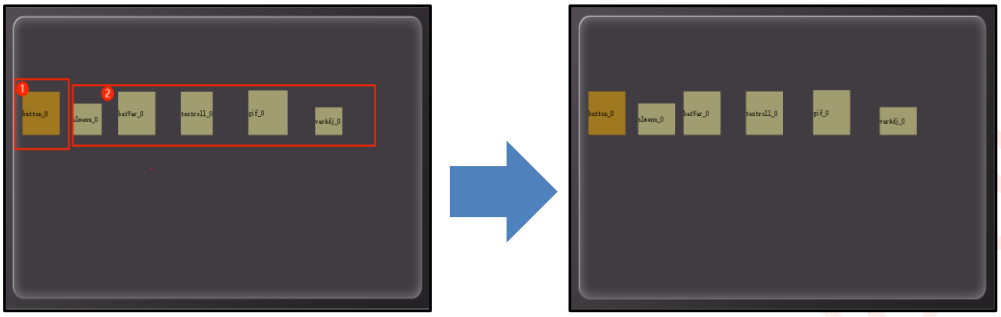


图 6-53：等宽效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件调整方式，随后框选需要被调整的控件（如上左图红框 2 内控件），释放鼠标，被调整控件以基准控件的的宽度进行调节。

6.22.6.控件等高（Height_Align）

图标： 

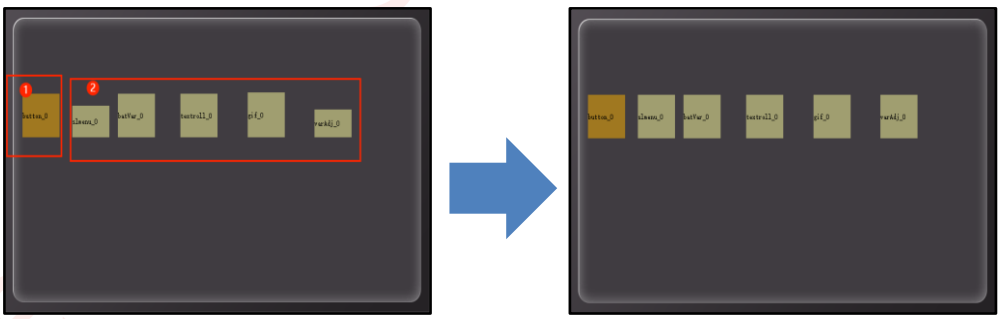


图 6-54：等高效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件调整方式，随后框选需要被调整的控件（如上左图红框 2 内控件），释放鼠标，被调整控件以基准控件的的高度进行调节。

6.22.7.控件等宽高（Shape Consistent）

图标： 

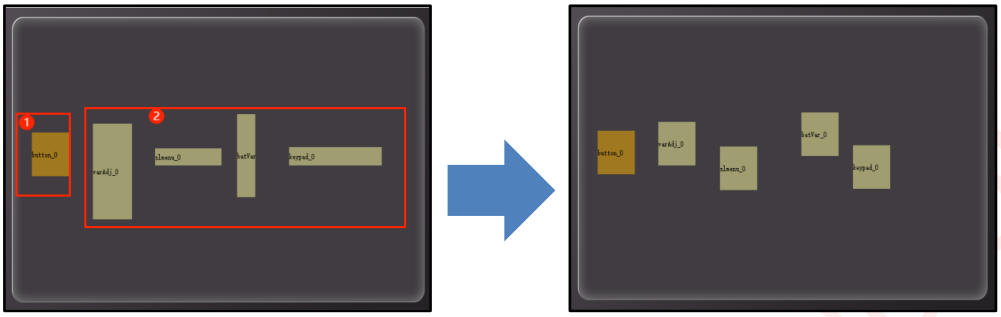


图 6-55：等宽高效果图

首先点击选择基准控件（如上左图红框 1 内控件），再点击控件调整方式，随后框选需要被调整的控件（如上左图红框 2 内控件），释放鼠标，被调整控件以基准控件的的宽高进行调节。

6.22.8.水平等距（Horizontal Equidistance）

图标： 

直接点击水平等距控件，鼠标切换为手指图标，框选需要调整的控件，释放鼠标，软件会以选定的多个控件的最左边控件和最右边控件的 X 坐标的差值作为**布局范围**，将里面的控件水平等距。假设选中区域从左到右共 N 个控件，水平布局后每个控件左上角坐标的 X 值相差 $\left(\frac{X_N-X_1}{N-1}\right)$ 个像素点。

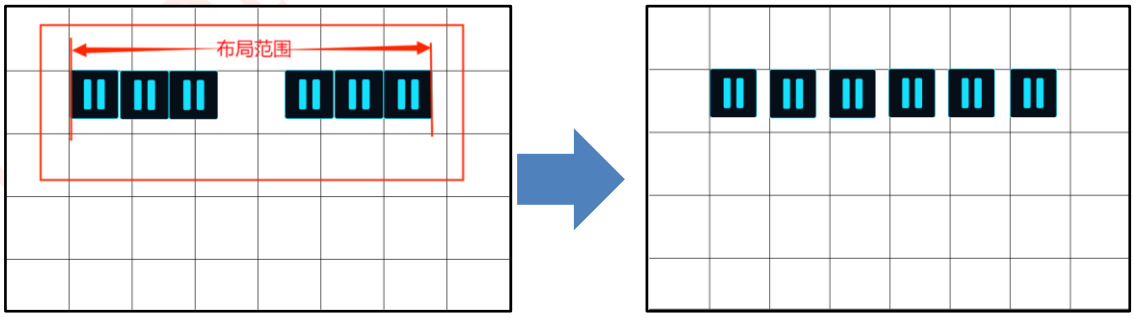


图 6-56：水平等距效果图

6.22.9.垂直等距（Vertical Equidistance）

图标： 

与水平布局的说明同理，只是把横坐标改为纵坐标。

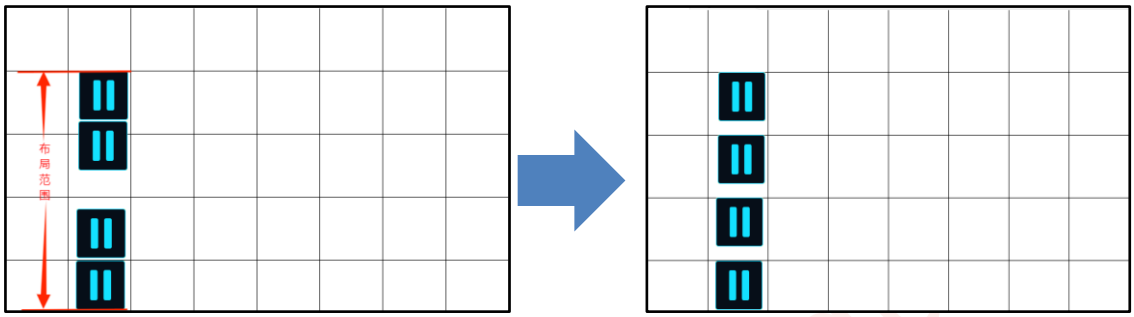


图 6-57：垂直等距效果图

6.22.10. 放大缩小功能（Zoom in or out）

图标： 

包括放大、缩小、原始大小三个控件，缩小最小能到原始大小的 40%，放大最大可达 300%，每次点击放大或者是缩小控件时，编辑界面放大或者缩小 20%，也就是说缩放范围为 40% ~ 300%，步进为 20% 。处于缩放状态时，所有的控件坐标大小不会受变化比例影响，设置时依旧按原定位置；在编辑窗口左上角可查看缩放倍数。处于放大状态时，可通过编辑窗口右方及下方的滑条控制编辑界面的位置（鼠标滚轮可控制其上下移动）。

注：此类操作支持快捷键操作，Ctrl + I 放大，Ctrl + U 缩小，Ctrl + Q 原始大小。

放大与缩小效果图如下图所示（灰色为不可编辑区域）：

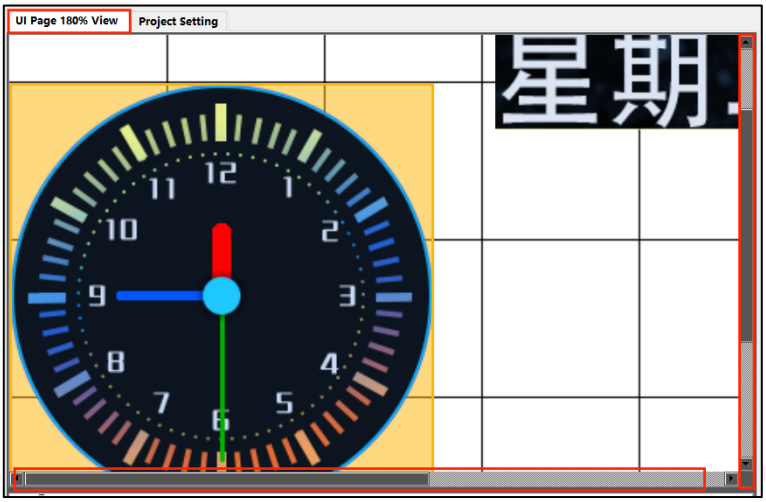


图 6-58：编辑界面放大

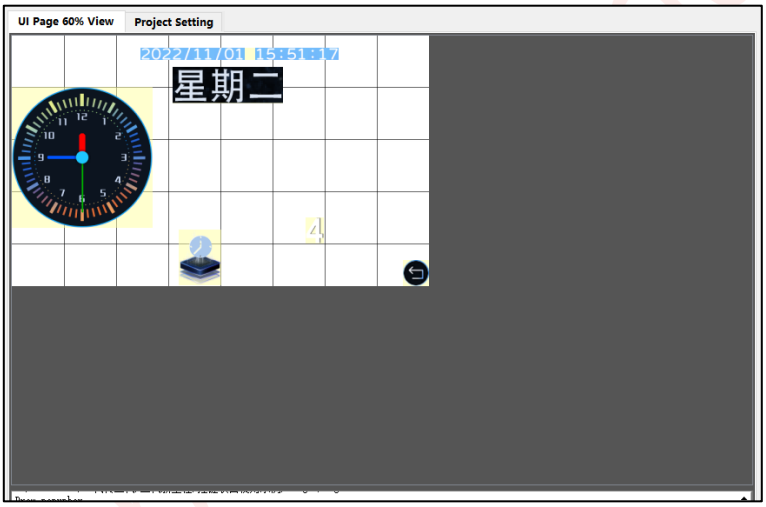


图 6-59：编辑界面缩小

7. 存储空间与变量地址说明

7.1. RAM 的储存空间

165 系列的 RAM 空间为 1KB，地址范围只有 0x0000 ~ 0x0200。每个地址对应的空间占 2Bytes。

在 UI_Editor_II 使用变量地址或参数地址地址时，设置的地址为数据存储空间的首地址，即数据从设置的地址（首地址）开始按序依次储存。每个变量地址（首地址）都指向的空间大小是不固定的，**因此在 UI_Editor_II 软件中给各个变量分配变量地址时，应计算好需储存的数据量，否则将可能出现分配空间的重叠而导致出错。**

7.2. 变量地址（writeAddr）

变量地址是 RAM 空间中储存某一个或多个变量的子空间的首地址，在该地址指向的空间中储存了显示变量的编码或状态变量的值，例如，将一个文本显示控件的变量地址设置为 0x0100，控件中显示的文本内容为“乐升半导体”，那么在 RAM 空间中的储存方式如下图所示。可以看出，**每个地址中可储存 2 Byte 的内容。**

图 7-1：变量地址内数据的储存方式

如果需要改变这个文本显示控件中显示的字符，只需要改变相应变量地址中储存的字符编码即可。通过发送指令或者触摸屏录入都可修改变量地址中储存的数据。例如，修改文本显示控件可通过文本录入控件实现，只需将两个控件设置相同的变量地址即可，详情请参考键盘系统的使用说明；同时，通过发送指令的方式也可修改变量地址中储存的值，串口指令详情请参考串口通信指令。

7.3. 参数地址（parameterAddr）

参数地址是 RAM 空间中储存描述某一变量的属性的子空间的首地址，在该地址指向的空间中存了显示变量的各项属性值，如显示坐标、颜色、字体大小等。需要指出的是，参数地址与变量地址共用 RAM 空间，分配变量地址时应避免空间的重合。详情可参考串口发送直接修改控件参数。

7.4. 特殊寄存器地址说明

0x7000~0x71FF 为特殊寄存器地址范围，部分特殊寄存器以及功能如下。控制寄存器的完整串口指令可查看发送串口指令控制寄存器举例。

1. **换页寄存器：**地址 0x7000，串口发送页码可直接跳转到目标页。可查看串口发送控制页面跳转。

2. **背光寄存器：**地址 0x7001，写入 0 ~ 63 就可以改变屏幕亮度等级，屏幕亮度分为 0 ~ 63 共 64 个等级。可查看串口发送控制屏幕亮度。

3. **时间寄存器：**地址 0x7002 ~ 0x7007，分别对应年月日时分秒。若通过 UI 控件可以向对应寄存器写入对应的时间，写入数据后系统时间不会马上修改，还需要通过确认修改时间寄存器进行修改。可查看串口发送修改时间。

表 7-1：时间寄存器表

寄存器地址	时间	赋值范围
0x7002	年	10—99
0x7003	月	01—12
0x7004	日	01—31
0x7005	时	00—23
0x7006	分	00—59
0x7007	秒	00—59

4. **确认修改时间寄存器：**地址 0x7008，向寄存器写入以下数值可实现修改对应时间。0：年月日时分秒，1：年月日，2：年月，3：月日，4：时分秒，5：时分，6：分秒。使用串口更新时间时不需要往 7008 写入确认值。

表 7-2：修改时间确认值对应表

寄存器地址	写入数值	修改对象
0x7008	0	年
	1	月
	2	日
	3	时
	4	分
	5	秒
	6	

5. **WAV 控制寄存器：**地址 0x700A，用于控制 WAV 的播放。写入 0x0000 表示停止播放，写入 0x0001 表示播放编号为 0000 的音频，写入 0x8001 表示循环播放编号为 0000 的音频；写入 0x0005 表示播放

编号为 0004 的音频，写入 0x8005 表示循环播放编号为 0004 的音频。若写入 0x0005，但是工程内没有编号为 0004 的 WAV 文件则不执行播放。可查看串口发送控制音频播放。

6. **音量调节寄存器：**地址 0x700B，支持 17 个音量等级，写入 0 ~ 16 可调节音量，16 表示最大音量等级，0 是静音。可查看串口发送调节音量。
7. **电阻屏校准寄存器：**地址 0x700C，写入 0x005A 执行电阻屏校准，校准完成后会将寄存器清零。可查看电阻屏校准指令。
8. **键码触发寄存器：**地址 0x700D。键码触发的具体使用方法详情请参考键码触发说明。
9. **自动背光控制寄存器：**地址 0x700E，写入 0 表示关闭自动背光，写入 1 表示开启。同参数 Auto Dimming。可查看自动背光控制寄存器。
10. **休眠背光亮度控制寄存器：**地址 0x700F，写入值为休眠时屏幕背光亮度。同参数 Sleep（0~63），可查看休眠背光亮度控制寄存器。
11. **自动背光休眠时间控制寄存器：**地址 0x7010，写入的值为无操作时进入休眠所需的时间，单位秒。同参数 Hold time（s）。可查看自动背光休眠时间控制寄存器。
12. **串口升级寄存器**：地址 0x7011，写入 0xAA55 进入串口升级模式（需要 **bootloader** 支持）。可查看串口升级寄存器。
13. **多国语言寄存器**：地址 0x703F，写入不同的数值可以切换不同的语言。多国语言功能使用请查看多国语言功能说明。
14. **横竖屏 UI 工程切换寄存器**：地址 0x7040，写入 0 或 1 切换横屏或竖屏的 UI 工程。使用前请查看横竖屏 UI 工程说明 。
15. **屏幕显示检测寄存器：**地址 0x7041，写入 1~3 可令串口屏进入屏幕显示加测模式，可查看屏幕显示检测寄存器 。
16. **UI 工程升级寄存器**：地址 0x7042，通过串口发送数据升级 UI 工程，可查看使用 Flash RW-II 软件更新 UI 工程。
17. **编码器旋转切换页面寄存器**：地址 0x7043，该寄存器的变量值为 1 时，表示开启编码器旋转切换页面功能；为 0 时表示关闭编码器旋转切换页面功能。编码器旋转切换页面功能请查看编码器旋转切换页面功能说明。

注：

1、标号 9、10、11 的 3 个寄存器功能与设置中的 Backlight Control 中的含义一致，创建这几个寄存器的目的是为了用串口控制自动背光。

8. 多国语言功能说明

多国语言功能可以通过切换不同语言体系的 Icon 图片达到一键切换语言的显示效果，也可以使用字符串控件和文本滚动控件通过切换不同的文字编码实现。该功能受到 0x703F 多国语言寄存器控制，通过向该寄存器写入对应的数值控制切换对应的素材。UI_Editor-Lite_V2.10 及以上版本的编辑软件和 Mcu_Code 才支持该功能。

8.1. 通过图片切换实现多国语言切换

使用切换图片的方法实现多国语言功能需要以下三个步骤：

1、在工程设置的语言数量参数框填入对应的语言数量。

2、按要求建立多国语言文件夹。

3、按要求存放多国语言素材。

8.1.1. 多国语言 Icon 文件夹设置

与普通工程相比，多国语言工程的区别在于 Icon 文件夹内增加了名称为 0001~NNNN 的其他国语言文件夹。需要注意的是，编辑 UI 工程时，控件导入的 Icon 素材只能从主文件夹即 Icon 文件夹内选择，不能选择子文件夹即 0001~NNNN 的其他国语言文件夹内的素材。

图 8-1：多国语言 Icon 文件夹设置

8.1.2. 其他语言的文件夹素材说明

如上图所示，其他语言文件夹内存放其他国家语言的图片素材，只有在切换语言时需要变动的图片才需要存放在这些文件夹内。例如，主文件夹内有一张编号为 0000 的图片，切换语言 1 是这张图片需要切换，则将语言 1 的该张图编号为 0000 存放在文件夹 0001 中。注意不同语言编号的图片分辨率需要一致，图片格式需要相同。

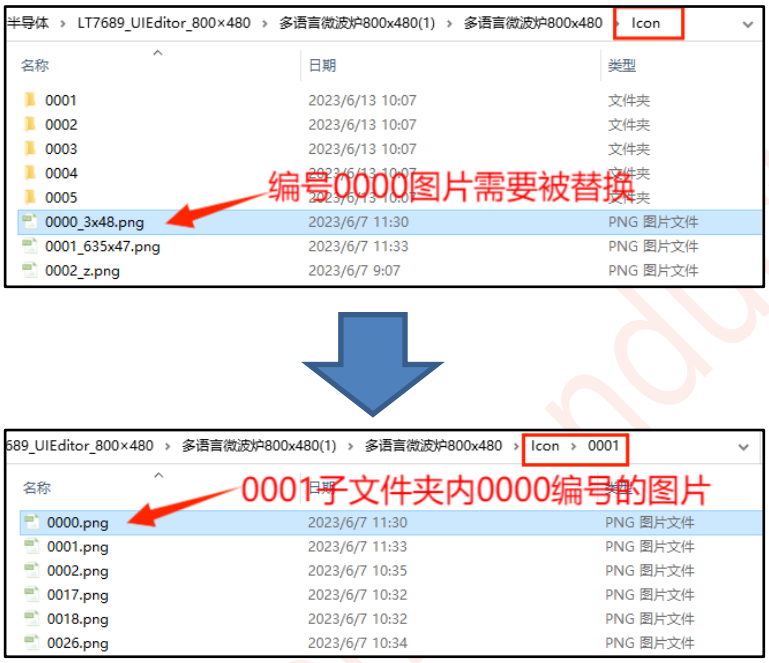


图 8-2：其他语言的文件夹素材说明

8.1.3. 多国语言所支持的控件

只有 Icon 文件夹内的素材才支持多国语言切换，也就是说，所有使用 Icon 文件夹内素材的控件都支持多国语言切换。

8.1.4. 多国语言切换过程

假设文件夹 0001 内存放的是英语的图片素材，文件夹 0002 内存放的是韩语的图片素材，如果需要切换语言为 English，则往 0x703F 写 0x0001；如果需要切换语言为韩语，则往 0x703F 写 0x0002；需要切换回默认语言则写 0x0000。

例如切换 Language1 的完整指令：5A A5 07 10 70 3F 00 01 0E CF

图 8-3：多国语言切换过程

注：目前，LT165 系列支持多国语言。

8.2. 通过字符编码切换实现多国语言切换

使用切换字符编码的方法实现多国语言功能需要以下三个步骤：

1、在工程设置的语言数量参数框填入对应的语言数量。

2、使用 Font Tool 采集 unicode 字库。

3、设置字符串控件或文本滚动控件的多国语言。

8.2.1. 采集 Unicode 字库

Font Tool 的使用可参考字库取模工具使用说明，需要注意所需语言的字符集范围，采集字库时需要完全包括这些字符集。Unicode 字库编码字符集范围可参考软件目录下的 **unicode 编码字符集表.docx** 文档。

8.2.2. 设置多国语言

1. 将字符串控件或文本滚动控件的 multiLanguage 参数设置为 Enable，点击 defaultText 弹出以下弹窗。

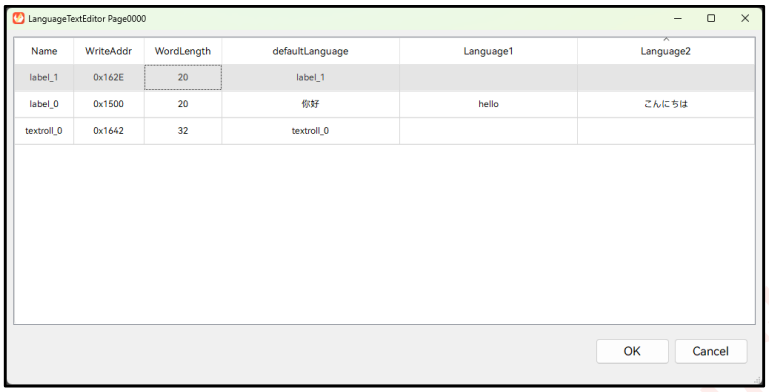


图 8-4：输入多国语言

2.在不同语言列下的输入框输入对应的语言，需注意在 unicode 编码里，所有字符的编码均用 2Bytes 表示，包括 ASCII 码，例如大写字母‘A’在 GBK 编码里用 0x41 表示，在 unicode 编码里用 0x00 0x41 表示。所以在设置控件的 WordLength 时应当以最长的字符串为准。字符串长度最大为 2Kbytes。

3.查看输入的字符串是否超过字符串控件的宽度，在多国语言弹窗内点击一次输入框，控件就会显示预览效果。

图 8-5：多国语言预览

8.2.3. 多国语言切换过程

往 0x703F 写 0x0001 表示切换为 Language1，写 0x0002 表示切换为 Language2。
例如切换 Language1 的完整指令：5A A5 07 10 70 3F 00 01 0E CF

9. 配套工具使用说明

如下图，三个工具均位于软件目录，这三个软件必须使用 UI_Editor-Lite 压缩包解压出来的，不能使用其它来源。

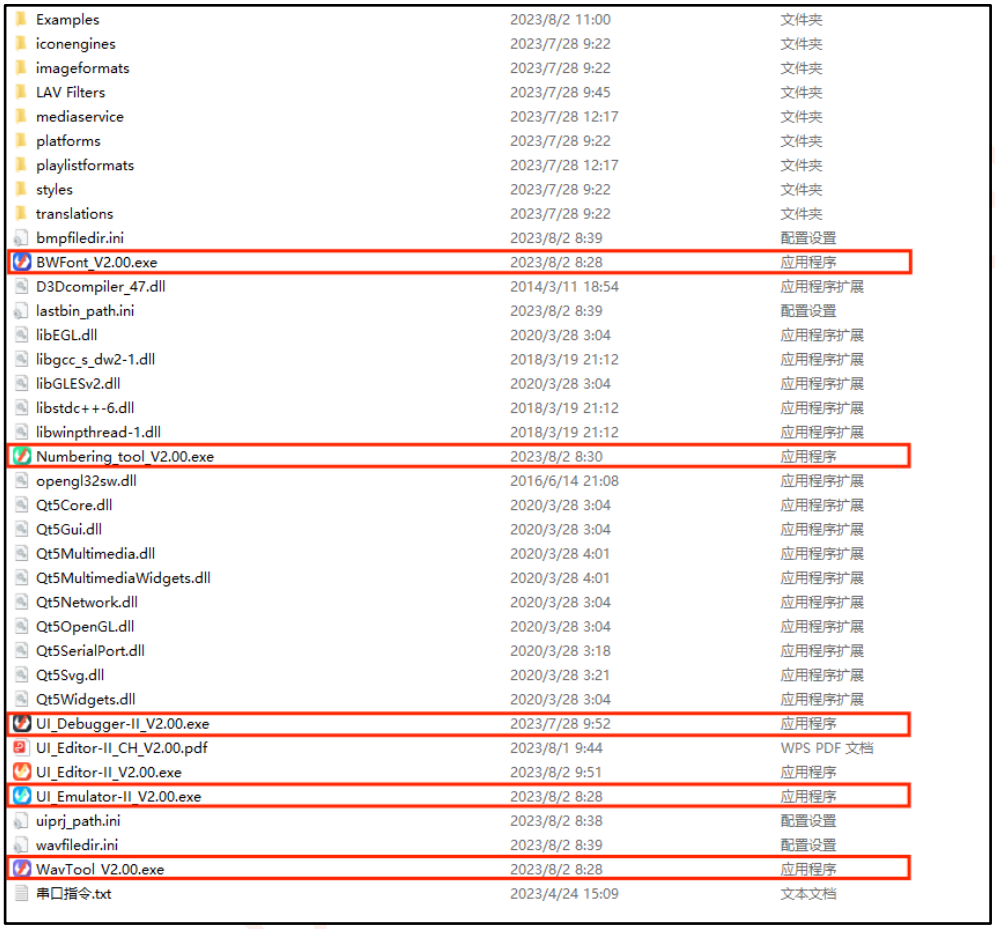


图 9-1：配套处理工具

9.1. 二代串口屏模拟器（UI_Emulator-II）使用说明

9.1.1. 模拟器入口及主界面说明

UI_Emulator-II 软件是二代串口屏工程在电脑上的运行环境，可以快速地、初步地排查 UI 工程是否存在问题。**模拟器功能相当于硬件绝对没有问题的串口屏，所以模拟器运行工程时所有的串口屏功能都是可用的（功能的使用方式对时），同样的工程烧录至实体串口屏（板子）上可能会出现某些功能无法使用的现象，那是因为板子上的硬件不支持。最常见的例子就是时钟和计时器功能，模拟器模拟时钟功能时，时间数据的来源是电脑的时钟，所以可以正常运行；如果板子上没有 RTC 电路或者 RTC 电路故障，同样的工程烧录至实体串口屏上，时钟功能是不可用的，因为硬件不支持。**

入口如下图所示，编译工程后在 Tool 栏打开模拟器，模拟器自动导入编译好的 UartTFT-II_Flash.bin 文件。必须要编译工程后模拟器才能自动找到当前工程 UartTFT-II_Flash.bin 的路径。

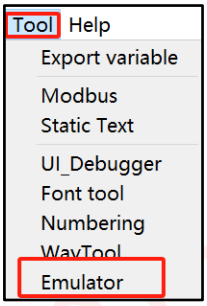


图 9-2：模拟器入口一

或者在软件安装包内单独打开模拟器，导入需要查看的工程。

图 9-3：模拟器入口二

点击进入软件后，如果出现下图的提示，表示没有安装视频解码器，可以点击 Yes 安装视频解码器，再重新打开软件。安装教程可以参照链接

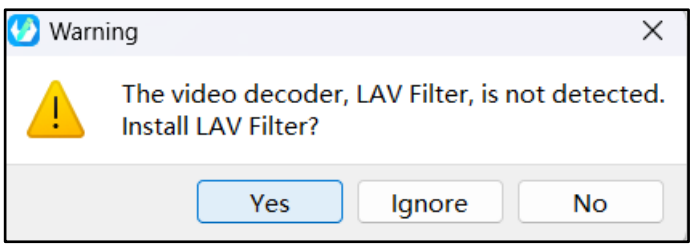


图 9-4：视频解码器安装提示

软件主界面如下图所示：主要包括功能栏、工程查看与操作区、信息区、模拟器变量区和操作记录栏。

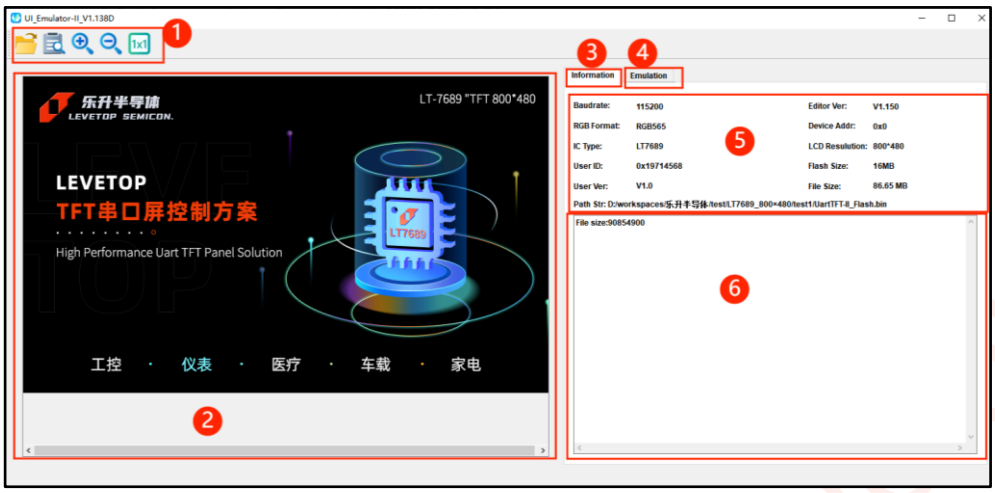


图 9-5：UI_Emulator-II 主界面

1、功能栏：可以导入工程、查看工程设置、放大或缩小工程界面。

2、工程查看与操作区：在此区域可以查看工程，用鼠标左键点击模拟触控操作。

3、信息区：点击查看部分工程信息。

4、模拟器变量区：点击查看**模拟器变量操作。**

5、信息显示：模拟器将一部分工程信息读取显示在标注 5 的位置。

6、操作记录栏：可查看导入记录。

9.1.2. 模拟器的变量操作

编译工程成功后点击 Emulation 后，跳转到以下页面。

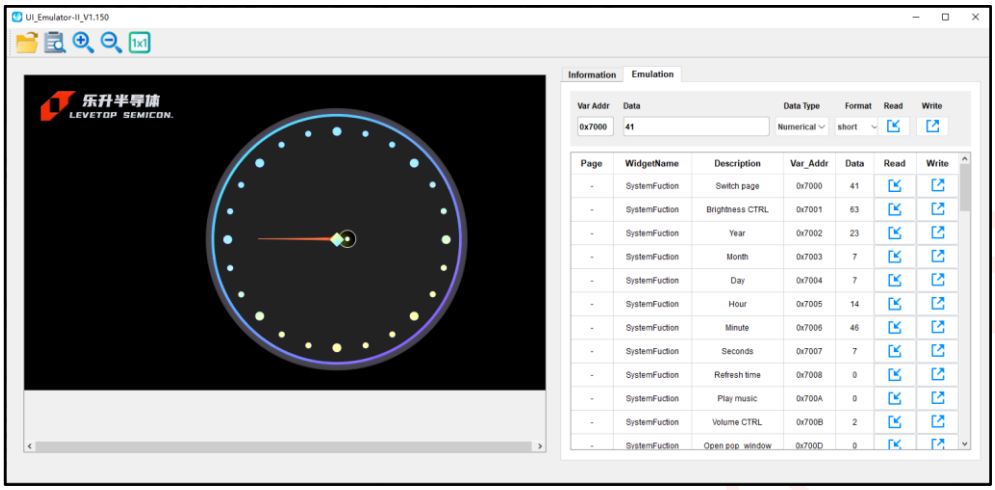


图 9-6：模拟器的变量操作栏

9.1.2.1. 手动输入框

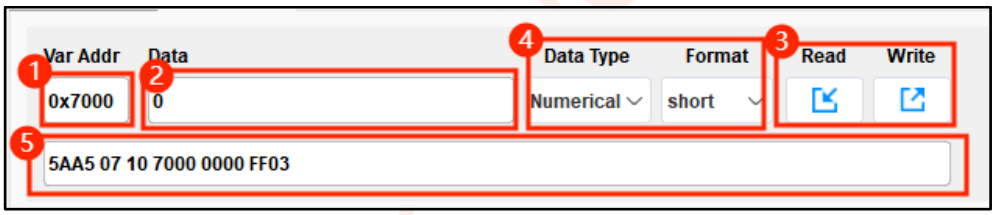


图 9-7：手动输入框

1、**变量地址**：输入需要读或者写的变量地址。

2、**变量数据输入（返回）框**：在此输入需要发送的变量数据，读回来的变量数据显示在该位置，变量数据可以是十进制的数据，也可以是十六进制数据，如果输入的是十六进制数据，则需要在数据前面加上 **0x**。

3、**Read/Write**：对变量地址进行读或者写，但是不能对多个变量地址一次性进行读写操作

4、**数据类型选择以及编码类型选择**：当数据类型选择的是基本数据类型时，对应的 Format 出现的是 7 个基本的数据类型，可以结合不同的空间进行参数的设置。当数据类型选择的是 String 类型，对应的 Format 出现的就是编码类型。发送数据的时候，数据类型选择需要结合控件的参数设置，详情请参考模拟器发送数据参考。

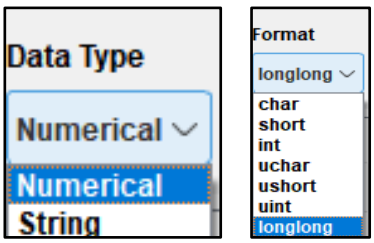


图 9-8：基本数据类型选择

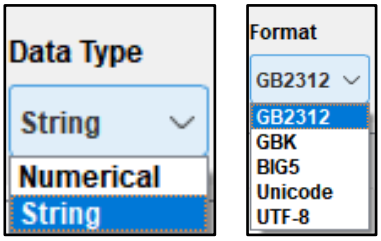


图 9-9：String 数据类型选择及编码类型选择

5、**串口指令预览框**：执行操作时，该框会生成对应的串口指令，可以复制该指令通过串口调试工具与串口屏进行通信，不能通过该框对模拟器进行操作。

9.1.2.2. 工程地址列表

该列表仅包含显示控件，不包括带触控属性的控件。其界面图如下：

Page	WidgetName	Description	Var_Addr	Data	Read	Write
-	SystemFuction	Year	0x7002	23		
-	SystemFuction	Month	0x7003	7		
-	SystemFuction	Day	0x7004	13		
-	SystemFuction	Hour	0x7005	9		
-	SystemFuction	Minute	0x7006	19		
-	SystemFuction	Seconds	0x7007	8		
-	SystemFuction	Refresh time	0x7008	4		
-	SystemFuction	Play music	0x700A	0		
-	SystemFuction	Volume CTRL	0x700B	2		
-	SystemFuction	Open pop_window	0x700D	0		
-	SystemFuction	Auto-dimming	0x700E	1		
-	SystemFuction	Dimming level	0x700F	20		
-	SystemFuction	Time to enter sleep	0x7010	60		

图 9-10：工程地址列表

(1) Page 列：控件所在的页。仅供查看，不可修改。右击 Page 的页码出现 goto page 按钮，点击可以使得模拟页面跳转到对应的页面（**注意：**如果 Page 没有页码右击无效）。

(2) WidgetName 列：控件编号（SystemFuction：系统功能，即特殊寄存器）。仅供查看，不可修改。

(3) Description 列：控件的自定义描述，可在编辑工程时设置。仅供查看，不可修改。

(4) Var_Addr 列：控件地址。仅供查看，不可修改。双击对应的控件地址，可以把该控件信息加载到最上面，加载的信息包括 Var Addr，Data，dataType，Format。

(5) Data 列：在此可查看对应地址数据；双击也可以填入数据。数值输入为 10 进制，字符串数据直接输入，无需转码。

(6) 数据读写控制：点击  可将 Data 列数据写入模拟器；点击  可将模拟器内对应地址的内容读取并显示在 Data 列。

9.1.3. 模拟器写数据举例

1、写变量数据，其步骤如下：

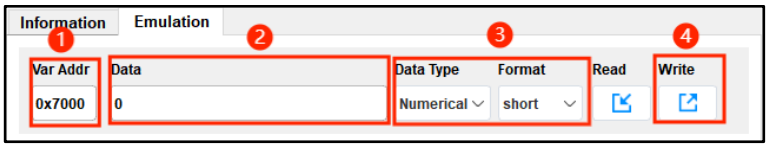


图 9-11：模拟器写变量

(1) 确定需要修改的变量地址。

(2) 输入变量数据，输入数据为 10 进制或者 16 进制，输入为 16 进制时数据前面要加上 **0x**，如 0x1234。

(3) 根据关联该地址的控件选择数据类型，若该地址无关联控件，则默认选择 ushort。

(4) 点击  写入变量。

注：写数据的对象为带小数部分的字库数字和图片数字控件时，若控件设置 3 位整数 2 位小数，想显示 123.45，则输入框直接输入 12345 然后发送，不能输入 123.45。

2、写字符串数据，其步骤如下：

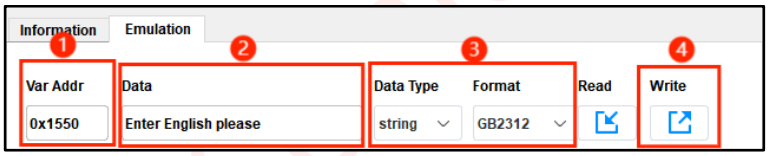


图 9-12：模拟器写字符串数据

(1) 确定需要修改的变量地址。

(2) 输入字符串，软件在发送时会自动在字符串末尾加上结束符。

(3) 根据控件使用的字库确定编码类型。

(4) 点击  写入字符串。

9.1.4. 模拟器模拟编码器的使用

编码器的操作可以用以下按键模拟，仅在当前页为编码器页面可以操作。

方向键左键：模拟编码器旋钮逆时针旋转，变量减。

方向键右键：模拟编码器旋钮顺时针旋转，变量增。

数字按键 1：模拟编码器点按。

数字按键 2：模拟编码器双击。

数字按键 3：模拟编码器长按。

9.1.5. 模拟器显示带旋转角度的工程的注意事项

因为模拟器不支持显示带旋转角度的工程，如果需要显示带旋转角度的工程，需要对该工程作以下设置然后再编译导入模拟器显示。

1、将工程设置的 **Rotate**（角度）设置成 **0 Degree** 。

2、若原工程旋转角度为 0°或者 180°，则工程设置的屏幕分辨率 **X-Pixel** 和 **Y-Pixel** 保持原样；若原工程旋转角度为 90°或者 270°，则工程设置的屏幕分辨率 **X-Pixel** 和 **Y-Pixel** 数值互换。例子如下图所示

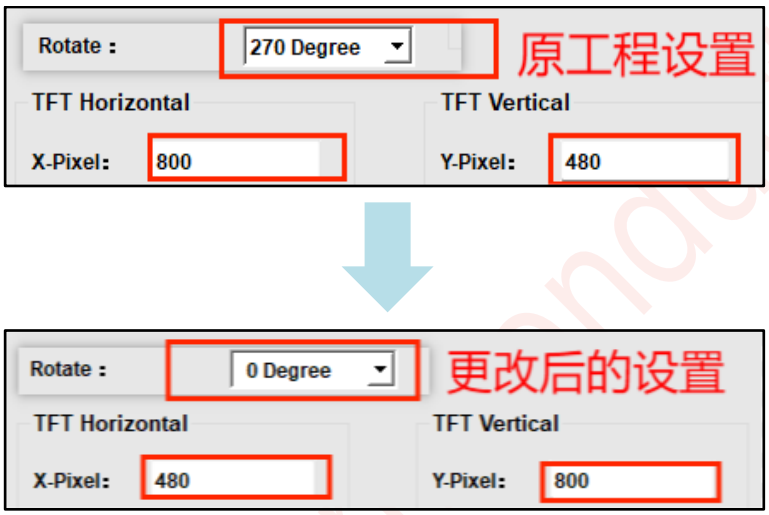


图 9-13：模拟器旋转角度设置

9.1.6. 模拟器的限制说明

1、暂不支持发送曲线数据显示。

2、暂不支持按键音，即 Key with beep 设置。

3、模拟器的视频播放功能需要视频解码器 LAV Filters 的支持，因此使用前需要安装好解码器。

4、不支持串口命令控制模拟器显示。

9.1.7. 模拟器写数据参考

注： “ - ”代表没有或者不能发送。

表 9-1：模拟器发数据参考表

控件名称	占用变量空间长度/Bytes	数据类型选择	控件名称	占用变量空间长度/Bytes	数据类型选择
按键	-	-	数字时钟	-	-
弹窗	-	-	动图	2	ushort
变量调节	控件设置数据类型	-	编码器	2	-
多变量操作	-	-	音频	-	-
环形滑条	2	-	滑条	2	-
键盘键值	-	-	位元状态	2	ushort
数字键盘	控件设置数据类型	-	小图标	2	ushort
字符串显示	(wordLength+1)*2	String	计时器	2*3（需要 3 个变量）	ushort
图片数字	控件设置数据类型	跟随控件	静态文本	-	-
字库数字	控件设置数据类型	跟随控件	自动变量	控件设置数据类型	跟随控件

9.2. 串口调试工具（UI_Debugger-II）使用说明

串口调试工具入口如下：

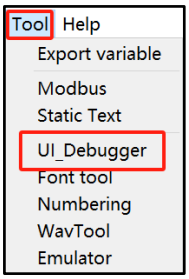


图 9-14：串口调试工具入口

9.2.1. 软件主界面

打开软件后，其主界面如下图所示。具体软件版本以下载到的软件为主。软件主界面主要包括以下内容：

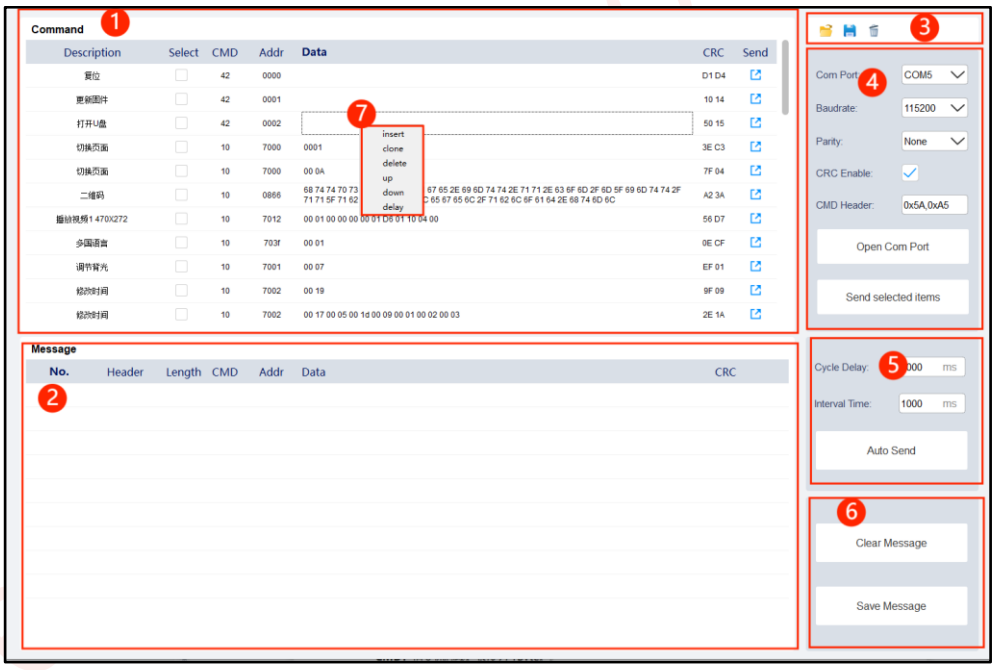


图 9-15：软件主界面

1、指令编辑区

Description：描述项，可在此处添加关于指令的描述。

Select：勾选项，勾选后的指令可以进行 Send All、Star Loop 操作。

CMD：指令项，10 表示写入数据、03 表示读取数据，长度为 1Byte。

Addr：地址项，目的变量地址，即对该地址读或者写。长度 2Bytes。

Data：数据项，需要写入的数据或者是操作的参数。长度是 2*n Bytes。

CRC：校验码项，用于检验数据。长度 2Bytes。（填写指令项、地址项和数据项后自动生成）。

Send：发送按钮，点击发送此行的指令。

2、信息区：收发指令都可以在此查看。黑色指令为发送出的指令，蓝色指令为接收到的指令。

NO：信息排序。

Header：帧头项。长度 2Bytes。

Length：长度项，指令长度。长度 1Byte。

CMD：指令标志项。长度为 1Byte。

Addr：目标地址。

Data：数据项，返回数据。长度是 2*n Bytes。若是上面发的是写指令则返回的数据为 0xFF，若上面发的是读指令，则读反馈指令数据为 0xFF，读指令返回指令的数据为**读指令的所需要的数据长度+变量地址对应的数据。**

CRC：校验码项，用于检验数据。长度 2Bytes。

3、功能区，从左到右分别为，

 ：载入 txt 格式的指令原始数据列表文件，不可导入信息文档，见下方 **Save Message。**

 ：保存 txt 格式的指令列表文件

 ：清空指令编辑区的所有指令

4、配置信息区

Com Port：选择通信端口。

Baudrate：选择波特率，需要与工程设置的波特率对应，支持用户自定义波特率，如下图，选择 custom 选项可以填写用户自定义波特率。

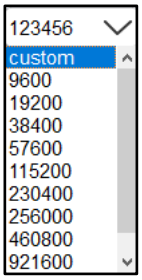


图 9-16：用户自定义波特率

Parity：奇偶校验，需要与工程设置的 Parity 相同。

CRC Enable：：是否进行 CRC 校验，若工程勾选 No CRC，此处不用勾选。

CMD Header：通信帧头，需要与工程设置的 User Start Bytes 相同。

Open Com Port：开启端口。开启端口后才能发送指令。

Send selected items：按照从上到下的顺序发送全部 select 项已勾选的指令

5、循环配置区

Cycle Delay：循环发送时每次循环之间的时间间隔。

Interval Time：循环时相邻指令之间发送的时间间隔。

Auto Send：按照从上到下的顺序循环发送全部 select 项已勾选的指令。

6、信息操作区

Clear Message：清空信息接收区的信息。

Save Message：保存信息接收区的信息为 Txt 文档。

7、指令行处右键呼出菜单项

insert：在所选指令行上方插入一行空白的指令行。

clone：在选中指令行的下方克隆一行选中的指令。

delete：删除所选指令行。

up：将所选指令行上移。

down：将所选指令行下移。

Delay：在所选指令行上方插入发送延时行（时间单位为 ms）。

8、在 Select 列右键点击可以全选或全不选指令，如下图

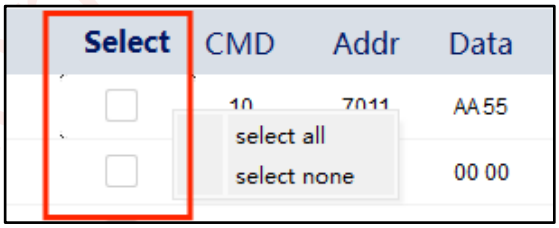


图 9-17：指令全选功能

9.2.2. 指令发送使用教程

1、发送单条指令：其步骤如下：

（1）配置信息，选择对应端口、对应的波特率以及奇偶校验，确定帧头对应，勾选 CRC Enable，然后打开端口（Open Com Port）。

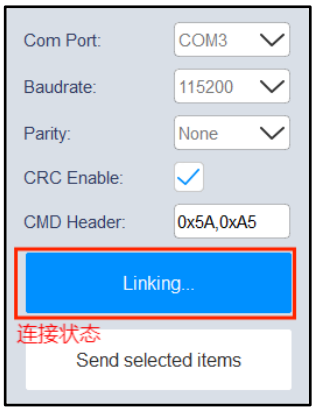


图 9-18：软件参数配置

（2）添加指令，双击指令行的单元格，对指令进行编辑。或点击右上角的  ，载入已有的指令 txt 文件。

（3）发送指令，单击发送，即可发送单条指令。

Description	Select	CMD	Addr	Data	CRC	Send
		10	7011	AA55	11 99	
切换页面		10	7000	00 00	FF 03	
发送数据至0901		10	0901	00 20	B6 47	
读取0901和0902两个变量地址的内容		03	0901	00 02	B3 9D	
调节背光		10	7001	00 2D	6E DE	
缓冲曲线1		10	C001	00 C8 00 64 00 C8 00 64 00 C8 00 64	66 42	
缓冲曲线2		10	C002	00 C8 00 64 00 C8 00 64 00 C8 00 64	63 81	
清除曲线1和2		10	E003		79 C4	

图 9-19：发送指令

（3）随后可在信息区获取到发送与接收指令的信息。

No.	Header	Length	CMD	Addr	Data	CRC
Uart					Insert COM3	
1	5AA5	07	10	7000	00 00	FF 03
2	5AA5	04	10		FF	FB 6B

图 9-20：指令信息区

2、一次发送多条指令与循环发送指令，其步骤如下：

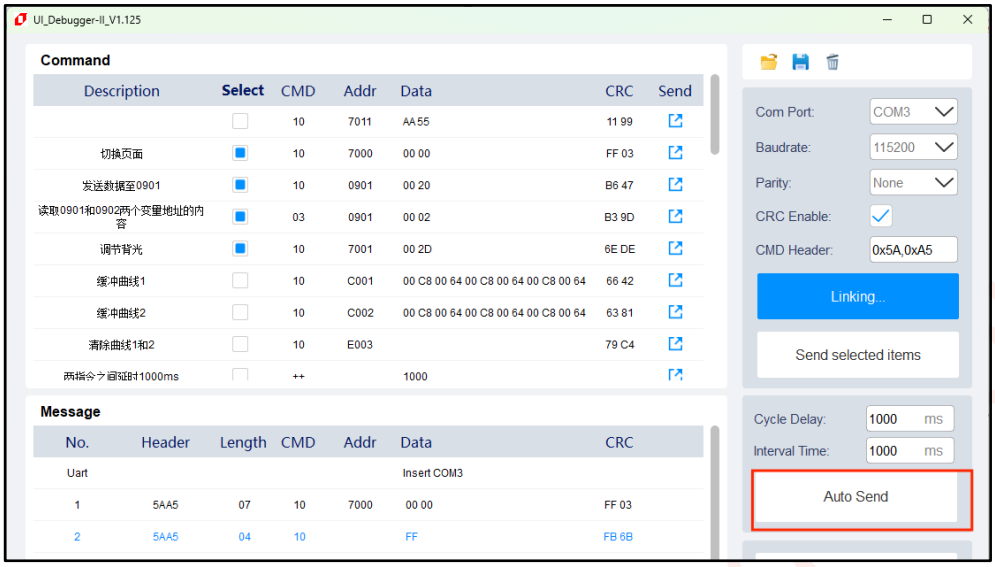


图 9-21：一次发送多条指令

(1) 右键调整指令上下位置（指令是按从上到下的顺序发送的）。

(2) 在 Select 项勾选所要发送的指令。

(3) 点击 Send selected items 或 Auto Send，即可一次发送多条指令与循环发送指令。

(4) 若要调整循环发送的时间间隔，可调整 Circle Delay 与 interval。

(5) 在 loop 模式下，可以自定义添加 delay 时间其中++为 delay 标志（不可修改），可在 data 栏添加间隔时间（ms），希望启用该延时需勾选该行的 select 项。

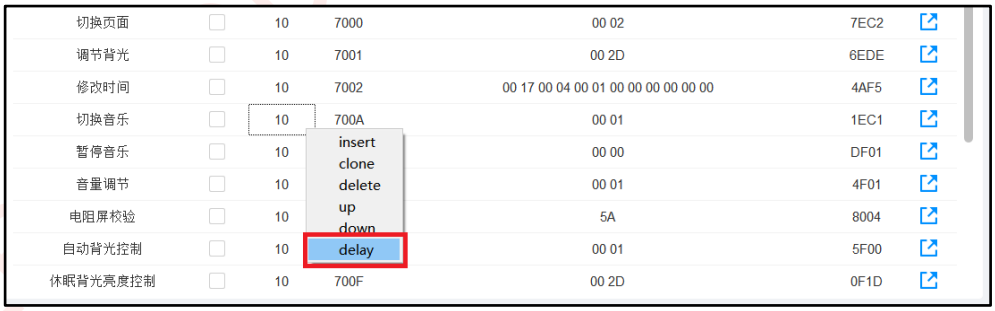


图 9-22：延时指令

System Function	Select	Order	Addr	Data	CRC	Send
切换页面		10	7000	00 02	7EC2	
调节背光		10	7001	00 2D	6EDE	
修改时间		10	7002	00 17 00 04 00 01 00 00 00 00 00 00	4AF5	
Daley(ms)		++		1000		
切换音乐		10	700A	00 01	1EC1	
暂停音乐		10	700A	00 00	DF01	
音量调节		10	700B	00 01	4F01	
电阻屏校验		10	700C	5A	8004	
自动背光控制		10	700E	00 01	5F00	

图 9-23：设置延时时间

延时时间为 10 进制，单位为 ms。延时指令只由**指令标志“++”**和**延时时间**组成，其他部分不需要填写。延时指令穿插在两条正常指令之间，**延时时间 + Interval Time** 是两条指令之间的发送间隔。

其他具体串口指令说明详情，请参考串口通信指令。

9.2.3. 指令文档说明

图 9-24：导入导出普通指令

指令在指令文档内的格式如下表，可以在 UI_Debugger-II 内编辑后导出，也可以在 txt 文档手动编辑，需注意编辑指令文档时**不可留下空白行。**

指令示例如下：

```
发送数据至0901,unselect,10 0901 00 20
读取0901和0902两个变量地址的内容,unselect,03 0901 00 02
切换页面,unselect,10 7000 00 02
调节背光,unselect,10 7001 00 2D
缓冲曲线1,unselect,10 C001 00 C8 00 64 00 C8 00 64 00 C8 00 64
缓冲曲线2,unselect,10 C002 00 C8 00 64 00 C8 00 64 00 C8 00 64
清除曲线1和2,unselect,10 E003
两指令之间延时1000ms,unselect,++  1000
修改时间,unselect,10 7002 00 17 00 04 00 01 00 00 00 00 00 00
切换音乐,unselect,10 700A 00 01
暂停音乐,unselect,10 700A 00 00
音量调节,unselect,10 700B 00 01
电阻屏校验,unselect,10 700C 00 5A
自动背光控制,unselect,10 700E 00 01
休眠背光亮度控制,unselect,10 700F 00 2D
自动背光休眠时间控制,unselect,10 7010 00 0A
```

图 9-25：指令文档内容

9.2.4. 指令信息文档说明

点击 **Save Message** 可将信息区的指令收发记录保存为 txt 文档，文档内容示例如下：

文档内的指令均为完整指令，包括帧头、长度等部分。注意该文档不能重新导入 UI_Debugger-II 。

```
1, 5A A5  07  10 0901  00 20  B6 47
2, 5A A5  04  10   FF  4C 30
3, 5A A5  07  03 0901  00 02  B3 9D
4, 5A A5  04  03   FF  41 00
5, 5A A5  0B  03 0901  00 02 00 20 00 00  B6 A0
6, 5A A5  07  10 7000  00 02  7E C2
7, 5A A5  04  10   FF  4C 30
8, 5A A5  07  10 7001  00 2D  6E DE
9, 5A A5  04  10   FF  4C 30
10, 5A A5  11  10 C001  00 C8 00 64 00 C8 00 64 00 C8 00 64  66 42
11, 5A A5  04  10   FF  4C 30
12, 5A A5  11  10 C002  00 C8 00 64 00 C8 00 64 00 C8 00 64  63 81
13, 5A A5  04  10   FF  4C 30
14, 5A A5  05  10 E003   79 C4
15, 5A A5  04  10   FF  4C 30
```

图 9-26：指令信息记录文档

9.3. 字库取模工具使用说明

字库取模工具的入口如下：

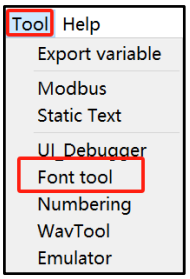


图 9-27：字库取模工具入口

点击打开字库取模工具，弹出下图界面。

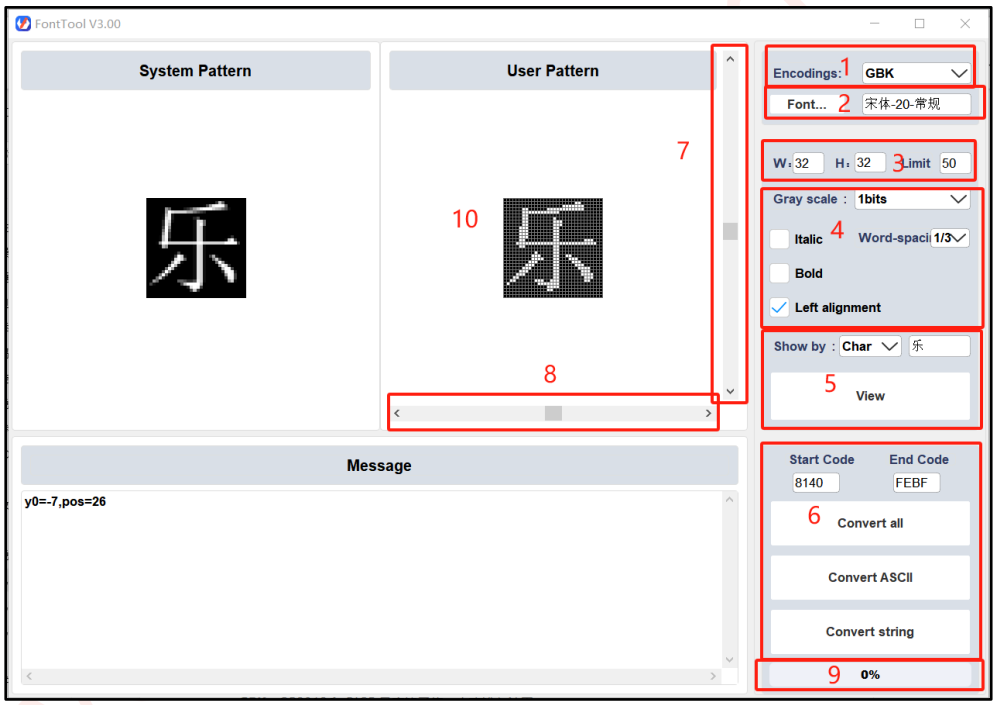


图 9-28：字库取模工具主界面

1、点击选择字库编码类型，如下图所示支持以下编码。

GB2312：支持简体汉字。

BIG5：支持繁体汉字。

GBK：GB2312 和 BIG5 是它的子集，中文输入法用。

Unicode：万国码，包括世界上大部分有文字的语言。Unicode 字库每个字符都自带宽度。

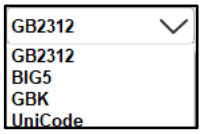


图 9-29：编码类型选择

2、点击 Font 弹出以下图窗口，在此可以选择字体和字号。

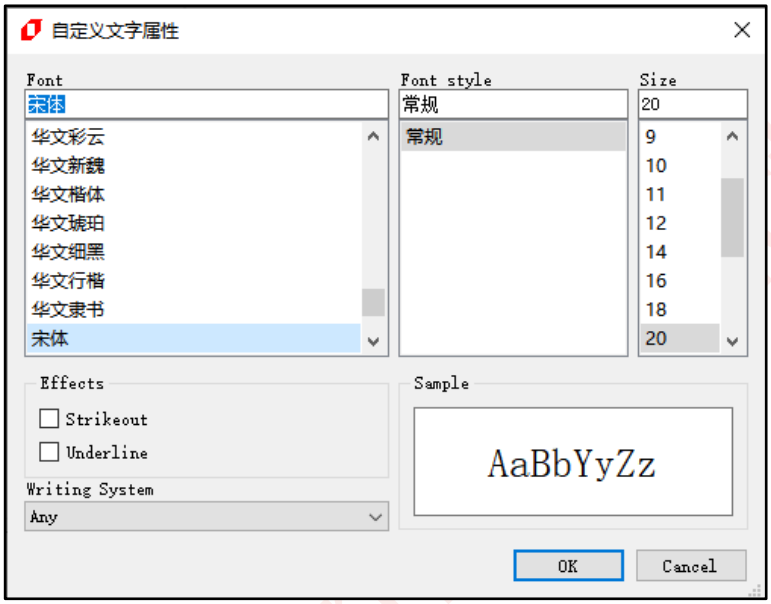


图 9-30：字体字号选择

3、点击输入数字修改文字外框的大小，宽和高可以相等或不等，取决于使用者的需求效果，单位是像素。Limit 默认即可，字体越大且 Limit 值越小，文字就相对丰满一点。

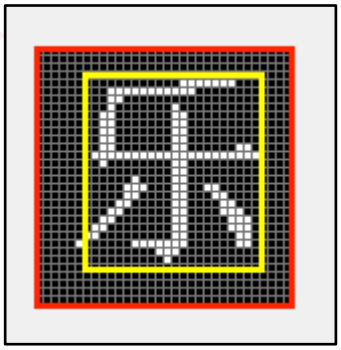


图 9-31：字符示例

注：文字外框指的是上图整个红线包括的区域，字体大小的设置只会影响到黑框内黄色区域文字的大小。所以设置文字的像素和字体大小时，必须要处理好两者的大小关系，否则会出现文字过大失真或者字体过小模糊等问题。

4、Gray scale：灰阶，可选 1、2、4bit，灰阶越多显示时效果越好，相应的字库文件越大。注意产生字库时不能选择 8bits 或αRGB4444。

Italic：斜体。

Bold：加粗字体。

Left alignment：字符左对齐。

Word-spacing：空格宽度，例如字库宽度选择 32，空格宽度选择 1/2，则空格宽度为 16pixel。

5、点击选择显示字符还是字符编码，输入需要显示的字符或者字符编码，点击 View 预览该字符的显示效果。更改 3、4、5 这三个部分的参数时，都需要点击 View 才能看到效果。

6、Start code 和 End code：对于 GBK，GB2312，BIG5 这三种编码，这两个参数默认即可。如果是 Unicode 编码，则需要去网上查询对应语言的对应编码分区进行设置。

Covert：生成整个字库，包括 ASCII 码和中文字符。

Covert ASCII：只生成 ASCII 码的字库。

ConVert string：自定义字库。

7&8、微调：通过滑条和滑条两端的箭头精确地调节文字在文字框内地位置，以达到较好的显示效果。

9、进度条：字库制作进度条。显示字库生成的进度。

字库制作过程，如下图：

1、选择字库编码、字体和合适的字号。

2、确定文字外框的宽高（即字库宽高）。

3、选择 bit 数（灰阶）。

4、点击 View，查看文字位置是否合适，不合适则微调位置。

5、根据需要的字库内容，选择生成全部字库，或者是 ASCII 字库。

6、选择合适的保存路径，按照命名规则给字库命名（**命名不能存在’ * ‘，输入关键信息如编码类型字库宽高等作为命名**），点击保存，字库制作完成。

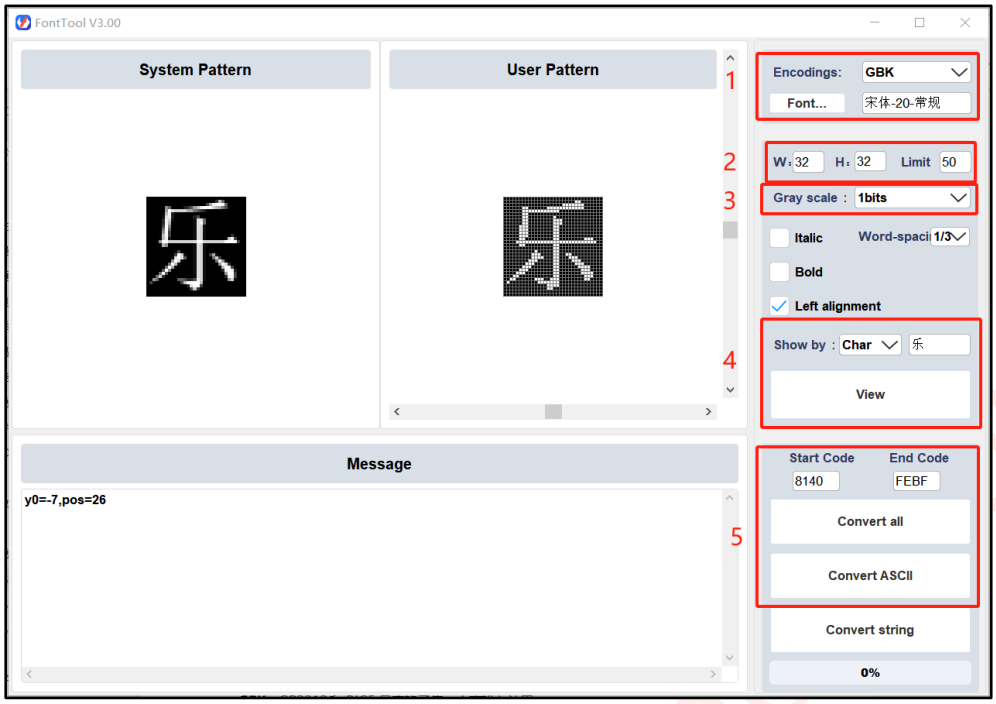


图 9-32：字库制作过程 1

图 9-33：字库制作过程 2

自定义字库制作过程，如下图：

1、区别于全字库，自定义字库**只能选 Unicode 编码格式**，字体和字号可自行选择合适的。

2、确定文字外框的宽高（即字库宽高）。

3、选择 bit 数（灰阶）。

4、点击 View，查看文字位置是否合适，不合适则微调位置。

5、选择提前准备好的自定义字库 txt 文档。

6、选择合适的保存路径，按照命名规则给字库命名（**命名不能存在' * '，输入关键信息如编码类型字库宽高等作为命名**），点击保存，自定义字库制作完成。

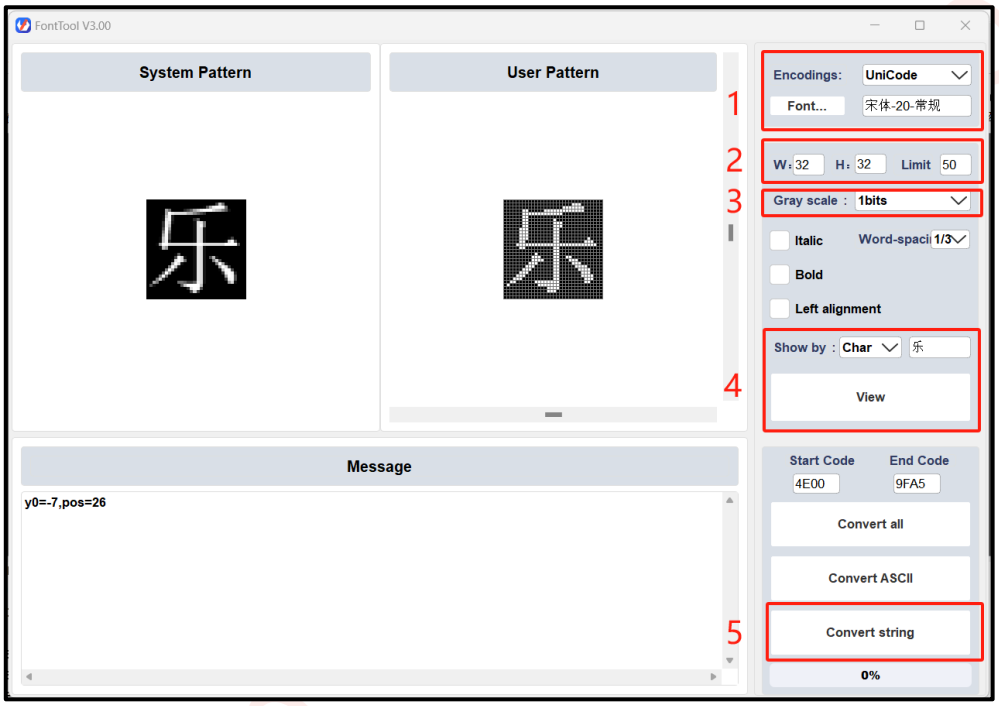


图 9-34：自定义字库制作过程 1

图 9-35：自定义字库制作过程 2

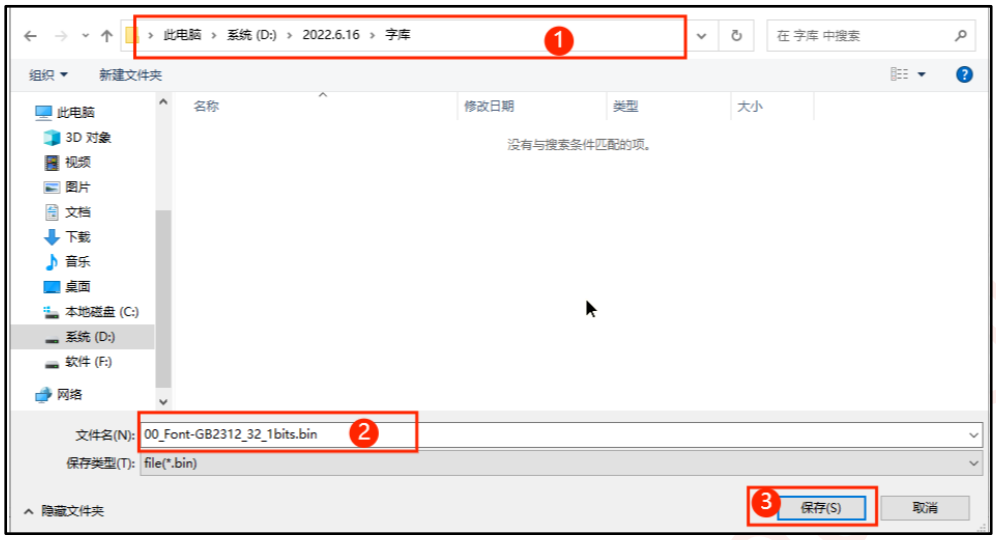


图 9-36：自定义字库制作过程 3

9.4. 图片编号工具使用说明

以下是图片编号工具的入口：

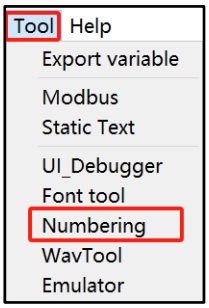


图 9-37：图片编号工具入口

点击打开编号工具，主界面如下图所示。该工具操作对象为图片，包括 Picture 和 Icon 两个文件夹的素材。

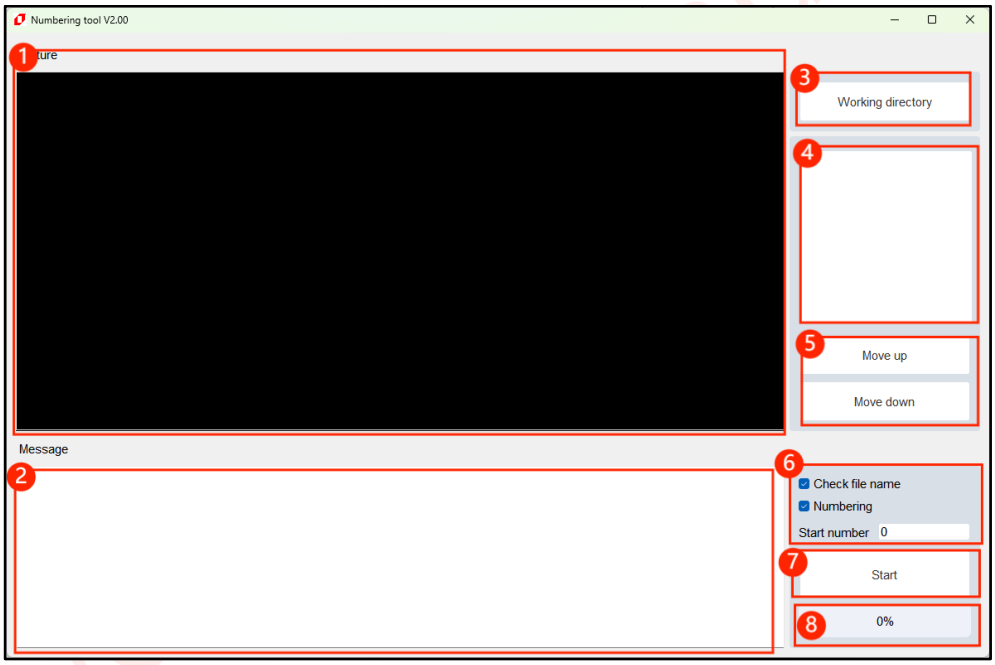


图 9-38：图片工具主界面

功能介绍：

1、图片查阅区，在此区域查看选中的图片。

2、操作信息区，编号没成功时返回信息。

3、Working directory ：工作目录，点击添加需要快速编号的图片。注意，添加时会将整个文件夹内的图片导入。

4、图片目录，在此点击图片名称可选中图片并查看。

5、Move up：将选中的图片上移一个位置。

Move down：将选中的图片下移一个位置。

6、功能设置区：

Check file name：勾选后，软件会自动校正不符合 UI_Editor-Lite 的图片名称。

Numbering：勾选后，软件会对图片进行编号。

Start number：列表内图片的起始编号

7、Start：开始按键，点击后执行功能设置区预设的操作。

8、处理进度条展示。

使用图片编号工具对图片进行编号的步骤如下：

1、点击 Working directory。如下图，打开需要排序的图片所在的文件夹。随便选中一张，点击打开。

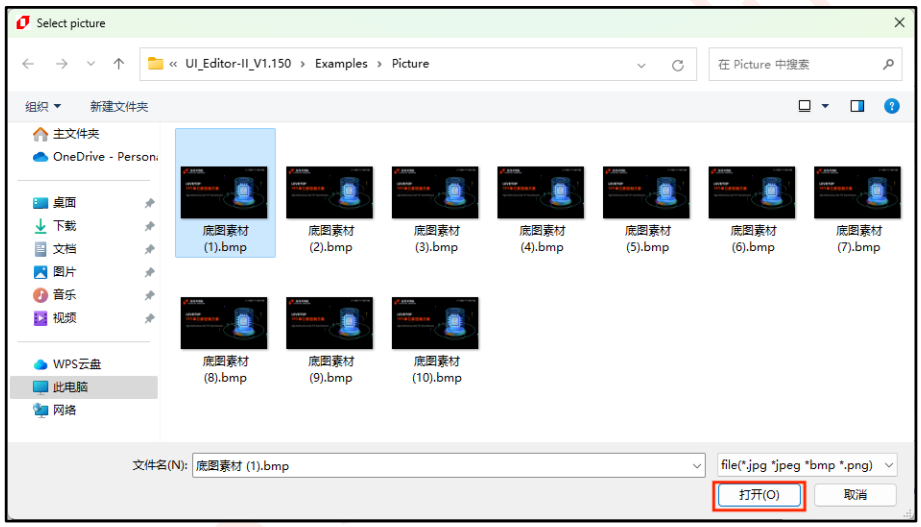


图 9-39：选择图片

2、点击打开后，工具主界面如下图所示，可以看到，右边图片列表显示的是当前图片的排列顺序，这个顺序从上到下对应编号 0000 ~ NNNN。如果这个顺序不是我们需要的顺序，可以通过 Move up 和 Move down 来调整顺序。先点击选中需要变换顺序的图片，上移点击 Move up ；需要下移则点击 Move down。

图 9-40：顺序调整

3、点击 Start，执行预设的操作。

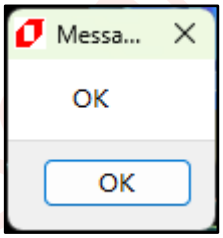


图 9-41：编号完成

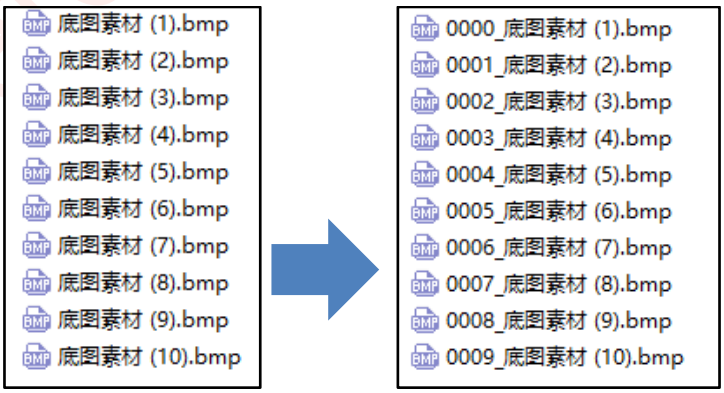


图 9-42：自动编号效果展示

9.5. 音频转换工具使用说明

9.5.1. 制作 Wav 文件

如果音频文件不是 Wav 文件或者采样率不是 22050，那就需要先把音频文件转换成符合条件的 Wav 文件,下面以“格式工厂”免费版为转换平台进行操作。

1、首先打开软件，选择音频，选择“-> WAV”，进入添加文件界面。

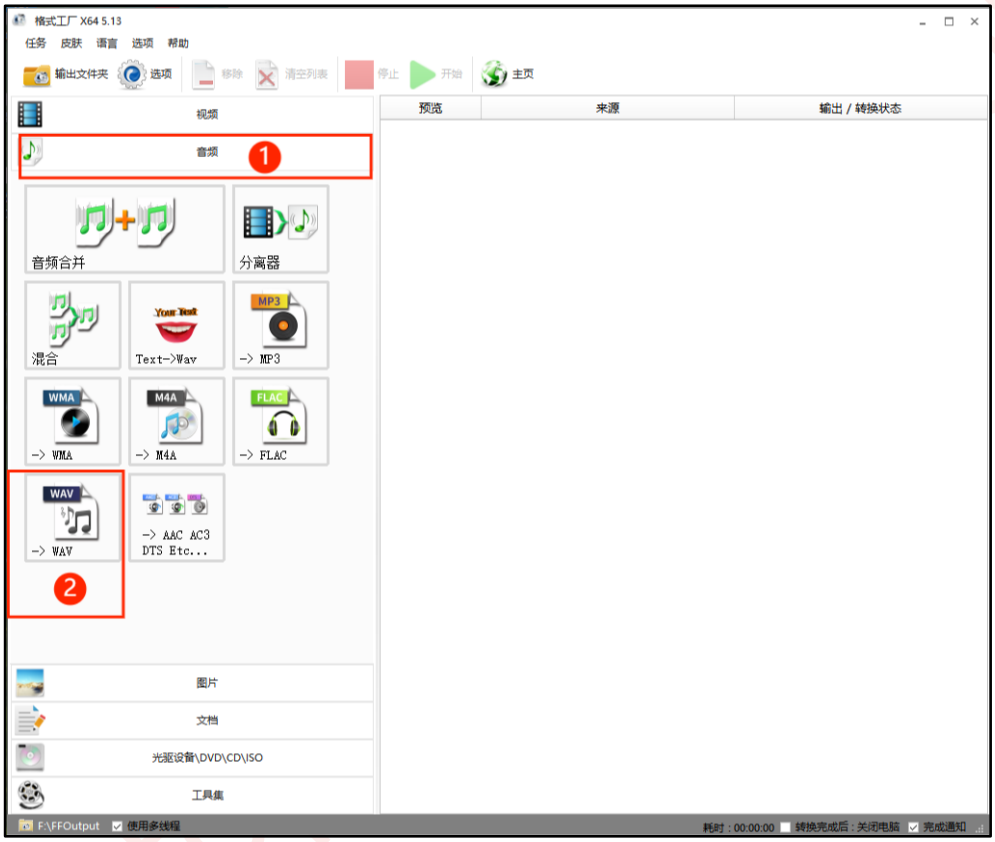


图 9-43：格式工厂主界面

2、如下图，点击标注 1 可以选择 Wav 文件的输出目录，点击标注 2 可以添加需要转换格式的文件，点击标注 3 可更改输出配置，点击添加文件。

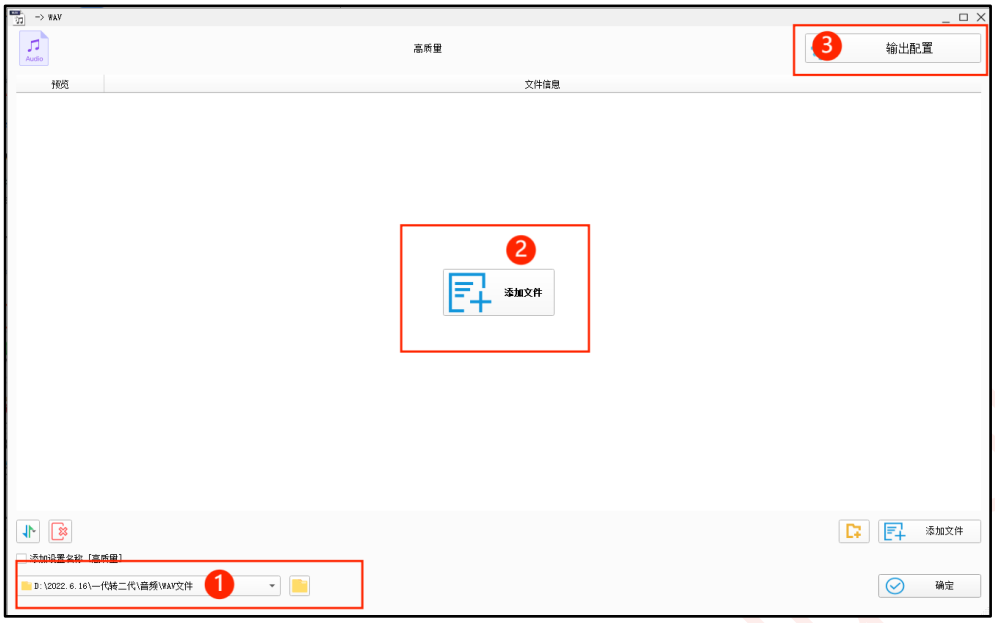


图 9-44：添加音频文件

3、需要修改采样率时，点击上图的输出配置，选择低质量输出，将采样率选择为 22050，声道可以选择单声道，单声道的噪音相对较少一点，然后点击确定。

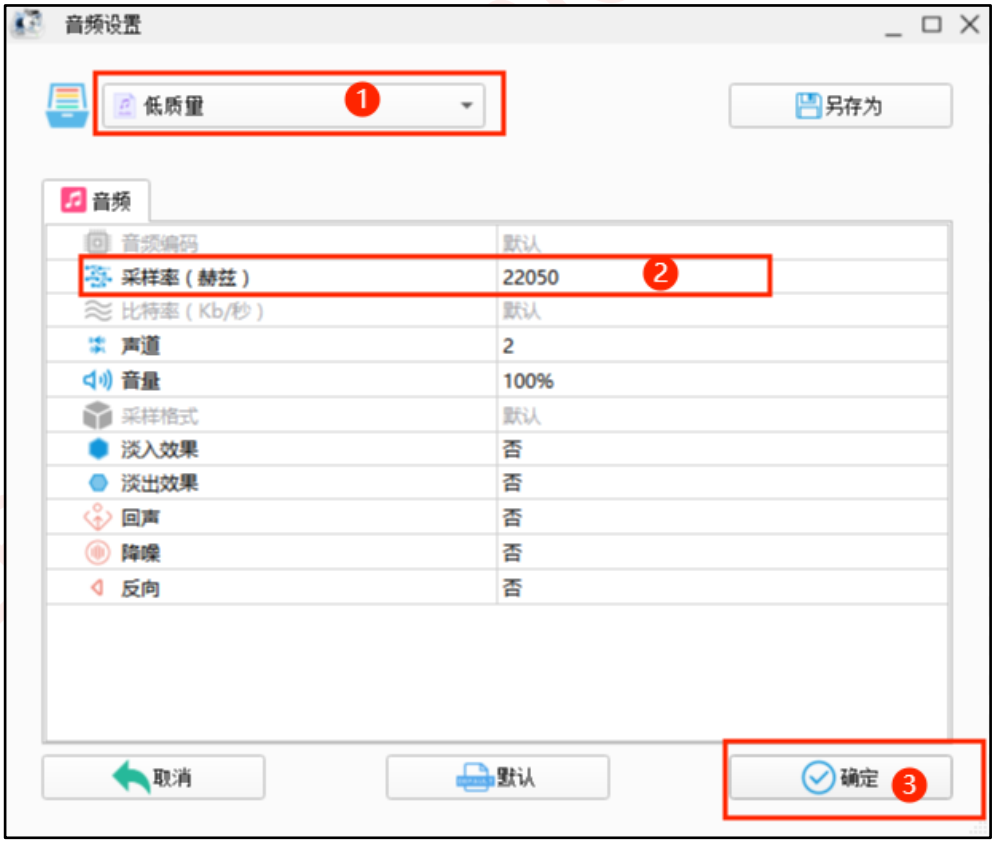


图 9-45：输出配置

4、添加音频后，如果需要剪辑音频，我们可以点击音频列表内的音频进入剪辑模式，如下图所示。标注 1 选择截取片段的起始节点，标注 2 选择截取片段的终止节点，最后点击确定，退出剪辑界面。然后再点击上图右下角确定按键

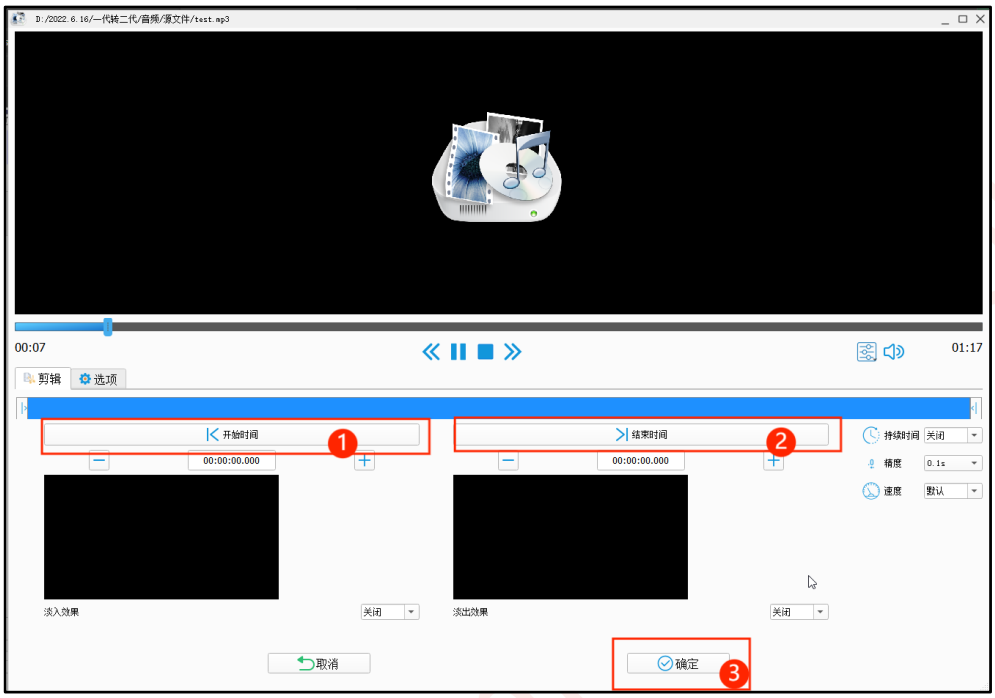


图 9-46：截取音频

5、截取音频后，点击开始转换。

图 9-47：开始转换

6、等待其转换完成。

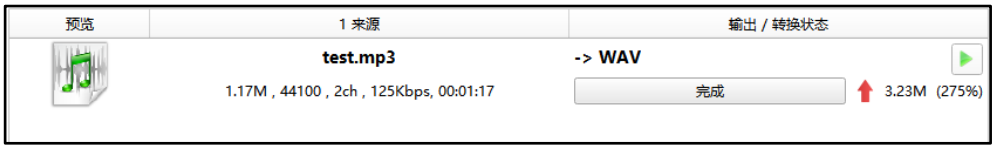


图 9-48：转换完成

9.5.2. Wav 转 bin 文件

在 UI_Editor-Lite 内打开 Wav Tool，

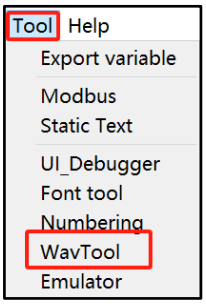


图 9-49：打开音频转换工具

主界面如下图所示：

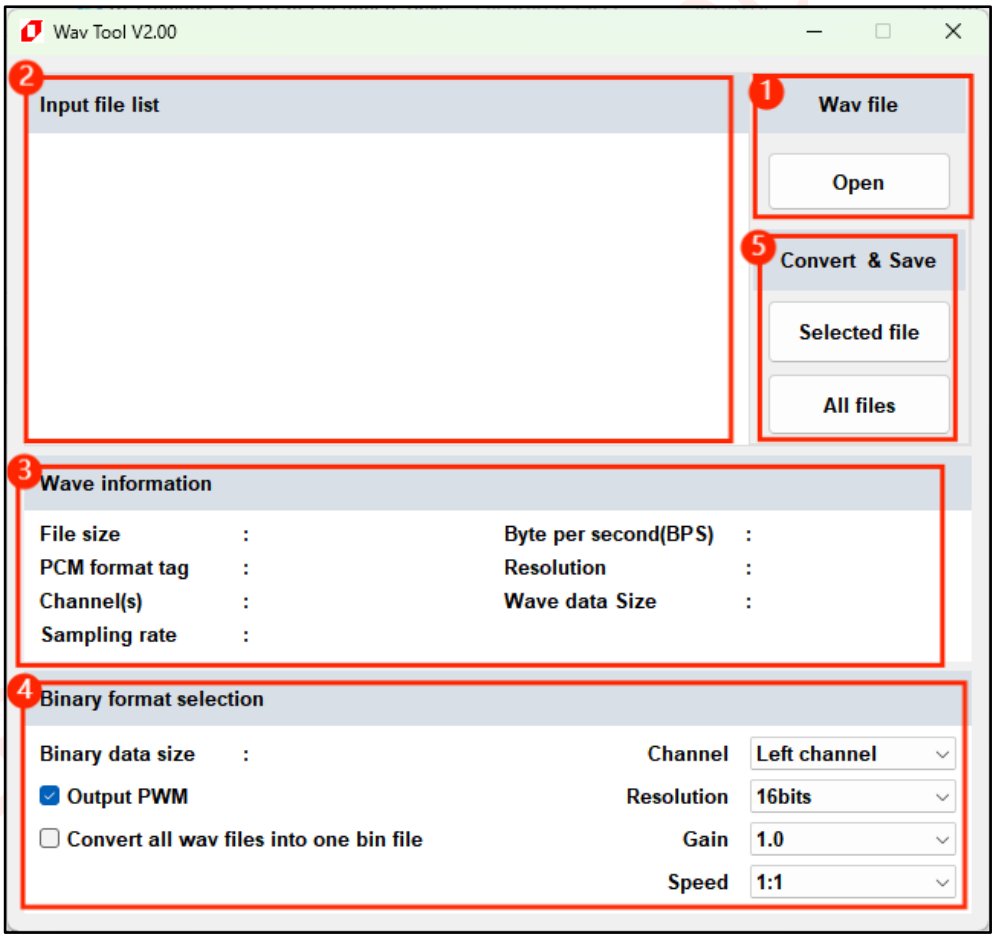


图 9-50：音频转换工具主界面

参数说明：

1、Open：导入 wav 文件。

2、导入文件后显示列表。

3、**Wav information**：导入的 Wav 文件的参数信息，不可修改。

Sampling rate ：采样率，需确保为 22050。

其他参数不作要求。

4、Binary format selection：目标 bin 文件的参数，。

Binary data size：目标 bin 文件大小，无需修改。

Output PWM ：输出 PWM 值，无需修改。

Convert all wav files into one bin file ：勾选后会将所有的 wav 文件打包成 1 个 bin 文件。

Channels：声道选择，默认单通道。

Resolution：采样位数，需选择 16bits。

Gain：音量增益，默认 1.0。

Speed：播放速度，默认 1:1 即可。本软件非专业音频软件，随意调节参数会使得音频受损。

5、**Convert & Save**：生成和保存。

Selected file：生成当前选中的 wav 文件生成 bin 文件。

All files：所有 wav 文件都生成 bin 文件。

使用步骤如下：

1、点击 open 按键添加 Wav 文件，点击 open 后，弹出一个文件选择窗口，如下图所示，先找到目标 Wav 文件所处的文件夹，选中其中一个 Wav 文件，然后点击打开，即可导入整个文件夹内的 Wav 文件。

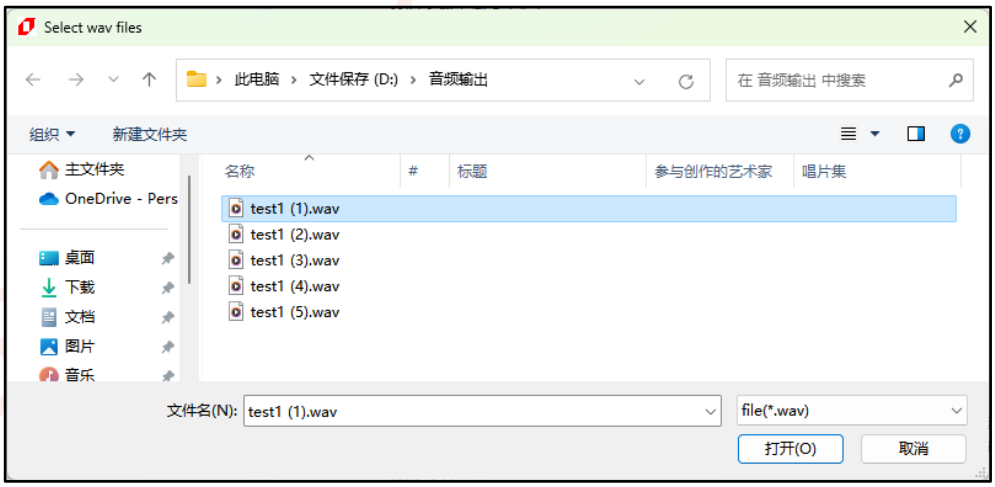


图 9-51：选择 Wav 文件

2、选择 Wav 文件后，如下图所示，可以看到列表中为已添加的 WAV 文件名称。如果采样率（Sampling rate 下图左边蓝色框）不是 22050，先按照制作 Wav 文件的步骤将采样率转换成 22050。确认无误后直接点击 Selected file 或者 All filesl。

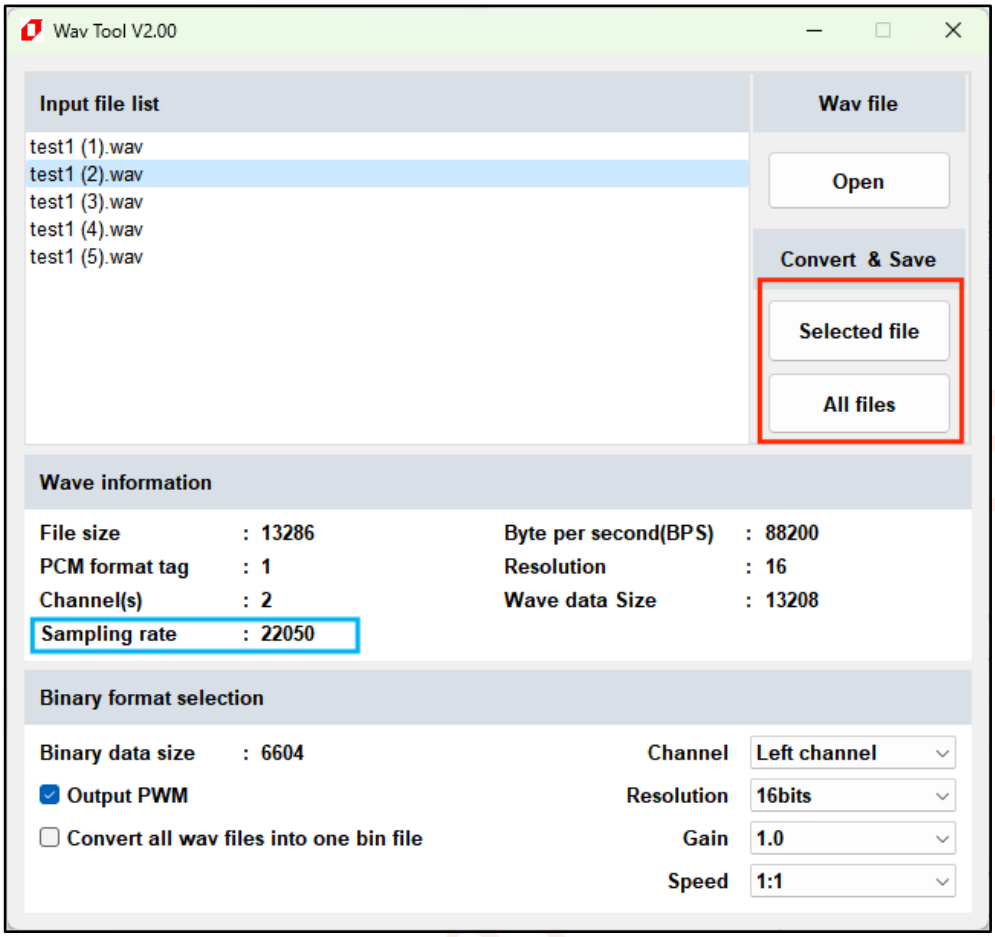


图 9-52：设置音频参数

3、最后再保存，如下图所示，保存时不能直接覆盖原有的音频 bin 文件，需要命名为新的名字。

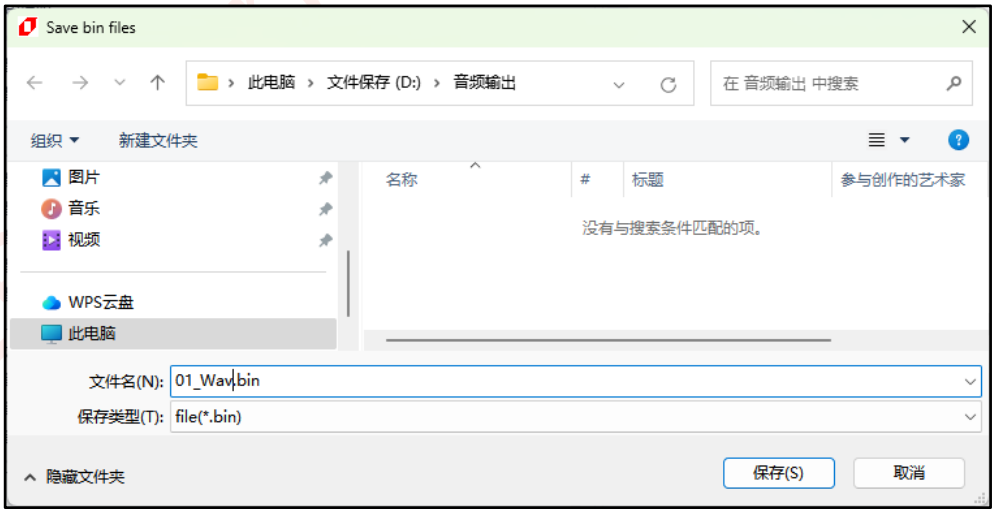


图 9-53：命名保存

注：保存时不能包含右边 10 个字符 \ / : * ? " < > |

10. 串口通信指令

串口通信指令分为三种，一种是写数据指令，即往串口屏指定的变量地址（寄存器）写数据；一种是读数据指令，即从串口屏指定的变量地址（寄存器）读数据；最后一种是反馈指令，包括读数据反馈、触控反馈和指令校验反馈。指令详情请参考下表**（其中 0x 表示十六进制数，发指令时需要删掉，直接发数据部分即可）**。当写指令发送，不管其 CRC 是否通过校验会返回相关的反馈指令

串口通信需要串口通信调试工具的支持，工具详情可参考串口调试工具（UI_Debugger-II）使用说明。

表 10-1：指令格式总表

写数据指令（串口发送数据）	**帧头 0xXXXX**	**长度 0xXX**	**指令码 0x10**	**变量地址 0x0000 ~ 0x5FFF**	**写入数据 0xXXXX……0xXXXX (2*n Bytes)**		**CRC 0xXXXX**
读数据指令	**帧头 0xXXXX**	**长度 0xXX**	**指令码 0x03**	**变量地址 0x0000 ~ 0x5FFF**	**读出的 Word 数量 0xXXXX**		**CRC 0xXXXX**
读数据返馈指令	**帧头 0xXXXX**	**长度 0xXX**	**指令码 0x03**	**变量地址 0x0000 ~ 0x5FFF**	**读出的 Word 数量 0xXXXX**	**数据 (2*n Bytes)**	**CRC 0xXXXX**
触摸反馈指令	**帧头 0xXXXX**	**长度 0xXX**	**指令码 0x41**	**变量地址 (寄存器地址) 0xXXXX**	**返回的 returnValue 0xXXXX**		**CRC 0xXXXX**

指令组成：

1、**帧头：**用于确认数据帧的起始和结尾。上位机默认为 0x5A 和 0xA5，如果需要修改可去工程设置界面修改，更改后确保与串口屏通信的主控端程序内帧头与此帧头一致。

2、**长度：**指令的长度，计算方式为：**长度 = 指令（1） + 变量地址（2） + 写入数据（2*N） + CRC（2）。**

3、**读/写指令码：**用于辨别发送的指令的属性，0x10 表示写入数据，0x03 表示读取数据，0x41 表示触摸反馈，长度 1Byte。

4、**变量地址：**变量地址，即对该地址读或者写；或者是触摸反馈的地址或寄存器，长度 2Bytes。

5、**内容：**需要写入的数据或者是读取数据的长度，长度是 **2*n Bytes。**

6、**CRC：**校验码，用于检验数据，长度 2Bytes。

10.1. 串口写数据指令

串口写数据指令即往具体的地址写数据，使得串口屏做相应的操作，包括换页、调节亮度等。串口写数据指令的目的地址分为两种，一种是变量地址，通常是控件的 writeAddr 或者 parameterAddr，这种地址需要用户自行设置，串口发数据给这些地址可以达到控制控件的目的；另一种是寄存器地址，不同寄存器有不同的用途。各类地址说明请查看存储空间与变量地址说明。

对于串口写数据指令，发送的 word 的数量最大为 0x007D，即单次发送的数据最多为 250Bytes。

主控端与串口屏通信流程如下：

主控端发送写数据指令给串口屏，串口屏接收指令，开始校验指令，CRC 校验成功则串口屏向主控端发送 CRC 校验成功反馈指令并将数据写入指定地址；若 CRC 校验失败则串口屏向主控端发送 CRC 校验失败反馈指令。

串口写数据指令的指令码为 0x10，指令格式如下表：

表 10-2：串口写数据指令及其反馈指令格式表

写数据指令 (串口发送数据)	帧头 0xXXXX	长度 0xXX	指令码 0x10	变量地址 0x0000 ~ 0x5FFF	写入数据 0xXXXX..... 0xXXXX (2*n Bytes)	CRC 0xXXXX
往变量地址 0x2001 写入 0x5152 0x5354 的数据：	0x5AA5	0x09	0x10	0x0001	0x5152 0x5354	0xBB23
CRC 校验成功反馈指令（固定）	0x5AA5	0x04	0x10	0xFF		0x4C30
CRC 校验失败反馈指令（固定）	0x5AA5	0x04	0x10	0x00		0x0C70

使用乐升串口通信软件（UI_Debugger-II）举例：

1、往地址 0x0001 写入 0x1020： **0x10 0x0001 0x1020**

CMD	Addr	Data	CRC	Send
10	0001	1020	B8 1B	

图 10-1：写数据指令举例（1）

2、往地址 0x0003 写入 0x1020、0x2022： **0x10 0x0003 0x1020 0x2022**

CMD	Addr	Data	CRC	Send
10	0003	1020 2022	D2 12	

图 10-2：写数据指令举例（2）

注：1、由于串口通信的底层代码不支持接收数据长度（不是指令长度）为奇数 Bytes 的数据，所以通过串口发送数据时需要确保数据长度为 **2*n Bytes** 。

错误指令：0x10 0x0000 **0x31 0x32 0x33 (数据只有 3 Bytes)**

正确指令：0x10 0x0000 **0x31 0x32 0x33 0x34 (数据有 4 Bytes)**

2、若串口通信调试工具不是配套的 UI_Debugger-II ，则指令需要按照上表发送完整的，包括发送帧头、长度和 CRC。

3、上表两条反馈指令为固定指令，只要写数据指令通过 CRC 校验，就会返回只有数据 **FF** 的指令提示，如果写数据指令没有通过 CRC 校验，就会返回只有数据 **00** 的指令提示。

10.1.1.发送串口指令控制控件举例

10.1.1.1.串口发送控制字符串

参数	数据
控件名称	label_0
参数地址	0xFFFF
变量地址	0x00CF
数据长度(Word)	20
X坐标	110
Y坐标	95
宽度	123
高度	50
字体宽度	27
字体高度	32
字库ID	02_Font-思源...
字库编码	GB2312
对齐方式	Left
背景色使能	Disable
_颜色	0xD3D3D3
字体颜色	0xFFFFFF
初始字符串	文字测试
密码模式	Disable
多国语言	Disable

图 10-3：文字控件参数

如上图所示，字符串控件初值设置为“文字测试”，地址为 0x00CF。我们现在向其发送“乐升”两个中文字符，“乐”字的汉字编码为 0xC0D6，“升”字的汉字编码为 0xC9FD。那么发送的指令应当为 **0x10 0x0720 0xC0D6 0xC9FD 0x0000** ，注意末尾的 **0x0000 为结束符**，在结束符后的内容都不会显示，发送字符串的内容都需要在末尾加结束符 0x0000 或 0x000000，**结束符都要 2Bytes 格式补齐**，

Select	CMD	Addr	Data	CRC	Send
☐	10	00CF	0720 C0D0 C9FD 0000	5D 4B	

图 10-4：写字符串指令举例

发送前后对比如下图。

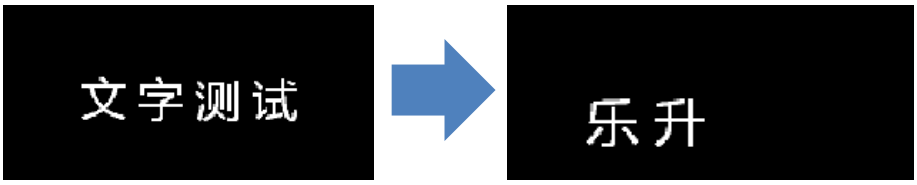


图 10-5：串口发送文字

若需要发送中英混合字符串，如“乐升 1234“，则串口写数据指令为 **0x10 0x0720 0xC0D6 0xC9FD 0x31 0x32 0x33 0x34 0x0000**（ASCII 字符直接转化为 ASCII 编码发送）。

注：发送字符串控件的数据时可在完整的文字编码之间插入 0x0A 作为换行符，换行后需要文本框高度足够才能显示；文本滚动只有一行，不能换行。若发送的文字编码长度等于控件 wordLength，则末尾无需增加结束符。

10.1.1.2. 串口发送控制数字显示

Parameter	Data
name	number_0
parameterAddr	0xFFFF
writeAddr	0x0049
byteLength	4
X	41
Y	58
W	135
H	32
fontWidth	32
fontID	02_Font-微软...
encoding	GB2312
alignment	Left
integerDigit	6
decimalDigit	3
dataType	int
unitSymbol	
_length	0
fontColor	0xFFFFFF
defaultNumber	0

Parameter	Data
name	pngNumber_0
parameterAddr	0xFFFF
writeAddr	0x0049
byteLength	4
X	184
Y	110
W	75
H	20
integerDigit	6
decimalDigit	3
dataType	int
alignment	Left
firstIcon	0059.png
lastIcon	0069.png
defaultNumber	0
leadingZero	Disable

图 10-6：数字控件参数对比

如上图参数设置，两控件地址均为 0x0049，数据类型均为 Int，整数位和小数位均相同且为 6 和 3，初值为 0。我们如果让它们显示数字 6.000，那么发送给其的指令应为 **0x 10 0x0049 0x0000 0x1770**，注意 1770 前面面补 0x0000 是因为数据类型为 int，占了 4 个 Bytes，我们需要把其高位的两个 Bytes 补齐才能正常显示。后面发 0x1770 而不是 0x0006 是因为这里设了 3 位小数，如果要显示数字 6.000，则需要在 6 后面补 3 个 0 即发送 6000 过去，6000 的 16 进制为 0x1770。

Select	CMD	Addr	Data	CRC	Send
☐	10	0049	0000 1770	D9 1B	

图 10-7：写数字指令举例

如果小数位设为 2，需要显示的数据为 6.00，那指令应当为 **0x10 0x0280 0x0000 0x0258**，因为 600 的十六进制为 0x0258。多少位小数，就需要补多少个 0。发送前后对比如下图，图中上面部分为字库数字显示，下面部分为图片数字显示。

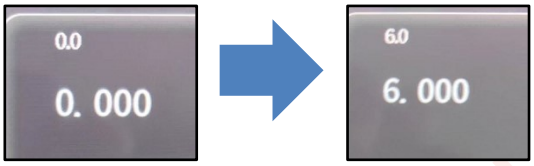


图 10-8：串口发送数字

10.1.1.3.串口发送控制位元状态

Parameter	Data
name	bitIcon_0
parameterAddr	0xFFFF
writeAddr	0x0060
bitIndex	bit0
X	60
Y	64
W	156
H	156
offStateIcon	0079.png
onStateIcon	0056.png
overlap	Disable

图 10-9：位元状态参数设置

位元状态参数如上图所示，地址为 0x0060，触发位为 bit0（第 0 位为 1 就可以），如果想要触发这个位元状态控件，那我们需要发送指令 **0x10　0x0060　0x0001**，效果如下图所示。0 为白环，1 为蓝环。

Select	CMD	Addr	Data	CRC	Send
☐	10	0060	0001	24 1D	

图 10-10：写位元状态指令举例

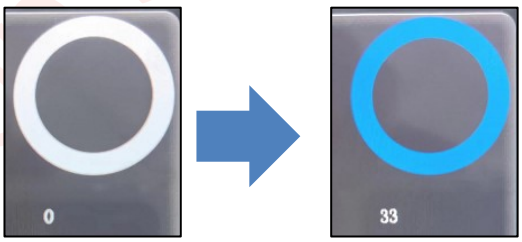


图 10-11：串口控制位元状态显示

10.1.1.4.串口控制切换 Icon

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x0070
byteLength	2
X	243
Y	101
W	12
H	20
firstIcon	0059.png
lastIcon	0069.png
dataFormat	
defaultDisplayID	
minDisplayID	0
maxDisplayID	10
overlap	Disable

图 10-12：Icon 参数设置

Icon 参数如上图所示，地址为 0x0070，受控范围为 0 ~ 10，这 11 张图片为 0 ~9，加上小数点。如果我们想显示数字 8，我们需要向其发送指令 **0x10 0x0500 0x0008** ，效果如下图所示。

Select	CMD	Addr	Data	CRC	Send
☐	10	0070	0008	E5 DE	

图 10-13：写小图标切换指令举例

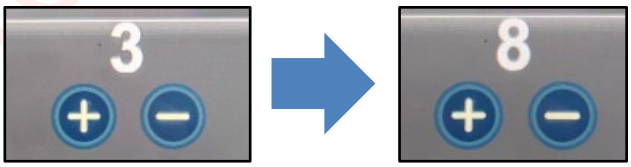


图 10-14：串口发送切换 Icon

10.1.2.发送串口指令控制寄存器举例

原理和串口发送数据到指定地址没有区别，不同的是，寄存器地址范围为 0x7000 ~ 0x71FF，用户可自行分配使用的地址为 0x0000 ~ 0x0200，具体 IC 的用户地址范围可查看不同 IC 的变量地址范围及寄存器地址说明。

10.1.2.1. 串口发送控制页面跳转

0x7000 为换页寄存器，0x0002 为页 ID 的 16 进制数。若跳转目标页不存在工程内，则串口屏无响应。

串口发送跳页到第 2 页：**0x10　0x7000　0x0002**

Select	CMD	Addr	Data	CRC	Send
☐	10	7000	00 02	7E C2	

图 10-15：写页面跳转指令举例

完整指令为：5A A5 07 10 7000　00 02 7E C2

10.1.2.2. 串口发送控制屏幕亮度

0x7001 为背光寄存器，0x002D 为亮度等级，亮度等级范围可设 0~63。

串口发送修改屏幕亮度至 45（十六进制为 2D）：**0x10　0x7001　0x002D**

Select	CMD	Addr	Data	CRC	Send
☐	10	7001	00 2D	6E DE	

图 10-16：写控制屏幕亮度指令举例

完整指令为：5A A5 07 10 7001　00 2D　6E DE

10.1.2.3. 串口发送修改时间

0x7002：修改年，修改范围 00~99。

0x7003：修改月，修改范围 01~12。

0x7004：修改日，修改范围 01~31。

0x7005：修改时，修改范围 00~23。

0x7006：修改分，修改范围 00~59。

0x7007：修改秒，修改范围 00~59。

串口发送修改时间至 2010/10/10/10:10:10：**0x10　0x7002　0x000A　0x000A　0x000A　0x000A　0x000A　0x000A**

Select	CMD	Addr	Data	CRC	Send
☐	10	7002	00 0a 00 0a 00 0a 00 0a 00 0a	EB 3D	

图 10-17：写修改时间指令举例

完整指令为：5A A5 11 10 7002 00 0A 00 0A 00 0A 00 0A 00 0A 00 0A EB 3D

注：串口修改时间必需要从 7002 开始发数据，一次性发送年月日时分秒的内容，不需要往 7008 写入确认，相当于直接更新时间。

10.1.2.4. 串口发送控制音频播放

WAV 控制寄存器地址为 0x700A，写入 0x0000 表示停止播放，写入 0x0001（0x000N）表示播放一次编号为 0000（000N+1）的音频，写入 0x8001（0x800N+1）表示循环播放编号为 0000（0x000N）的音频。

停止播放音频：**0x10　0x700A　0x0000**

Select	CMD	Addr	Data	CRC	Send
☐	10	700A	00 00	DF 01	↗

图 10-18：写控制音频播放指令举例（1）

完整指令为：5A A5 07 10 700A 00 00 DF 01

播放一次编号为 0001 的音频：**0x10　0x700A　0x0002**

Select	CMD	Addr	**Data**	CRC	Send
☐	10	700A	00 02	5E C0	↗

图 10-19：写控制音频播放指令举例（2）

完整指令为：5A A5 07 10 700A 00 00 5E C0

循环播放编号为 0001 的音频：**0x10　0x700A　0x8002**

Select	CMD	Addr	**Data**	CRC	Send
☐	10	700A	80 02	3F 00	↗

图 10-20：写控制音频播放指令举例（3）

完整指令为：5A A5 07 10 700A 80 02 3F 00

10.1.2.5. 串口发送调节音量

音量调节寄存器地址为 0x700B，写入 0~16 可调节音量。

音量调节至等级 1：**0x10　0x700B　0x0001**

Select	CMD	Addr	**Data**	CRC	Send
☐	10	700B	00 01	4F 01	↗

图 10-21：写音量调节指令举例

完整指令为：5A A5 07 10 700B 00 01 4F 01

10.1.2.6. 电阻屏校准指令

电阻屏校准寄存器地址为 0x700C，写入 0x005A 执行电阻屏校准，校准完成后会清零。

Select	CMD	Addr	Data	CRC	Send
☐	10	700C	00 5A	BF 3B	

图 10-22：写电阻屏校准指令

完整指令为：5A A5 07 10 700C 00 5A BF 3B

10.1.2.7. 键码触发寄存器

键码触发寄存器地址为地址 0x700D，详情请查看键码触发说明。

10.1.2.8. 自动背光控制寄存器

自动背光控制寄存器地址 0x700E，写入 0 表示关闭自动背光，写入 1 表示开启。同参数 Auto Dimming。

开启自动背光指令：5A A5 07 10 700E 00 01 5F 00

Select	CMD	Addr	Data	CRC	Send
☐	10	700E	00 01	5F 00	

图 10-23：开启自动背光指令

关闭自动背光指令：5A A5 07 10 700E 00 00 9E C0

Select	CMD	Addr	Data	CRC	Send
☐	10	700E	00 00	9E C0	

图 10-24：关闭自动背光指令

10.1.2.9. 休眠背光亮度控制寄存器

休眠背光亮度控制寄存器地址 0x700F，写入值为休眠时屏幕背光亮度。同参数 Sleep（0~63）。

休眠背光亮度调整为 10 ： 5A A5 07 10 700F 00 0A 4F 07

Select	CMD	Addr	Data	CRC	Send
☐	10	700F	00 0a	4F 07	

图 10-25：写休眠背光亮度指令举例

10.1.2.10. 自动背光休眠时间控制寄存器

自动背光休眠时间控制寄存器地址 0x7010，写入的值为无操作时进入休眠所需的时间，单位秒。同参数 Hold time（s）。

自动背光休眠时间调整为 20 ： 5A A5 07 10 7010 00 14 FE C9

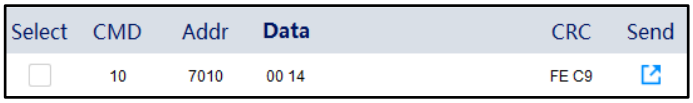


图 10-26：写休眠背光时间指令举例

10.1.2.11. 串口升级寄存器

串口升级寄存器地址 0x7011，写入 0xAA55 进入串口升级模式（需要 **bootloader** 支持）。详细更新过程可查看通过串口更新 LT165 。
进入串口升级模式指令：5A A5 07 10 7011 AA 55 11 99

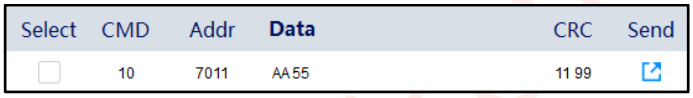


图 10-27：写串口升级指令举例

具体操作可查看使用串口更新 MCU_Code.bin 和 UartTFT-II_Flash.bin。

10.1.2.12. 屏幕显示检测寄存器

屏幕显示检测寄存器地址 0x7041，写入 0 表示退出检测模式，写入 1 表示开启红绿蓝灰阶显示，写入 2 表示开启纯黑显示，写入 3 表示开启纯白显示。
开启红绿蓝灰阶显示指令：

图 10-28：写屏幕检测指令举例（1）

完整指令： 5A A5 07 10 7041 00 01 6E D7
-
开启纯黑显示指令：

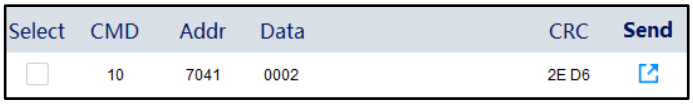


图 10-29：写屏幕检测指令举例（2）

完整指令： 5A A5 07 10 7041 00 02 2E D6

开启纯白显示指令：

Select	CMD	Addr	Data	CRC	Send
☐	10	7041	0003	EF 16	↗

图 10-30：写屏幕检测指令举例（3）

完整指令：5A A5　07　10 7041　00 03　EF 16

退出检测模式指令：

Select	CMD	Addr	Data	CRC	Send
☐	10	7041	0000	AF 17	↗

图 10-31：退出屏幕检测指令

完整指令：5A A5　07　10 7041　00 00　AF 17

10.2. 串口读数据指令

串口读数据指令即主控通过串口发送读指令给串口屏，要求串口屏返回从指定地址开始的指定长度的信息数据。

对于串口读数据指令，读回来的 word 的数量最大为 0x007C，即单次最多读回 248 bytes 数据。

主控端与串口屏通信流程如下：

主控端发送写数据指令给串口屏，串口屏接收指令，开始校验指令，CRC 校验成功则串口屏向主控端发送 CRC 校验成功反馈指令并将读取的数据发送给主控端；若 CRC 校验失败则串口屏向主控端发送 CRC 校验失败反馈指令。

串口读指令和返回指令的指令码都为 0x03，其指令格式如下表：

表 10-3：串口读数据指令与返回指令及其反馈指令

读数据指令	**帧头 0xXXXX**	**长度 0xXX**	**读指令 0x03**	**变量地址 0x0000 ~ 0x5FFF (2 Bytes)**	**读出的 Word 数量 0xXXXX (2 Bytes)**		**CRC 0xXXXX**
从变量地址 0x2050 读出 2*2 个 Bytes 数据：	0x5AA5	0x07	0x03	0x2050	0x0002		0xEA10
读数据返馈指令	**帧头 0xXXXX**	**长度 0xXX**	**读指令 0x03**	**变量地址 0x0000 ~ 0x5FFF (2 Bytes)**	**读出的 Word 数量 0xXXXX (2 Bytes)**	**数据 (2*n Bytes)**	**CRC 0xXXXX**
返回指令读取变量地址 0x2050 的数据：	0x5AA5	0x0B	0x03	0x2050	0x0002	0x3031 0x3233	0x3E67
CRC 校验成功反馈指令（固定）	0x5AA5	0x04	0x03	0xFF			0x4100
CRC 校验失败反馈指令（固定）	0x5AA5	0x04	0x03	0x00			0x0140

读数据指令指令格式：0x03　变量地址　读出长度（word）,如下图所示：

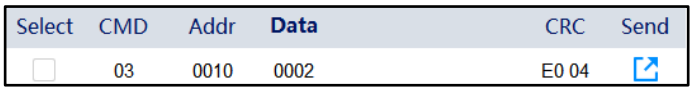


图 10-32：读数据指令举例（1）

串口屏接收到读指令后的返回指令格式：帧头　指令长度　0x03　变量地址　数据长度　数据　CRC

举例：

1、查询地址 0x0020 开始的 4 个 Bytes (两个变量地址) 的内容，即查询 0x0020 和 0x0021 地址的内容，

应发送指令：**0x03 0x0020 0x0002**

Select	CMD	Addr	Data	CRC	Send
☐	03	0020	0002	E0 0B	

图 10-33：读数据指令举例（2）

2、发送指令后，串口屏应答 **0x5A 0xA5 0x04 0x03 0xFF 0x41 0x00 0x5A 0xA5 0x0B 0x03 0x0020 0x0002 0xC0D6 0xC9FD 0x0C 0x2B。**前半绿色部分表示 CRC 通过校验，第二个帧头开始就是整个返回指令，橙色部分为 0x0220 和 0x0221 的存储内容，最后部分为 CRC。

注：

1、若串口通信调试工具不是配套的 UI_Debugger-II，则指令需要按照上表发送完整的，包括发送帧头、长度和 CRC。

2、上表两条反馈指令为固定指令，只要读数据指令通过 CRC 校验，就会返回只有数据 **0xFF** 的指令提示，如果读数据指令没有通过 CRC 校验，就会返回只有数据 **0x00** 的指令提示。

10.3. 触摸反馈指令

触摸反馈指令主要是针对触摸控件的，当触摸控件开启 reportToHost 后（默认是关闭的），触摸控件，串口屏就会返回带有 returnValue 的值的指令给用户，用户可以根据 returnValue 的值的不同判别当前触摸的是哪一个控件，returnValue 的值是用户自行设定的。开启触摸反馈如下图所示：

图 10-34：开启触摸反馈

触摸反馈指令的指令码都是 0x41，只有返回的数据即 returnValue 的值和变量（寄存器）地址有所不同。触摸反馈指令格式表如下所示：

表 10-4：触摸反馈指令格式表

触摸反馈指令	帧头 0xXXXX	长度 0xXX	指令 0x41	变量地址（寄存器地址）0xXXXX	返回的 returnValue 0xXXXX	CRC 0xXXXX
触摸反馈指令	0x5AA5	0x07	0x41	0xFFFF	0x0011	0xD827

触摸反馈控件及其反馈指令如下：

表 10-5：触摸反馈控件表

反馈名称	帧头 (2 Bytes)	长度 (1 Bytes)	标识符 (1 Bytes)	变量/寄存器地址 (2 Bytes)	键值/数据	CRC (2 Bytes)
滑动跳页	0x5AA5	0x07	0x41	0xFFFF	键值 0xXXXX (2 Bytes)	0xXXXX
按键		0x07	0x41	0xFFFF	键值 0xXXXX (2 Bytes)	
弹出窗体		0x07	0x41	0xFFFF	键值 0xXXXX (2 Bytes)	
变量调节/编码器子功能二和三		0x07	0x41	变量/寄存器地址 0xXXXX	数据 0xXXXX (2 Bytes)	
滑条		0x07	0x41		数据 0xXXXX (2 Bytes)	
环形触控		0x07	0x41		数据 0xXXXX (2 Bytes)	
数字键盘		0xXX	0x41		输入的数据 0xXXXX..... 0xXXXX (2*n Bytes)	
多变量操作/编码器子功能一和四		0x23	0x41	(变量/寄存器地址 + 数据)共 8 组 0xXXXX 0xXXXX.....0xXXXX 0xXXXX (4*8 Bytes)		
计时器	0x5AA5	0x23	0x41	(变量/寄存器地址 + 数据)共 8 组 0xXXXX 0xXXXX.....0xXXXX 0xXXXX (4*8 Bytes)		0xXXXX
自动变量	0x5AA5	0x07	0x41	目标变量地址 0xXXXX	停止时数值 0xXXXX (2 Bytes)	0xXXXX

10.4. CRC 计算代码

```
/*****************************CRC 校验*****************************/
//高位字节的 CRC 值
const unsigned char auchCRCHi[] = {
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1,0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80,0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00,0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,0x00, 0xC1, 0x81,
0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1,0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01,0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,0x00, 0xC1, 0x81,
0x40
} ;
//低位字节的 CRC 值
const char auchCRCLo[] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7,0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB,0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE,0xDF, 0x1F, 0xDD,
0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2,0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32,0x36, 0xF6, 0xF7,
```

```
0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E,0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B,0x2A, 0xEA, 0xEE,
0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27,0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1,0x63, 0xA3, 0xA2,
0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD,0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8,0xB9, 0x79, 0xBB,
0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4,0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0,0x50, 0x90, 0x91,
0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94,0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59,0x58, 0x98, 0x88,
0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D,0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83,0x41, 0x81, 0x80,
0x40
};
unsigned short CRC16(unsigned char *puchMsg,unsigned short usDataLen)  /* 函数以 unsigned short 类型返回 CRC */
{
    unsigned char uchCRCHi = 0xFF ;  /* CRC 的高字节初始化 */
    unsigned char uchCRCLo = 0xFF ;  /* CRC 的低字节初始化 */
    unsigned uIndex ;  /* CRC 查询表索引 */
    while (usDataLen--)  /* 完成整个报文缓冲区 */
    {
        uIndex = uchCRCLo ^ *puchMsg++ ; /* 计算 CRC */
        uchCRCLo = uchCRCHi ^ auchCRCHi[uIndex] ;
        uchCRCHi = auchCRCLo[uIndex] ;
    }
    return (uchCRCHi << 8 | uchCRCLo);
}
```

源码出处：UI_Editor-Lite 的 Mcu_Code 代码里搜索函数名 CRC16 。

10.4.1.读写数据指令的 CRC 计算

对于读写数据指令，其组成部分为：**帧头(2)+长度(1)+读/写指令(1)+变量地址(2)+数据(X)+CRC(2)**，如下表。

表 10-6：读写数据指令的 CRC 计算

指令类型	帧头	长度	指令码	变量地址	数据	CRC
写数据指令	帧头 0xXXXX	长度 0xXX	写指令 0x10	变量地址 0~~0x5FFF (2 Bytes)	写入数据 0xXXXX……0xXXXX (2*n Bytes)	CRC 0xXXXX
读数据指令	帧头 0x5AA5	长度 0xXX	读指令 0x03	变量地址 0~~0x5FFF (2 Bytes)	读出的 Word 数量 0xXXXX (2 Bytes)	CRC 0xXXXX

上表绿色部分为参与 CRC 计算的部分，

对于写变量指令（0x10）：其参与 CRC 计算的部分为 **写指令 变量地址 写入数据**

对于读变量指令（0x03）：其参与 CRC 计算的部分为 **读指令 变量地址 读出数据的长度（Word）**

10.4.2.读返回指令的 CRC 计算

表 10-7：读返回指令的 CRC 计算

指令类型	帧头	长度	指令码	变量地址	数据数量	数据	CRC
读数据反馈指令	帧头 0xXXXX	长度 0xXX	读指令 0x03	变量地址 0~~0x5FFF (2 Bytes)	读出的 Word 数量 n (2 Bytes)	读出的数据 0xXXXX (2*n Bytes)	CRC 0xXXXX

上表绿色部分为参与 CRC 计算的部分，

对于读指令的反馈内容：其参与 CRC 计算的部分为 **读指令 变量地址 读出数据的长度（Word） 读出的数据**

10.4.3.触摸反馈指令的 CRC 计算

控件串口返回指令有以下几种类型，具体是哪种控件的反馈指令可参考上位机工程以及触摸反馈指令。

表 10-8：控件串口返回指令的 CRC 计算

帧头 0x5AA5	长度 0x07	标识符 0x41	地址 0xFFFF	键值 0xXXXX	NULL	CRC 0xXXXX
帧头 0x5AA5	长度 0x07	标识符 0x41	变量/寄存器地址 0xXXXX (2 Bytes)	数据 0xXXXX (2 Bytes)	NULL	CRC 0xXXXX
帧头 0x5AA5	长度 0xXX	标识符 0x41	变量/寄存器地址 0xXXXX (2 Bytes)	数据 0xXXXX……0xXXXX (2*n Bytes)		CRC 0xXXXX
帧头 0x5AA5	长度 0xXX	标识符 0x41	(变量/寄存器地址 + 数据)共 8 组 0xXXXX 0xXXXX……0xXXXX 0xXXXX (4*8 Bytes)			CRC 0xXXXX

上表绿色部分为参与 CRC 计算的部分。控件串口返回指令的标识符为 0x41，键值的返回地址固定为 0xFFFF 。

10.5. 串口发送直接修改控件参数

165 系列芯片不支持通过参数地址修改控件显示状态。

10.6. 键码触发说明

控件是否支持键码触发可查看下表：

表 10-9：控件是否支持键码触发

控件名称	是否支持键码触发	控件名称	是否支持键码触发
按键	Y	数字时钟	N
弹出窗体	Y	动图	N
变量调节	Y	二维码	N
多变量操作	Y	音频播放	N
环形触摸	N	滑条	N
数字键盘	Y	位元状态	N
键盘按键	N	编码器	N
字符串	N	计时器	N
小图标	N	静态文本	N
字库数字	N	自动变量	N
图片数字	N	指针	N

注：Y 代表支持，N 代表不支持。

键码触发指通过串口发送的方式模拟触控触发控件，例如一个按键（button）,pageGoto 设置了 Page0001，hostControl 设置了 Enable，然后将_triggerValue 设置为 0x0001，则我们使用串口发送 0x0001 至键码触发寄存器 0x700D，串口屏就会执行该按键的预设操作，跳转页面至 Page0001。

使用键码触发功能时需要注意以下事项：

1. 键码触发开启后，该控件就不能使用触控操作。如果还需要触控操作可以复制一个相同的控件并关闭键码触发功能，放在该位置。
2. 同一页面内所有开启键码触发功能的控件的_triggerValue 应当互不相同。
3. 键码触发功能开启后，该控件不会再显示，不管该控件是否添加了图片。
4. 键码触发值（_triggerValue）可设范围为 0x0001~0xFFFF。

11. ModBus 通信功能

ModBus 通信功能允许客户使用应用更为广泛的 ModBus 通讯协议来替代乐升的通讯协议。当客户选择使用 ModBus 通信功能时，主设备向从设备发送请求，从设备接收并处理主设备的请求，向主设备发送结果。

乐升 Modebus 协议为标准协议，仅支持 RTU 模式，使用 Modbus 协议后不再兼容乐升通信协议。乐升 Modebus 协议作主机时，支持寄存器操作和线圈操作；作从机时，仅支持寄存器操作，不支持线圈操作。

11.1. 创建 ModBus 主机指令文件

注：该章节仅串口屏作主机时需要查看。

ModBus 主机指令文件的全称是 **command.list**，是一个后缀为 **.list** 的文件，可以用记事本打开并编辑。ModBus 指令文件的存放与素材文件夹同级。创建该文件的方法有两种，一种是创建一个空白的 TXT 文档，然后将其名字重命名为 **command.list**，另一种方法是编辑 Modbus 指令后直接将指令保存为 **command.list**。

图 11-1：ModBus 指令文件

注：该工程作主机时，**command.list** 必须有且只能有一个。

11.2. ModBus 主机指令编辑界面

注：该章节仅串口屏作主机时需要查看。

ModBus 主机指令编辑界面的入口如下图，单击后进入指令编辑界面。

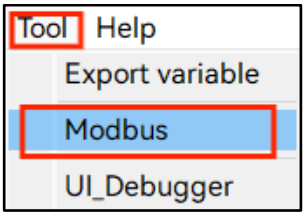


图 11-2：ModBus 指令编辑界面入口

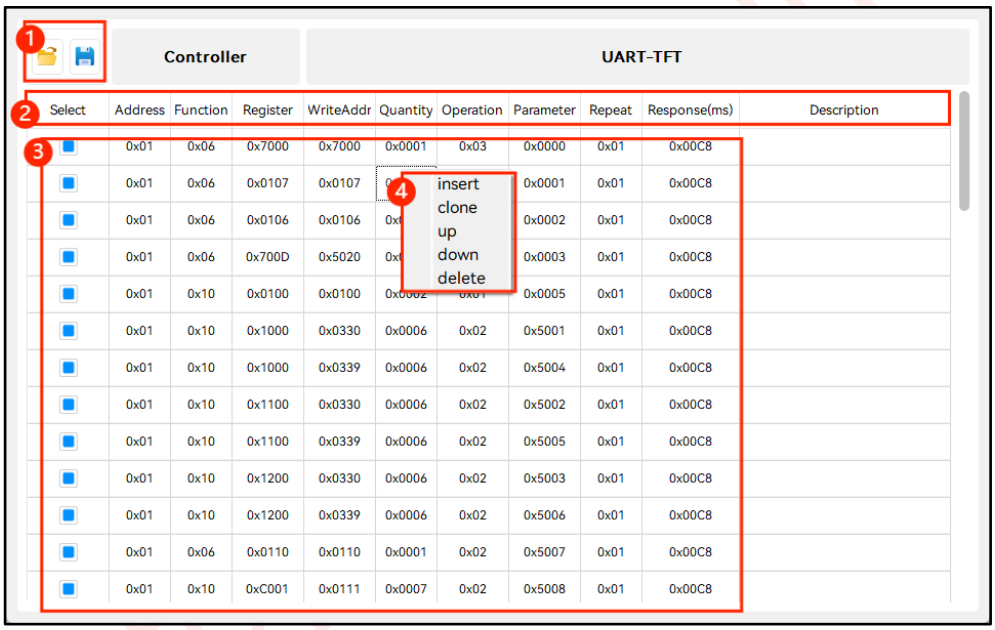


图 11-3：ModBus 指令编辑界面

1、  导入 ModBus 主机指令文件，指令文件名字为 **command.list**。  保存指令文件。保存指令时需注意，只有完整的指令才会被保存至 **command.list** 文件内。

2、指令组成说明：各参数说明如下：

Select：勾选上的指令才会在编译工程时被打包，需确保勾选的指令为完整指令。

Address：从机地址，**可设范围 0x01~0xFF，不能设置为 0x00。**

Function：指令功能码。

Register：从机的寄存器地址，读/写操作的起始地址；线圈的地址。

WriteAddr：主机的变量地址，读/写操作的起始地址。

Quantity：线圈的个数/寄存器的个数，寄存器个数为 1 时，数据长度为 2Bytes。

Operation：指令执行模式，总共有 4 种模式。

Parameter：指令执行模式对应参数，与 Mode 对应设置。

Repeat：重发次数，主机先发送指令一次给从机，若该从机超出响应时间后依旧没有应答，则主机再次发送该指令，若从机连续 **Repeat + 1** 个周期无响应，主机跳过该指令继续发送下一条指令。

Response（ms）：响应时间，单位 ms。

Description：中文注释，添加该指令的说明以增加可读性。

3、指令编辑区：

4、对选中指令进行操作（选中指令点击鼠标右键唤出此窗口）

Insert：在选中指令的上方添加空白行。

Clone：复制选中指令。

Up：将选中的指令往上移动一位，即与上方相邻的指令调换位置。

Down：将选中的指令往下移动一位，即与下方相邻的指令调换位置。

Delete：删除选中的指令。

注：如果需要使用 modbus 通信，则编辑好指令后需要点击  保存指令文件一次，再去编译生成 UartTFT-II_Flash.bin 文件。

11.3. ModBus 主机指令结构

注：该章节仅串口屏作主机时需要查看。

表 11-1：ModBus 指令结构

	从机地址	读写功能	主从机数据传输参数			指令发送的判断条件		指令响应参数	
名称	Address	指令功能码 Function	从机寄存器地址 Register	主机变量地址 WriteAddr	数据长度 Quantity	指令执行模式 Operation	指令执行模式对应参数 Parameter	重发次数 Repeat	响应时间 Response
长度/Bytes	1	1	2	2	2	1	2	1	2

从机地址：从设备地址，**不能设置为 0x00**。

指令功能码：如下表所示。

表 11-2：指令功能码

指令功能码	对应功能	寄存器或线圈个数
0x03	读保持寄存器	1~125
0x04	读输入寄存器	1~125
0x06	写单个保持寄存器	1
0x10	写多个保持寄存器	1~123
0x01	读线圈	1~2000
0x02	读离散输入	1~2000
0x05	写单个线圈	1
0x0F	写多个线圈	1~1968

从机寄存器地址：读/写操作的起始地址；线圈的地址。

主机变量地址：主机读/写操作的起始地址。

数据长度：线圈的个数/寄存器的个数。

指令执行模式：指令执行模式和指令执行模式对应参数这两部分构成了指令发送的判断条件，总共有 4 种模式，指令模式码及其对应功能如下。使用例程可参考 Modbus 指令执行模式具体使用教程。

表 11-3：指令执行模式参数对应表

指令执行模式	指令执行模式对应参数
0x00	固定 0x0000
0x01	页码
0x02	主机变量地址 （该变量为 0x4C54 时触发）
0x03	指定编号 （用户自行编写代码）

0x00：在所有页面下都执行此指令（即无限制指令），**对应的 Parameter 设置成 0x0000 即可。**

0x01:仅在指定页面下执行此指令,对应的 Parameter 设置成页码,例如 Page0003 对应设置成 0x0003。

0x02：仅在变量地址的数据为 **0x4C54** 时才执行此命令，对应的 Parameter 设置成变量地址。

0x03：客制化模式的对应编号的标志位置 1 才会执行该指令，对应的 Parameter 设置成标志位编号。每一个编号（编号唯一）在程序里都代表了一个固定的操作，主机触发了此操作，则对应编号的标志位置 1，主机检测到此标志位置 1 后，随即发送对应的指令给从机。

重发次数：主机先发送指令一次指令给从机，若该从机超出响应时间后依旧没有应答，则主机再次发送该指令，若从机连续 **Repeat + 1** 个周期无响应，则主机跳过该指令，主机继续发送下一道指令。

响应时间：主机发送指令后等待应答的时间，单位 ms。

11.4. ModBus 主机指令说明

ModBus 主机指令的作用：用于告知主机在什么情况下执行发送实际指令的动作。

Modbus 通信正常时，主机发送一条指令，对应从机应答一条指令。在编辑指令时，并不需要输入发送的数据内容，只需要确定主从机的变量地址以及数据长度，数据传输的内容都是主从机根据指令在各自的变量空间里自行获取的。UI_Editor-Lite 的 modbus 协议会循环查询列表内的指令，根据每一条指令的指令执行模式查询其是否符合发送条件，符合则发送，不符合则跳过。

注：主机发送多条需要连发的指令时，需要在指令与指令之间加上时间间隔，不能将两条或多条指令当成一个数据帧发送。

11.4.1.主机读保持寄存器（指令码 0x03）

表 11-4：主机读保存寄存器指令

从机地址	功能指令	从机寄存器地址	主机变量地址	数据长度	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0x01	0x03	0x0000	0x0020	0x0009	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x03 表示这是一条主机读从机保持寄存器的指令。主机发送指令到地址为 0x01 的从机后，该从机将其寄存器 0x0000 ~ 0x0008 的数据发送给主机，主机将这部分的数据存放在 0x0020 ~ 0x0028 这部分变量空间里。

对应实际通信指令示例如下：

表 11-5：主机读保存寄存器实际通信指令

读保持寄存器主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	寄存器地址 (2 Bytes)	读回的寄存器数量 Word (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x03	0x0000	0x0009	0x85 0xcc
读保持寄存器从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	返回的数据长度 Bytes (1 Byte)	数据内容 (2*n Bytes)	CRC16 (2 Bytes)
	0x01	0x03	0x12	0x0001 0x0002 0x0003 0x0004 0x0005 0x0006 0x0007 0x0008 0x0009	0x9c 0xb4

11.4.2.主机读输入寄存器（指令码 0x04）

表 11-6：主机读输入寄存器指令

从机地址	功能指令	从机寄存器地址	主机变量地址	数据长度	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0x01	0x04	0x0000	0x0020	0x0009	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x04 表示这是一条主机读从机输入寄存器的指令。原理与保持寄存器相同。

对应实际通信指令示例如下：

表 11-7：主机读输入寄存器实际通信指令

读保持寄存器主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	寄存器地址 (2 Bytes)	读回的寄存器数量 Word (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x04	0x0000	0x0009	0x30 0x0c
读保持寄存器从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	返回的数据长度 Bytes (1 Byte)	数据内容 (2*n Bytes)	CRC16 (2 Bytes)
	0x01	0x04	0x12	0x0001 0x0002 0x0003 0x0004 0x0005 0x0006 0x0007 0x0008 0x0009	0x29 0xb3

11.4.3.主机写单个保持寄存器（指令码 0x06）

表 11-8：主机写单个保存寄存器指令

从机地址	功能指令	从机寄存器地址	主机变量地址	数据长度	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0x01	0x06	0x0000	0x0020	0x0001	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x06 表示这是一条主机写单变量给从机的指令。主机发送数据到地址为 0x01 的从机，将主机 0x0020 这个地址共 2Bytes 的数据发送到从机 0x0000 这个寄存器中。

对应实际通信指令示例如下：

表 11-9：主机写单个保存寄存器实际通信指令

写单个保持寄存器主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	寄存器地址 (2 Bytes)	写入的数据 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x06	0x0000	0x0000	0x89 0xca
写单个保持寄存器从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	寄存器地址 (2 Bytes)	写入的数据 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x06	0x0000	0x0000	0x89 0xca

11.4.4.主机写多个保持寄存器（指令码 0x10）

表 11-10：主机写多个保持寄存器指令

从机地址	功能指令	从机寄存器地址	主机变量地址	数据长度	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0x01	0x10	0x0000	0x0000	0x0009	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x10 表示这是一条主机写多个保持寄存器给从机的指令。主机从变量空间 0x0000~0x0008 取 18 个 Byte 数据发送至从机保持寄存器 0x0000 ~ 0x0008。

对应实际通信指令示例如下：

表 11-11：主机写多个保持寄存器实际通信指令

写多个保持寄存器主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	寄存器地址 (2 Bytes)	写入的寄存器数量 Word (2 Bytes)	数据长度 Bytes (1 Byte)	写入的数据 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x10	0x0000	0x0009	0x12	0x0001 0x0002 0x0004 0x0008 0x0010 0x0020 0x0040 0x0080 0x0000	0x95 0x3c
写多个保持寄存器从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	寄存器地址 (2 Bytes)	写入的寄存器数量 Word (2 Bytes)	NULL		CRC16 (2 Bytes)
	0x01	0x10	0x0000	0x0009	NULL		0x00 0x0f

11.4.5.主机读线圈状态（指令码 0x01）

表 11-12：主机读线圈状态指令

从机地址	功能指令	从机线圈地址	主机变量地址	数据长度/线圈个数	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0x01	0x01	0x0009	0x0001	0x000A	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x01 表示这是一条主机读从机线圈状态的指令。主机的一个变量含有 bit0 ~ bit15 共 16 个位，从机线圈地址 % 0x10 = N （求余），主机会将接收到的从机线圈状态在变量 0x0001 的 bitN 开始保存，这里是 bit9 。变量 0x0001 的 Bit9 对应存放线圈 0x0009 的状态，变量 0x0001 的 Bit15 对应存放线圈 0x000F 的状态，变量 0x0001 只存放了 7 个线圈状态，剩下的 0x0010 ~ 0x0012 线圈状态对应存放在变量 0x0002 的 bit0 ~ bit2 。

对应实际通信指令示例如下：

表 11-13：主机读线圈状态实际通信指令

读线圈状态主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	线圈地址 (2 Bytes)	读回的线圈数量 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x01	0x0009	0x000a	0x6c 0x0f
读线圈状态从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	返回的数据长度 Bytes (1 Byte)	数据 (2*n Bytes)	CRC16 (2 Bytes)
	0x01	0x01	0x02	0xde 0x03	0xa0 0x5d

从机回复 0xde 0x03 的原因：

从机线圈 0x0009~0x0012 的状态为 0111 1011 11，将 0x0009~0x0010 的状态倒序得 1101 1110 ，倒序是为了对应主机 16 位变量高位在左，换成 16 进制为 0xde ；线圈 0x0011~ 0x0012 状态为 11，不满 8 位，在 11 之后补 0 补满 8 位，得到 1100 0000 ，再进行倒序得 0000 0011，即为 0x03。

11.4.6.主机读离散输入（指令码 0x02）

表 11-14：主机读离散输入指令

从机地址	功能指令	从机线圈地址	主机变量地址	数据长度/线圈个数	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0X01	0X02	0x0009	0x0001	0x000A	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x02 表示这是一条主机读从机离散输入的指令。原理与读线圈状态相同，不同的是离散输入只能读不能写。

对应实际通信指令示例如下：

表 11-15：主机读离散输入实际通信指令

读离散输入 主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	线圈地址 (2 Bytes)	读回的线圈数量 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x02	0x0009	0x000a	0x28 0x0f
读离散输入 从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	返回的数据长度 Bytes (1 Byte)	数据 (2*n Bytes)	CRC16 (2 Bytes)
	0x01	0x02	0x02	0xde 0x03	0xa0 0x19

11.4.7.主机写单个线圈（指令码 0x05）

表 11-16：主机写单个线圈指令

从机地址	功能指令	从机线圈地址	主机变量地址	数据长度/线圈个数	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0X01	0X05	0x0013	0x0001	0x0001	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x05 表示这是一条主机写从机单个线圈的指令。主机一个变量含有 bit0 ~ bit15 共 16 个位，从机线圈地址 % 0x10 = N （求余），主机会将变量 0x0001 的 bitN 的状态发送至对应从机线圈，若 bitN 为 0 （OFF 状态），则数据发送 0x0000；若 bitN 为 1 （ON 状态），则数据发送 0xFF00；其它所有值均为非法的，并且对线圈不起作用。

对应实际通信指令示例如下，主机变量 0x0001 的 bit3 状态为 1。

表 11-17：主机写单个线圈实际通信指令

写单个线圈主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	线圈地址 (2 Bytes)	线圈状态 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x05	0x0013	0xff00	0x7d 0xff
写单个线圈从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	线圈地址 (2 Bytes)	线圈状态 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x05	0x0013	0xff00	0x7d 0xff

11.4.8.主机写多个线圈（指令码 0x0F）

表 11-18：主机写多个线圈指令

从机地址	功能指令	从机线圈地址	主机变量地址	数据长度/线圈个数	指令执行模式	指令执行模式对应参数	重发次数	响应时间
0X01	0X0F	0x0009	0x0001	0x000F	0x00	0x0000	0x05	0x00C8

指令说明：功能指令 0x0F 表示这是一条主机写从机多个线圈的指令。主机的一个变量含有 bit0 ~ bit15 共 16 个位，从机线圈地址 % 0x10 = N （求余），主机会从变量 0x0001 的 bitN 开始发送，这里是 0x0001 的 bit9，连续发送 0x000F（15）个 bit，即变量 0x0001 的 bit9 ~ bit15 和变量 0x0002 的 bit0 ~ bit7。从机线圈 0x0009 对应变量 0x0001 的 bit9；从机线圈 0x0017 对应变量 0x0002 的 bit7。

对应实际通信指令示例如下，主机变量 0x0001 的内容为 0x5400，变量 0x0002 的内容为 0x0005。

表 11-19：主机写多个线圈实际通信指令

写多个线圈主机发送	从机地址 (1 Byte)	功能指令 (1 Byte)	线圈地址 (2 Bytes)	写入的线圈数量 数量 Word (2 Bytes)	数据长度 Bytes (1 Byte)	写入的数据 (2 Bytes)	CRC16 (2 Bytes)
	0x01	0x0f	0x0009	0x000f	0x02	0xaa 0x02	0x1a 0x0c
写多个线圈从机回复	从机地址 (1 Byte)	功能指令 (1 Byte)	线圈地址 (2 Bytes)	写入的线圈数量 (2 Bytes)	NULL		CRC16 (2 Bytes)
	0x01	0x0f	0x0009	0x000f	NULL		0xc5 0xcd

主机发送 0xaa 0x02 的原因：

主机变量 0x0001 的内容为 0x5400,主机变量 0x0002 的内容为 0x0005，两者按照地址顺序关系组合成 32 位 0x0005 5400，转换为 2 进制 0000 0000 0000 0101 0101 0100 0000 0000（最右边为 bit0）

根据上述**指令说明** 的描述，以上黄色背景状态位会被写到从机的线圈状态，即 bit23~bit9。bit16~bit9 为一个字节 1010 1010 即 0xaa；bit23~bit17 不够一个字节，往高位补零补齐八位得 0000 0010，即为 0x02。

11.5. Modbus 通信指令的 CRC 计算

Modbus 通信指令整条指令除 CRC 外均参与 CRC 的计算，计算代码可参考 CRC 计算代码。

11.6. Modbus 工程及程序设置例程

11.6.1.Modbus 从机工程设置

若工程需要作 modbus 从机时，需要对 UI 工程和 Mcu_code 进行以下操作。

UI 工程设置：

如下图，打开工程设置确认从机的设备地址（**从设备地址不能设为 0x00**），只有指令的设备地址（从机地址）与工程设置的设备地址对应，该指令才会被对应从机接收。

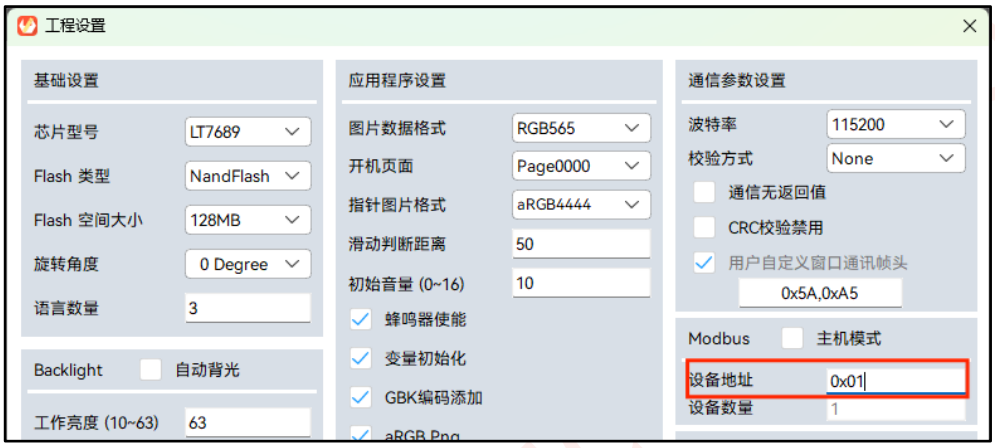


图 11-4：设置从机地址

Mcu_code 配置：

若使用 LT165x 的串口屏，需要在下图代码位置更改宏定义

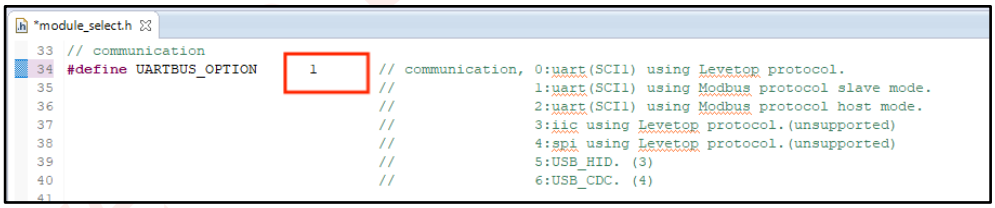


图 11-5：LT165 选择 modbus 从机模式

11.6.2.Modbus 主机工程设置

若工程需要作 modbus 主机时，需要对 UI 工程和 Mcu_code 进行以下操作。

UI 工程设置：

如下图，打开工程设置勾选 Modebus 主机模式，此时设备地址无需设置。

图 11-6：设置主机模式

Mcu_code 配置：

若使用 LT165x 的串口屏，需要在下图代码位置更改宏定义

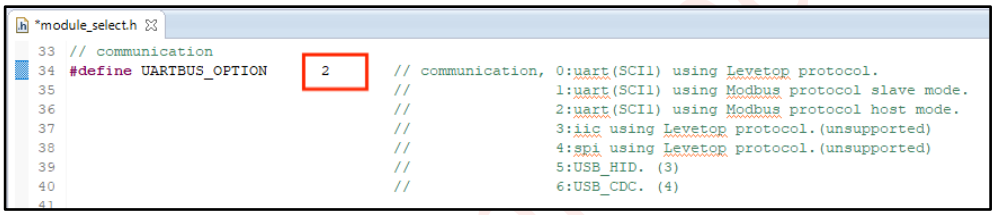


图 11-7：LT165 选择 modbus 从机模式

11.7. Modbus 指令执行模式具体使用教程

11.7.1.Modbus 指令执行模式 Operation 0x00

该模式属于无条件执行的模式，不需要额外设置。

11.7.2.Modbus 指令执行模式 Operation 0x01

Select	Address	Function	Register	WriteAddr	Quantity	Operation	Parameter	Repeat	Response(ms)
■	0x01	0x03	0x0000	0x0020	0x0009	0x01	0x0002	0x05	0x00c8

图 11-8：Operation0x01 模式通信指令

如上图指令，指令执行模式 Operation 为 0x01，表明该指令通过页判断执行，其对应 Parameter 为 0x0002，表明串口屏显示如下图示例 page0002 页面时，该指令执行。

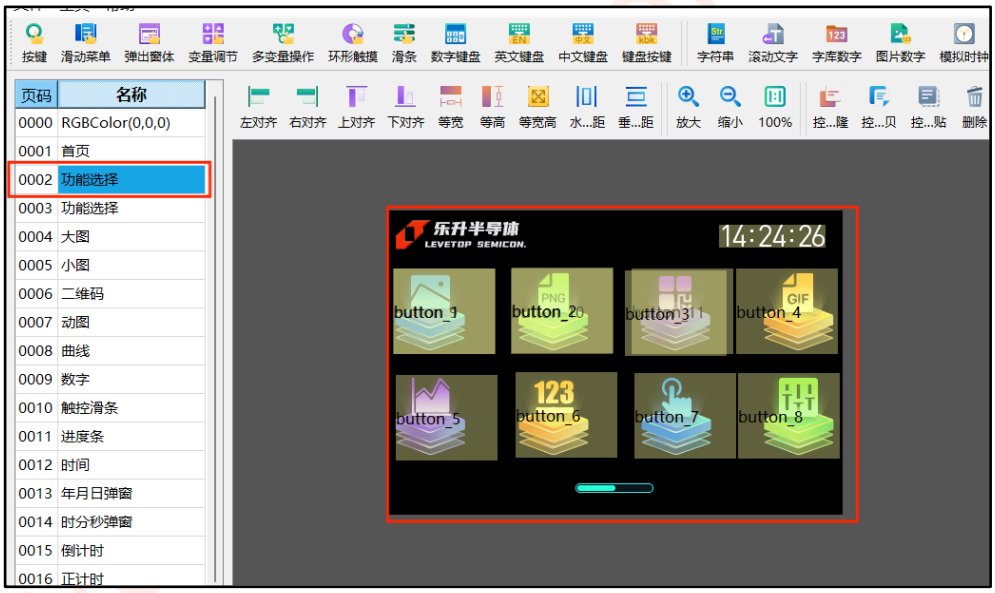


图 11-9：示例图页面

11.7.3.Modbus 指令执行模式 Operation 0x02

Select	Address	Function	Register	WriteAddr	Quantity	Operation	Parameter	Repeat	Response(ms)
■	0x01	0x03	0x0000	0x0020	0x0009	0x02	0x0010	0x05	0x00c8

图 11-10：Operation0x02 模式通信指令

如上图指令，指令执行模式 Operation 为 0x02，表明该指令通过主机变量地址判断执行，其对应 Parameter 为 0x0010，表明串口屏工程运行时变量地址 0x0010 的内容为 0x4C54，该指令执行。执行完指令后，程序会自动将变量地址 0x0010 的内容清零。

如下图，一般在 UI 工程内使用多变量操作控件对地址 0x0010 赋值 0x4C54,

参数	数据
控件名称	batVar_0
X坐标	417
Y坐标	215
宽度	32
高度	52
未按图标	
按下图标	
跳转页面	
操作变量地址0	0x0010
_赋值	0x4C54
操作变量地址1	0xFFFF
_赋值	0xFFFF
操作变量地址2	0xFFFF
_赋值	0xFFFF

图 11-11：多变量操作控件对地址赋值

11.7.4.Modbus 指令执行模式 Operation 0x03

Select	Address	Function	Register	WriteAddr	Quantity	Operation	Parameter	Repeat	Response(ms)
■	0x01	0x10	0x1500	0x1500	0x0002	0x03	0x0001	0x01	0x00C8

图 11-12：Operation0x03 模式通信指令

如上图指令，指令执行模式 Operation 为 0x03，表明该指令需通过程序内对指定标志位置 1 判断执行，其对应 Parameter 为 0x0001。

下图为例程

1、找到 modbus 指令发送函数入口 Uart_cmd_Send();

```
module_select.h | main.c | bsp.c | bsp.h | uart.c
255 #elif (UARTBUS_OPTION == 2)
256         if (Sum_ModbusTX)
257             Uart_cmd_Send();
258
259         LT_ModBus_REG_Cmd();
```

图 11-13：指令发送函数所在位置

2、获取 Modbus 指令执行模式 0x03 的标志位数组 Master_mode03_flag[]，数组 0x0001 号元素就是上述指令的标志位。

```
693
694 volatile uint8_t Master_mode03_flag[100] = {0}; // Customized variables
695 volatile uint8_t Master_mode03_Var[200] = {0};  // Customized variables
696
697 // The transmission mechanism of host timing and repeated serial port data
698 void Uart_cmd_Send(void)
699 {
700     uint8_t i = 0,j = 0;
701     uint16_t num=0, data_temp=0;
702     uint8_t byte_temp = 0;
703     uint16_t sum=0, count=0, cnt=0;
```

图 11-14：标志位数组

3、前往程序对应位置将标志位置 1

这里选择了 button 按钮点击触发，所以前往 button 的触控判断函数。

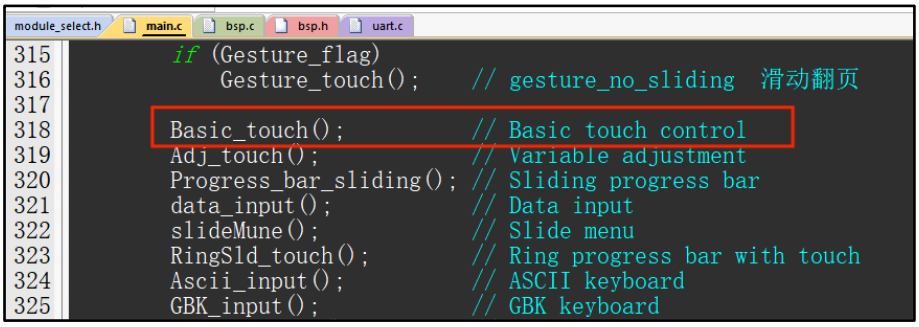


图 11-15：触控判断函数所在位置

Basic_touch(); 函数部分截图如下

if (gTpInfo.sta == 0 && Basci_flag == 1)是点击 button 按键时手指抬起的标志，在该判断内将 Modbus 指令执行模式 0x03 的标志位数组 Master_mode03_flag[] 的对应元素置 1。

Parameter 为 0x0001 ，则数组元素 Master_mode03_flag[**0x0001**]置 1。

图 11-16：对应元素置 1

UI 工程内触发 button 控件参数如下，程序内 gBasci_Info[Basci_num].Keyvalue 即为串口返回值，添加该判断是为了识别对应 button，若不加以判断则点击任何 button 都会将标志位置 1。指令执行完后程序会自动将 Master_mode03_flag[**0x0001**]置 0。

参数	数据
控件名称	button_2
X坐标	55
Y坐标	196
宽度	74
高度	77
串口返回值	0x0022
未按图标	
按下图标	
跳转页面	
串口返回使能	Disable
键码触发使能	Disable
_键码触发值	0x0000

图 11-17：添加串口返回值

12. 读写外部 SPI Flash 和 MCU Flash 说明

12.1. 指令说明

表 12-1：读写 Flash 指令表

指令	帧头 0x5A 0xA5 (2 Bytes)	固定 0x00 (1 Byte)	长度 (2 Byte)	指令码 (1 Byte)	操作地址 (4 Bytes)	内容 (n Bytes, 最大=2048)	CRC 0xXX 0xXX (2 Bytes)
写 MCU EFlash	0x5A 0xA5	0x00	数据的长度 +0x0007	0x61	0x00001000 ~ 0x00034000	数据	CRC16
响应	0x5A 0xA5	0x00	0x0009	0x61		OK/NG	CRC16
读 MCU EFlash	0x5A 0xA5	0x00	0x0009	0x62		读取数据长度 (2Bytes)	CRC16
响应	0x5A 0xA5	0x00	数据的长度 +0x0007	0x62		数据	CRC16
校验 MCU Flash	0x5A 0xA5	0x00	0x000B	0x63		校验数据长度 (4Bytes)	CRC16
响应	0x5A 0xA5	0x00	0x000B	0x63		校验码 CRC32 (4Bytes)	CRC16
写 SPI Flash	0x5A 0xA5	0x00	数据的长度 +0x0007	0x64	Addr	数据	CRC16
响应	0x5A 0xA5	0x00	0x0009	0x64	Addr	OK/NG	CRC16
读 SPI Flash	0x5A 0xA5	0x00	0x0009	0x65	Addr	读取数据长度 (2Bytes)	CRC16
响应	0x5A 0xA5	0x00	数据的长度 +0x0007	0x65	Addr	数据	CRC16
校验 SPI Flash	0x5A 0xA5	0x00	0x000B	0x66	Addr	校验数据长度 (4Bytes)	CRC16
响应	0x5A 0xA5	0x00	0x000B	0x66	Addr	校验码 CRC32 (4Bytes)	CRC16

注：OK = 0x4F 0x4B ;NG = 0x4E 0x47 。

必读：共 3 种指令，写数据指令、读数据指令和校验数据指令，由于操作对象有两个，一共有 6 条指令。

对于读或写 MCU EFlash 的数据，应当遵守以下规则：

1、操作地址限制为 0x00001000 ~ 0x00034000，0x0000 ~ 0x1000 这部分 Flash 空间内存放了串口屏参数，不可更改。

2、**写操作**时目标地址应当为 4 的倍数，否则无法写入；读操作地址不作限制。

3、单次读取或写入的最大数据量为 2K bytes 。

对于读或写外部 SPI Flash 的数据，应当遵守以下规则：

1、操作地址应当在 UI 工程的 UartTFT-II_Flash.bin 存放范围之外，小于 SPI Flash 芯片的额定容量。

2、**对 Nand Flash 进行写操作时**，目标地址应当为 128K 的倍数，否则无法写入，进行写操作前，程序会

将从当前地址开始往后 128K 的空间进行擦除，该 128K 空间内容全为为 0xFF。

对 Nor Flash 进行写操作时，目标地址应当为 4K 的倍数，否则无法写入，进行写操作前，进行写操作前，程序会将从当前地址开始往后 4K 的空间进行擦除，该 4K 空间内容全为为 0xFF。

读操作的地址不作限制。

3、单次读取或写入的最大数据量为 2K bytes ，写入较长（超过 2K）的数据时，软件会自动将数据从头到尾拆成 2K 的包发送，最后一包一般不足 2K。

注：写入数据较多（几百 K）时，可能会出现丢包现象。

12.2. 通信过程说明

主控端向串口屏发送上述 6 条指令中的一条时，串口屏接收到指令，进行校验，校验后返回校验通过或者不通过的反馈指令，然后进行下一步，执行写入数据并返回写入数据结果的操作、读取数据并返回数据内容的操作或者校验数据并返回 32 位校验码的操作。

6 条指令的 CRC 校验反馈指令如下表：

表 12-2：CRC 校验反馈表

指令	帧头 (2Bytes)	长度 (1Byte)	指令码 (1Byte)	内容 (1Byte)	CRC16 (2Bytes)
写 MCU EFlash 指令校验通过反馈指令	0x5A 0xA5	0x04	0x61	0xFF	0x68 0x60
写 MCU EFlash 指令校验失败反馈指令	0x5A 0xA5	0x04	0x61	0x00	0x28 0x20
读 MCU EFlash 指令校验通过反馈指令	0x5A 0xA5	0x04	0x62	0xFF	0x68 0x90
读 MCU EFlash 指令校验失败反馈指令	0x5A 0xA5	0x04	0x62	0x00	0x28 0xD0
校验 MCU EFlash 指令校验通过反馈指令	0x5A 0xA5	0x04	0x63	0xFF	0x69 0x00
校验 MCU EFlash 指令校验失败反馈指令	0x5A 0xA5	0x04	0x63	0x00	0x29 0x40
写 SPI Flash 指令校验通过反馈指令	0x5A 0xA5	0x04	0x64	0xFF	0x6B 0x30
写 SPI Flash 指令校验失败反馈指令	0x5A 0xA5	0x04	0x64	0x00	0x2B 0x70
读 SPI Flash 指令校验通过反馈指令	0x5A 0xA5	0x04	0x65	0xFF	0x6A 0xA0
读 SPI Flash 指令校验失败反馈指令	0x5A 0xA5	0x04	0x66	0x00	0x2A 0xE0
校验 SPI Flash 指令校验通过反馈指令	0x5A 0xA5	0x04	0x66	0xFF	0x6A 0x50
校验 SPI Flash 指令校验失败反馈指令	0x5A 0xA5	0x04	0x66	0x00	0x2A 0x10

12.3. CRC 计算及代码

12.3.1.CRC16 的产生

CRC16 的产生可参考 CRC 计算代码。如下表，读写指令的指令码项、操作地址项和内容项参与生成 CRC16 的计算。

表 12-3：通信指令表

指令	帧头 0x5A 0xA5 (2 Bytes)	固定 0x00 (1 Byte)	长度 (2 Byte)	指令码 (1 Byte)	操作地址 (4 Bytes)	内容 (n Bytes， 最大=2048)	CRC 0xXX 0xXX (2 Bytes)

12.3.2.CRC32 的产生

代码如下：函数 GetCrc32()返回 32 位校验码。

```
uint32_t CRC32_Table[256];

uint32_t GetCrc32(uint8_t *InStr,uint32_t len,uint32_t value)
{
    uint32_t i;
    uint32_t Crc;
    uint8_t *pch;

    Crc=value;
    pch = InStr;

    for(i=0; i<len; i++)
    {
        Crc = (Crc >> 8) ^ CRC32_Table[(Crc&0xFF)^(*pch)];
        pch++;
    }
    return Crc;
}

void Make_CRC32_Table(void)
{
    uint32_t c;
    uint32_t i = 0;
    uint32_t bit = 0;

    for(i = 0; i < 256; i++)
    {
        c   = (uint32_t)i;

        for(bit = 0; bit < 8; bit++)
        {
            if(c&1)
            {
                c = (c >> 1)^(0xEDB88320);
            }
```

```
                else
                {
                    c =   c >> 1;
                }
            }
            CRC32_Table[i] = c;
        }
    }
```

注：InStr 为 8 位无符号整型数组，里面存放需要校验的数据；len 为需要校验的数据的长度；
当需要校验较多的数据时，InStr 数组一次放不下，就需要将数据分为多包进行校验，value 就是分多次校验数据时上一次校验产生的 32 位 CRC，若该次校验为第一次，则 value = 0 。

12.4. Flash_RW-II 软件使用说明

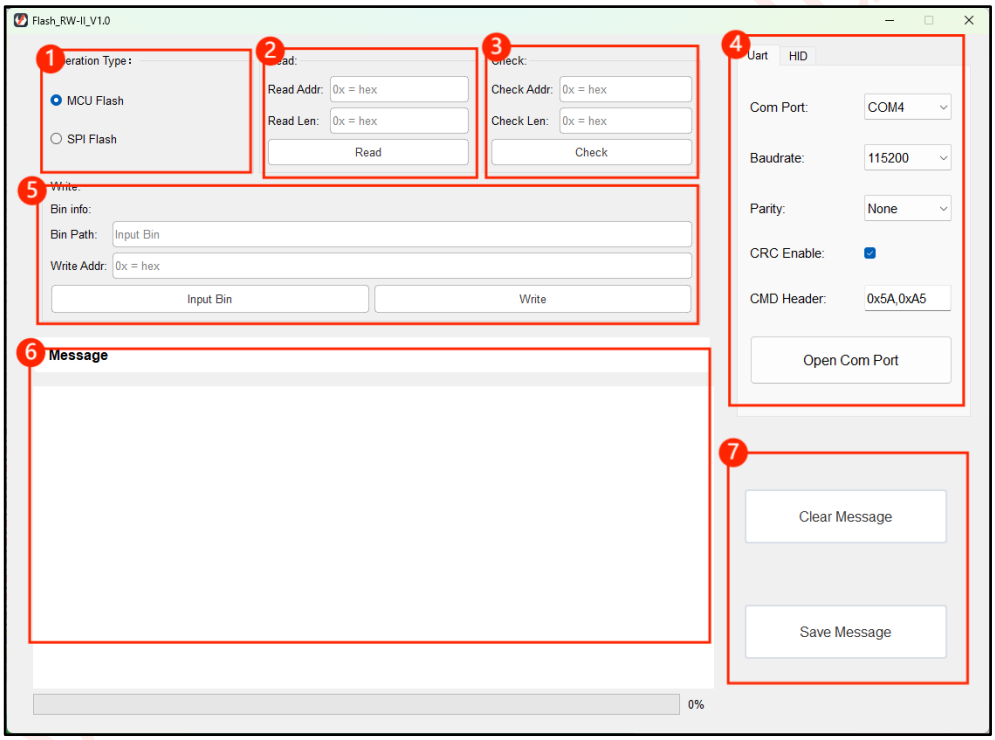


图 12-1：Flash_RW-II 软件主界面

1、操作对象选择区：勾选 MCU Flash 则表示读写操作对象均为 MCU Flash；勾选 SPI Flash 则表示读写操作对象均为 SPI Flash。

2、读取操作参数设置区：

Read Addr：输入读操作的起始地址。

Read Len：输入读取的长度，最大不超过 2Kbytes。

这两个输入框可以输入 10 进制或者 16 进制的地址或长度，区别在于 16 进制数前置需为 0x 。如下图，两图的操作相同。

图 12-2：地址输入框

Read：点击进行读取操作，数据显示在下方信息区

3、校验操作参数设置区：

Check Addr:输入校验操作的起始地址。

Check Len：输入校验操作的数据长度。

Check：点击开始校验，返回 32 位校验码在下方信息区。

4、通信端口选择区：后续章节说明。

Open Com Port：点击开启端口。

5、写取操作参数设置区：

Write Addr：输入写操作的起始地址。

Input bin ：点击添加需要写入的 bin 文件。

Write：点击开始写操作。下方信息区会根据成功与否返回 ‘Write flash over!’ 或者 ‘Write flash time over!’ 的提示。

6、信息区：显示读回的数据或者写入的结果。

7、

Clear Message：点击清空信息区。

Save Message：点击保存通信过程的指令到 TXT 文档里。

12.5. 通过串口读写

1、需烧录支持串口读写外部 SPI Flash 和 MCU Flash 的 Mcu_Code.bin ,若客户有乐升二代公版代码，可自行在代码工程内 module_select.h 头文件内更改 UARTBUS_OPTION 参数的宏定义为 0,选中串口通信模式。

```
module_select.h
13
14 #define IIC_BUS              0        // Communication,
15 #define UARTBUS_OPTION       0        // Communication,
16 #define DualFlash            0
17 #define encoder_on           0        // 1:Open 0:Close
18 #define Touch_selection      0        // 0:CTP  1:RTP
```

图 12-3：选择串口通信模式

2、连接好串口：将图 12-4 1 处的串口 1 与图 12-5 的 USB 转 TTL 模块连接好，使用图 12-4 2 处的 CON1 给板子供电。

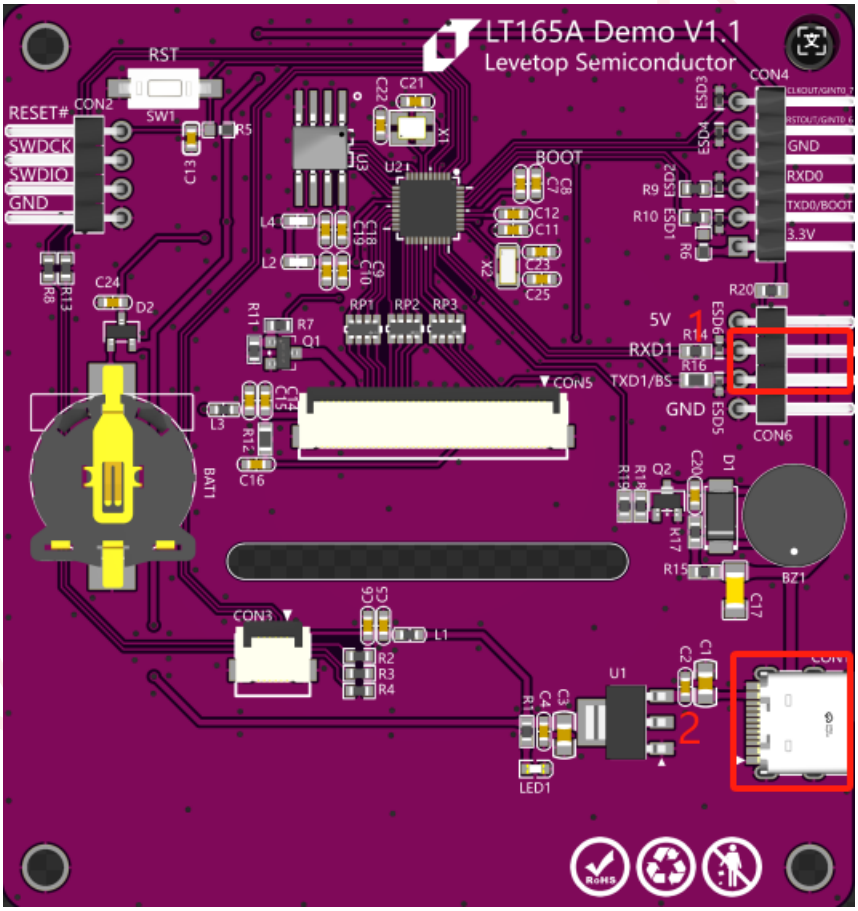


图 12-4：串口通信模式设置

图 12-5：USB 转 TTL 模块

3、打开 Flash_RW-II 软件，右上角通信端口选择区确保处于 Uart 页，选择与串口屏对应的端口，确认波特率帧头等参数，最后点击 Open Com Port。

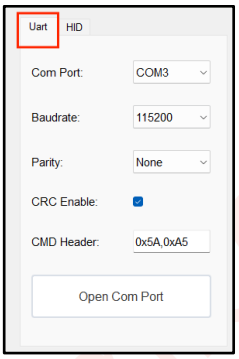


图 12-6：选择串口

使用 Flash_RW-II 软件更新 UI 工程

限制：该功能只能更新 UartTFT-II_Flash.bin ，不能更新 Mcu_code.bin 。

该功能的实现基于 写 SPI Flash 指令 ，以下部分通过串口模拟演示，更新过程也可以是主控 MCU 发送 UI 工程数据。

更新步骤如下：

1. 通过串口发送指令至串口屏：5A A5 07 10 7042 0001 9E D7 ，串口屏进入接收模式，断开串口软件的连接。
2. 打开 Flash_RW-II 软件，点击 Open Com Port 连接串口，点击 Input Bin 添加需要更新的 UI 工程文件

图 12-7：Flash_RW-II 软件更新 UI 工程

3. 在 Write Addr 填入地址 0 ，确保写入目标是 SPI Flash ，点击 Write，开是更新。
4. 更新完成后，发送指令至串口屏：5A A5 07 10 7042 0000 5F 17 ，串口屏回归正常工作状态。

13. 横竖屏 UI 工程说明

用途：双 UI 工程切换功能主要用于切换同一个 UI 的横竖屏模式。
限制：横竖屏切换仅支持横屏 0°，竖屏 90°，不支持 180°和 270°。

13.1. 横竖屏 UI 工程要求

13.1.1.页面内容和页码要求

由于涉及切换的两个 UI 不是独立的，所以这两个 UI 工程内的页面的页码和各控件的变量地址均需要相同。如下图，横屏 UI 和竖屏 UI 的页码和页面内容一一对应，竖屏工程内首页的页码是 page0000 那横屏工程的首页也应当是 page0000。

页码	名称
0000	竖屏_首页
0001	竖屏_页面1

页码	名称
0000	横屏_首页
0001	横屏_页面1

图 13-1：横竖屏 UI 页面页码要求

这样做的目的是，在 UI 工程切换后，使得画面内容不会发生改变。例如若横屏工程的首页在 page0001，此时屏幕画面处于竖屏首页（page0000），如果切换横屏 UI，切换后的屏幕画面也会处于横屏的 page0000，也就是”横屏_页面 1”，页面内容就发生了改变。

13.1.2.控件要求

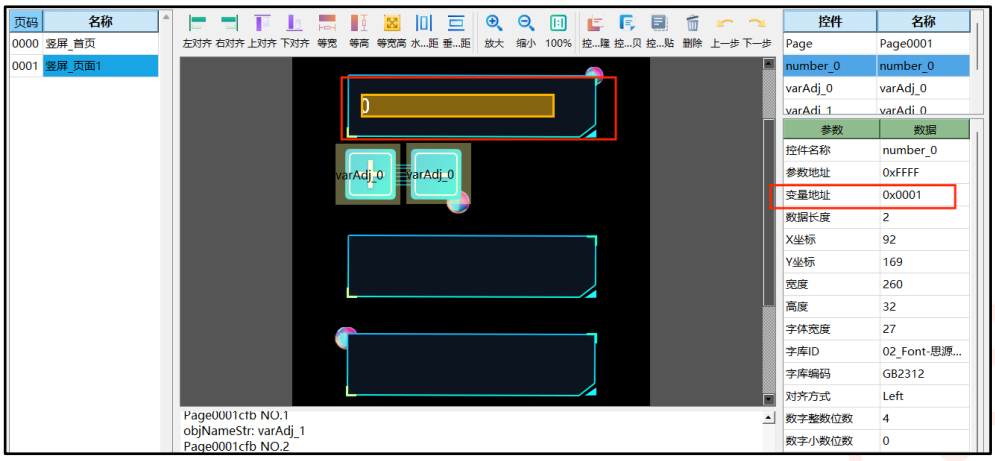


图 13-2：竖屏 UI

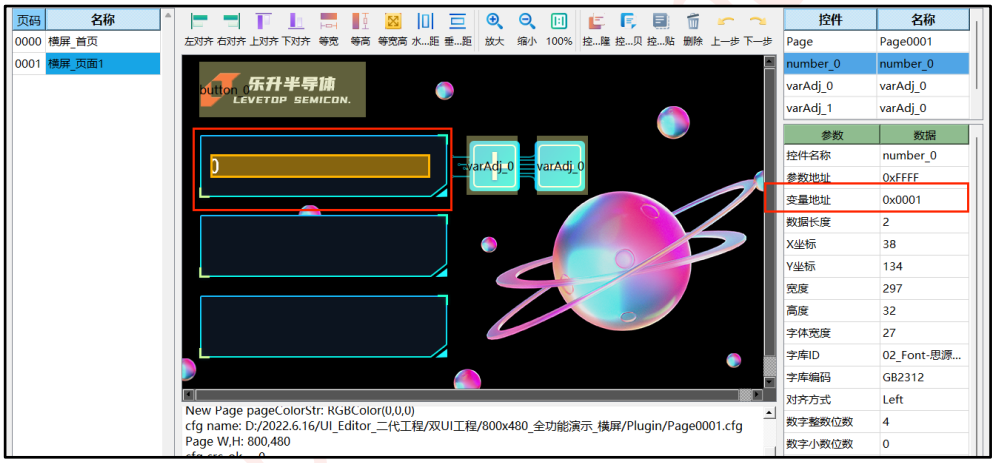


图 13-3：横屏 UI

如上两工程，在 page0001 均存在一个字库数字控件，这两个字库数字控件的参数需一致（坐标可不同）。

13.2. 横竖屏工程的合并及烧录过程

横竖屏的 UI 工程需要烧录到同一个 SPI Flash 里，所以烧录前需要将两个 UI 工程文件，即两个 UartTFT-II_Flash.bin 合并成一个。

13.2.1.合并工程文件

合并文件必须使用乐升的 bin 文件合并工具，该工具在合成时会重新产生 CRC，若使用其他合并工具则烧录时无法通过 CRC 校验。具体获取方式如下，在乐升官网下载 “UI_BinFiles_Merge”。

图 13-4：下载软件包

解压后，在文件夹内找到“UI_BinFiles_Merge.exe”，双击打开或者以管理员身份运行。

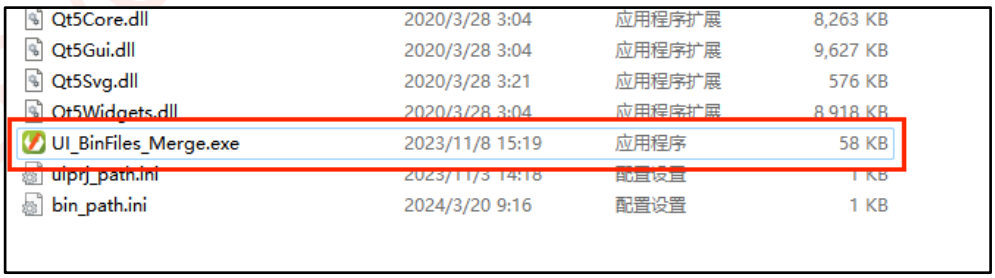


图 13-5：打开软件工具

软件界面如下

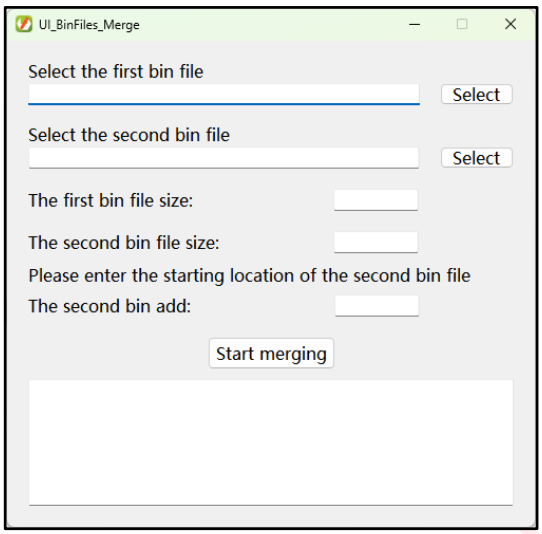


图 13-6：UI_BinFiles_Merge 软件界面

操作流程如下：

1. 点击下图 Select，添加第一个工程的 UartTFT-II_Flash.bin，该**文件必须选择横屏 UI 文件。**

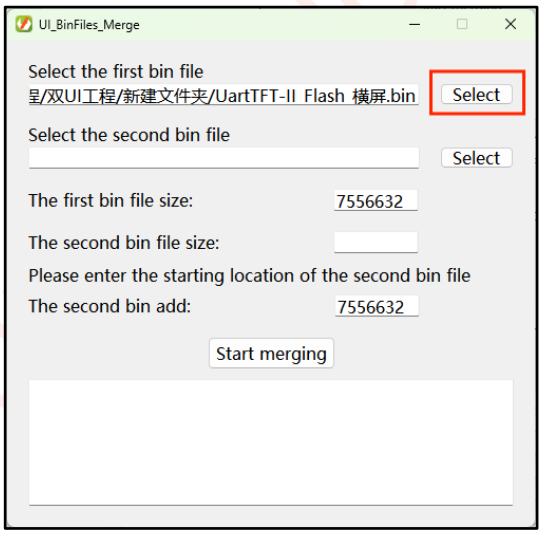


图 13-7：添加 file1

2. 点击下图 Select，添加第二个工程的 UartTFT-II_Flash.bin，该**文件必须选择竖屏 UI 文件。**

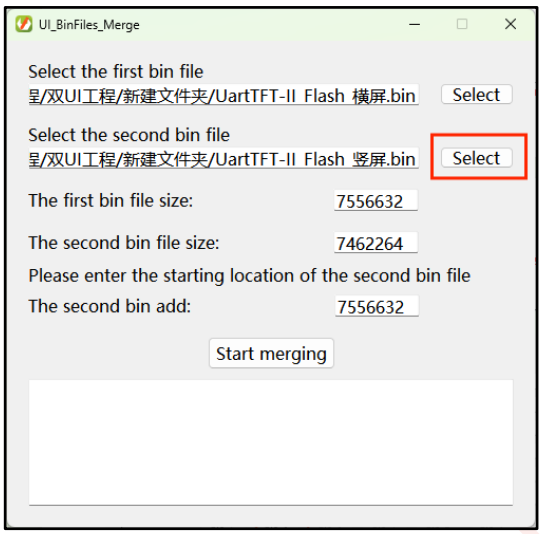


图 13-8：添加 file2

3. 点击 Start merging,产生合成文件

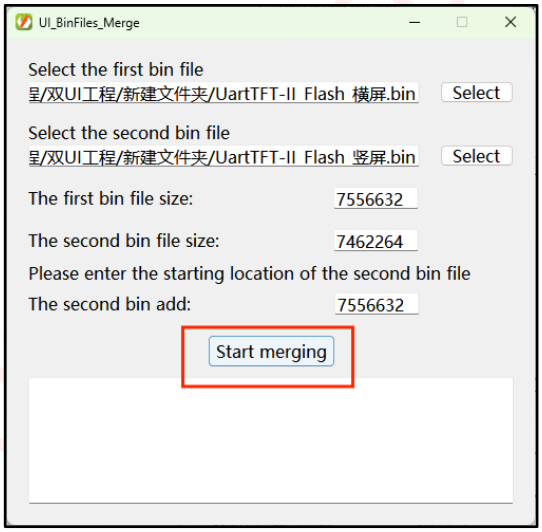


图 13-9：合成文件

4. 获取 second_UI_add ，如下图，红框内均为 second_UI_add 的值，7556632 为十进制，0x00734E18 为 16 进制，该地址后续需要用到。

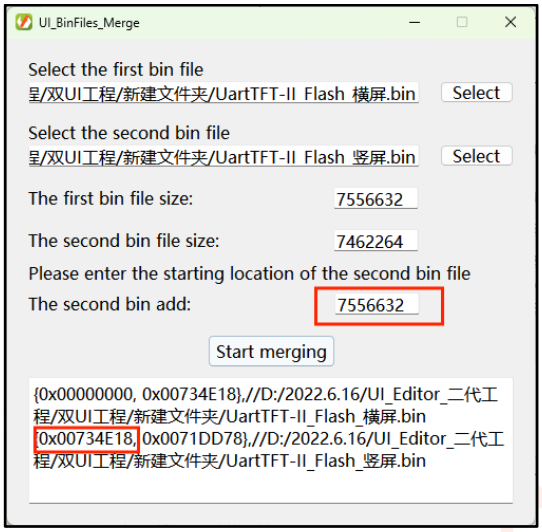


图 13-10：合成成功

可以看到，产生了两个文件，如下图，UartTFT-II_Flash.bin 里包含横竖屏两个 UI 工程的 bin 文件 ,UartTFT-II_Flash-Addr.txt 是合成文件的结构信息。

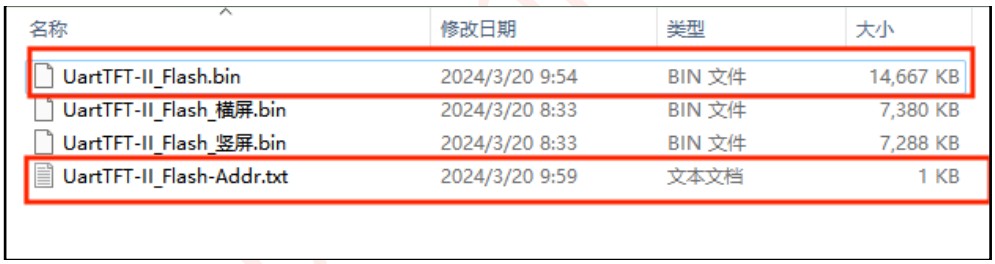


图 13-11：文件说明

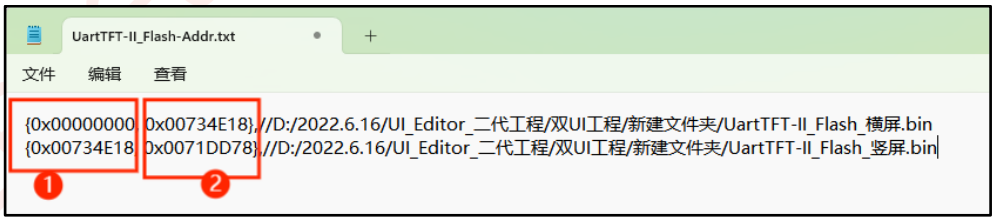


图 13-12：合并文件的结构说明

如上图标注 1，该列为地址列，即该子 bin 文件在合成的 bin 文件内的位置。标注 2 为长度列，即该子 bin 文件的长度。

13.2.2.更改 mcu_code

使用横竖 UI 切换功能需要更改 mcu_code，需要将前文获取到的 second_UI_add 更改到代码中，否则发送切换指令时无法正常切换，位置如下图。

module_select.h

```
18 #define LT_TFT_DIR   1 // only for Levetop public screen: 0:0°  1:90°  2:180°  3:270°
19 #define LT_TFT_COLOR 0 // TFT interface color format: 0:16bit/pixel(RGB565)  1:24bit/pixel(RGB888
20
21 // peripheral
22 #define LT_PERI_SELECT  1 // MCU peripheral, 1:LT165A     2:LT165B  0xFF:custom type
23 #define LT_TOUCH_SELECT   1 // 1:CTP_default   2:CTP
24
25 #define LT_TOUCH_FLAG   2 // touch panel type: 0:Null  1:RTP  2:CTP
26 #define LT_TOUCH_BUZZER 0 // buzzer feedback on touch: 0:disable  1:enable
27 #define LT_ENCODER      0 // encoder: 0:disable  1:enable
28 #define LT_SD_UPDATE    0 // SDcard update: 0:disable  1:enable
29 #define LT_Rtc_Enanbe   1 // rtc init: 0:disable  1:enable
30 #define LT_WAV_Enanbe   0 //
31
32 // UI
33 // #define LT_BG_ZIP            1       // zip picture as background: 0:disable  1:enable
34 #define LT_PIC_BYTE_SWAP 1 // 16bit picture byte swap(zip_RGB565/pure_color invalid)
35 // #define LT_TFT_TE        0 // screen TE mode
36 // #define LT_PAGE_FULLBUFF 0 // page full buff
37 // #define LT_PSRAM         0 //(unsupported)
38 // #define LT_BBM 0 // Bad Block Management(only for NandFlash and bootloader V2.0)
39
40 // custom parameters
41 #define Secondary_screen    0                   // Case secondary screen: 0:disable  1:enable
42 #define LT_EFLASH_BASE_ADDR (0x60064000 + 80000) // 400K
43 #define LT_UI_BASE_ADDR     (0x60080000) // 512K
44
45 #define second_UI_add     (0x00734E18)
46
47 // debug
```

图 13-13：填入 second_UI_add

将合成的 UartTFT-II_Flash.bin 和重新产生的 MCU_Code.bin 烧录至串口屏。按客户板子的正常方式烧录即可。

13.3. 发指令切换 UI 工程

烧录完成重新上电后，串口屏默认显示横屏工程，且 **横竖屏 UI 工程切换寄存器** 0x7040 内的值为 0，通过串口往 0x7040 发送如下指令可切换横屏或竖屏 UI 工程：

横屏切换成竖屏：5A A5 07 10 7040 0001 3F 17

竖屏切换成横屏：5A A5 07 10 7040 0000 FE D7

14. 控件叠加显示说明

14.1. 叠加限制说明

注：叠加功能目前仅支持使用 **UI_Editor-Lite_V3.00** 及以上版本上位机进行 UI 制作和编译。

1. 支持叠加功能的控件包括

前景控件：图片数字控件，小图标控件（不支持压缩）。

背景控件：小图标控件，动图控件。

2. 开启叠加的控件面积限制与 **png 大小限制一致**详情请查看 IC 类型及其参数限制。
3. 背景控件应当**大于**前景控件面积，前景控件应当**完全处于**背景控件范围内。
4. 叠加**不支持参数指针。**
5. **仅支持 2 层叠加**，不支持 3 层及以上层数叠加。
6. 在背景控件范围不限前景控件的类型和数量，但前景控件之间不能相互交叉叠加。

14.2. 叠加使用说明

对于 LT165 系列的 IC 进行控件叠加时，**不需要进行其他的设置**，只需要将前景控件完全置于背景控件范围内，同时注意**显示内容不能超出背景控件范围。**以小图标控件做背景控件，图片数字控件和小图标控件为前景控件为例子，右边的前景控件的大小超过背景控件的范围，只有像左边**完全置于背景控件范围内的才是正确**的做法。

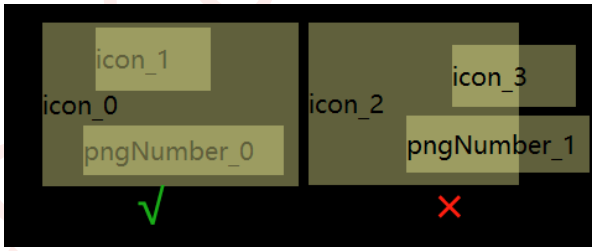


图 14-1：控件叠加

15. 注意事项说明

15.1. 第二代串口屏使用概括及常用文件说明

15.1.1.第二代串口屏使用的概括

第二代串口屏是分不同的页面设置，每个页面由底图与多个控件组成，在 UI_Editor_II 内设置好每页功能控件后，所有的显示和操作会根据 UI_Editor_II 设定的参数来工作，主控可通过主控修改变量或按键控制去实现页面内容的更新或页面的跳转。

控件联动是通过多控件共地址实现的，例如滑条控制亮度，只需要把滑条的地址关联到亮度寄存器，设置合适的范围（一般 10~63），即可实现控制亮度。共地址的控件是不限个数不限页的，只要控件设置没有相互干扰。存在以下设置会出现冲突，数字控件共地址时数据类型不同、小数位整数位不同；多控件共用地址时设置了多个不同的初值，上电后只会显示最后读取的那个初值。

15.1.2.常用文件说明

由素材到一个相对完整的工程，需要用户使用 UI_Editor-Lite 进行编辑。编辑过程中会接触到一些文件或者是程序，这里汇总说明一下。

bootloader 程序：用于下载 MCU_Code.bin 和 UartTFT-II_Flash.bin 文件。一般来说，只有需要更换下载方式的时候才会更换 bootloader 程序。

MCU_Code.bin 文件或程序：UI_Editor-Lite 控件实现的基础支持程序，用户可在此程序内编程，自定义一些基础程序不支持的操作。编译 MCU_Code 的程序产生，文件比较小，通常小于 256KB，烧录至单片机的程序空间。

UartTFT-II_Flash.bin 文件：用户素材和 UI_Editor-Lite 控件参数结合在一起的文件，里面包含了工程用到的素材以及用户在 UI_Editor-Lite 上编辑的内容。通过 UI_Editor-Lite 编译工程产生，文件通常较大，几十至上百 MB 不等（看素材的大小），烧录至外部 SPI Flash 内。

15.2. 以旧工程素材创建新工程说明

如果需要使用旧工程的素材，可以直接复制整个旧工程的文件夹，然后把 Plugin 文件夹的内容清空，同时删除下图红框内的工程文件，只保留 7 个素材文件夹，再重新新建工程即可。

图 15-1：删除旧工程原有配置文件

15.3. 工程备份说明

一般来说，备份工程需要将整个工程压缩为一个压缩包或者是复制整个工程文件夹，备份的工程处于一个不会被更改的状态，即使原工程受到了不可逆转的改变，我们也可以通过备份重新生成一个与原工程完全相同的工程。建议用户在使用 UI_Editor-Lite 编辑 UI 工程前，手动进行备份。

如下图，Plugin 文件夹内存放着工程内每一页的配置文件，配置文件分为 3 种，包括临时保存文件，最终配置文件以及工程备份文件。

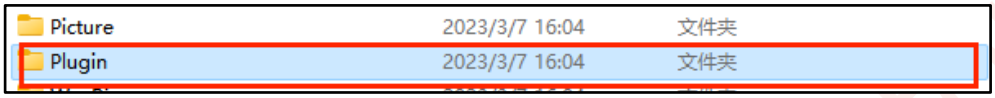


图 15-2：Plugin 文件夹

如下图，临时保存文件用于回退上一步（undo）和下一步（redo）。最终配置文件用于导入该页的控件配置，只有在编译或者保存时才会修改该文件。工程备份文件每一页有 2 个，两者循环备份，根据备份文件的修改日期就可以确定是哪一天的工程版本，例如若修改日期为 3 月 1 日，则该工程备份文件备份的是 3 月 1 日编辑工程时打开工程之前的版本。

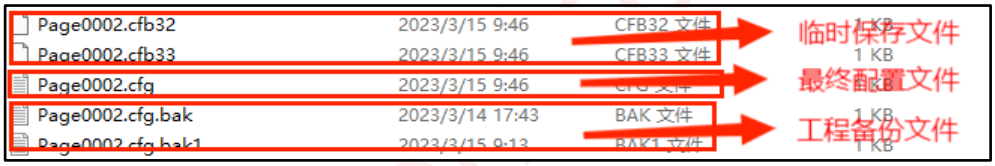


图 15-3：Plugin 文件夹内容

下面说明 UI_Editor-Lite 自动备份的过程以及限制。

UI_Editor-Lite 目前具有回退两个工程版本的能力，假设用户需要打开编辑 A 工程，那么我们的软件会在第一次打开 A 工程时将 A 工程每一页的配置分别保存至对应工程备份文件即 PageNNNN.cfg.bak 文件内。对 A 工程编辑编译后，A 工程就不是原先的 A 工程了，因为最终配置文件即 cfg 内容已经改变，这里定义为 A-1 工程。如果用户没有另外备份该工程，则可以通过以下手动还原过程将 A-1 工程还原至 A 工程。

手动还原过程如下：

首先关闭工程，然后进入工程的 Plugin 文件夹内，将想要还原的页的 cfg 文件删除，然后将备份文件重命名为最终配置文件，如将 **Page0000.cfg.bak** 文件重命名为 **Page0000.cfg** ，修改完成后重新打开工程，这样对应页就还原到了旧工程。

15.4. 画面旋转操作设置说明

如果我们需要把一个 800*480 的横屏达到竖用的目的，那我们只需要把原本的 800*480 的工程改成 480*800 的工程外加设置一个旋转角度参数即可。通俗地讲，就是用户想要什么样的工程就在上位机编辑出什么样的工程，然后把工程设置内的参数 Rotate 修改即可，所有控件的旋转均已在程序内完成，用户无需去自行旋转素材，只有底图需要用户重新设计。参数 Rotate 是依据屏幕与工程画面的角度关系得出来的，可参考下图说明。参数 Rotate 设置位置如下图所示。

注：

1、Rotate 设置成需要旋转的角度，屏的分辨率无需改动。

2、LT165-MCU 接口屏工程的旋转需要修改 MCU_Code 代码实现，Rotate 设置为 0°。

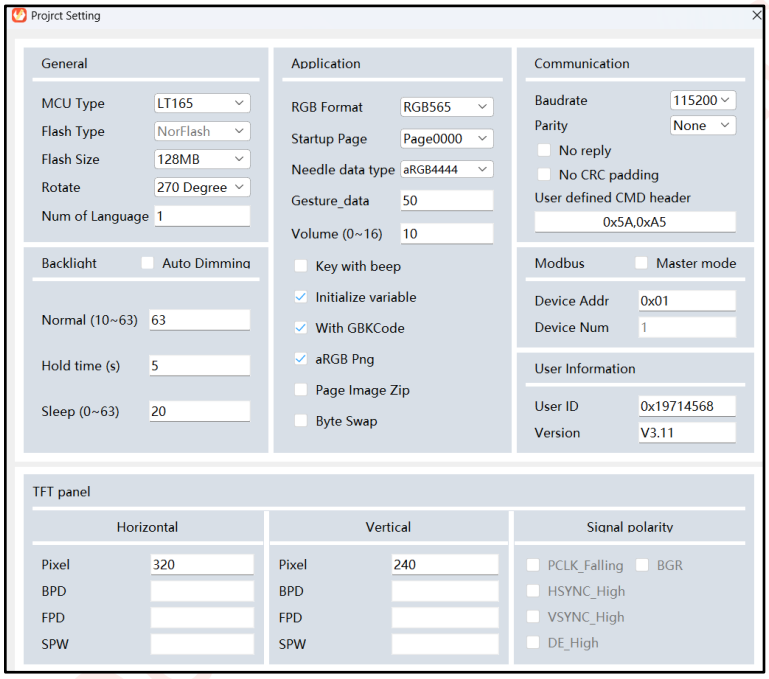


图 15-4：旋转参数设置

如下图所示，旋转基点如左图 A 点所示，旋转方向为顺时针旋转。下左图为 800*480 无旋转时的显示，右图为顺时针旋转 270°（逆时针旋转 90°）后得到的显示。

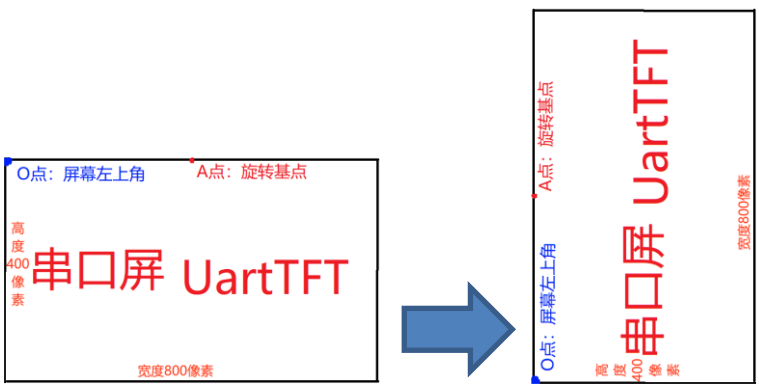


图 15-5：顺时针旋转 270°

15.5. UartTFT-II_Flash.bin 大小与素材关系说明

15.5.1.图片素材转 bin 文件说明

图片转换：以一张 800x480 的图片为例，选择 RGB565 格式时，转化为 bin 文件的大小为 750KB，计算方法为 800*480*2/1024 = 750KB;选择 RGB888 格式时,转化为 bin 文件的大小为 800*480*3/1024 =1125KB。

Gif 转换：UI_Editor-Lite 将 Gif 素材转化为 bin 文件的方式是将其按帧拆分，每一帧为一张图片，然后转化为 bin 文件，每一帧的 bin 文件按照顺序连接形成整个 Gif 的 bin 文件。不同编号的 Gif 的 bin 文件按照又编号顺序继续连接，最后并入 UartTFT-II_Flash.bin 文件。

15.5.2.UartTFT-II_Flash.bin 组成

UartTFT-II_Flash.bin 的组成包括字库 bin、Gif bin、图片 bin、Wav bin、页信息等。其中图片和 Gif 对 UartTFT-II_Flash.bin 的大小影响最大，帧数高的 Gif 即使在电脑内只占据几 MB 的空间，转化出的 bin 文件大小却有十几 MB，如下图，Gif 转换成 bin 文件后大小暴涨了 10.65 倍。

图 15-6：Gif 转 bin

在设计工程时，需要注意 SPI Flash 的大小是否支持存放 UartTFT-II_Flash.bin 。

15.6. 数据类型说明

这里说明一下常用的 7 种数据类型所占用的字节数量、长度、数值范围等。

表 15-1：数据类型说明表

	变量地址	长度	数据类型	理论最大值	数值范围
起始地址	writeAddr	8Bytes	longlong	0x7FFF	-2^63~2^63-1
	writeAddr+ 0x0001			0xFFFF	
	writeAddr+ 0x0002			0xFFFF	
	writeAddr+ 0x0003			0xFFFF	
起始地址	writeAddr	4Bytes	int	0x7FFF	-2^31 ~2^31-1
	writeAddr+ 0x0001			0xFFFF	
起始地址	writeAddr	4Bytes	uint	0xFFFF	0~2^32-1
	writeAddr+ 0x0001			0xFFFF	
起始地址	writeAddr	2Bytes	short	0x7FFF	-2^15~2^15-1
起始地址	writeAddr	2Bytes	ushort	0xFFFF	0~2^16-1
起始地址	writeAddr-H	1Bytes	char_H8	0x7F	-2^7~2^7-1
起始地址	writeAddr-L	1Bytes	char	0x7F	-2^7~2^7-1
起始地址	writeAddr-H	1Bytes	uchar_H8	0xFF	0~2^8-1
起始地址	writeAddr -L	1Bytes	uchar	0xFF	0~2^8-1

注：writeAddr-H 表示该地址的高 8 位，writeAddr-L 表示该地址的低 8 位。

15.7. 整数位和小数位设置

小数采用定点小数的表示，用户自定义小数位数。如 0x3039(12345)，规定小数为 2 位时，表示 123.45。

在使用字库数字和图片数字时，小数位数和整数位数加起来不能超过数据类型的位数。short&ushort：5 位，int&uint：10 位，longlong：19 位。而且输入 defaultNumber 时，整数部分和小数部分的位数都不能超过预设位数，输入数值大小不能超过以下设定。

当数据类型为 short 时，整数部分和小数部分的任何一个位都不能大于 32767 的对应位。例如设置 3 位整数 2 位小数时，极限值就是 327.67，大于这个数值的数在这个数字控件是会显示错误的，int 和 longlong 也是如此。

15.8. 同组的 Icon 宽高说明

图片数字使用到【0、1、2、3、4、5、6、7、8、9、小数点】等素材，数字时钟用到【0、1、2、3、4、5、6、7、8、9、：、/、/、/】和【Sun、Mon、Tues、Wed、Thur、Fri、Sat】等素材。

图片数字使用到的素材中所有数字素材的图片的宽高必须一致，但是数字素材和小数点的宽度可以不一致，高度要一致。

数字时钟使用到的素材数字素材的图片的宽高要一致，星期素材的图片素材宽高要一致，但是数字素材与星期素材以及其他素材（如冒号、小数点等）的图片的宽度可以不一致，高度要一致。

部分控件可以设置 unpressedIcon（无按压）和 pressedIcon（按压），这两个参数的图片宽高得一致。

15.9. 删除控件底图说明

如果想要去掉某页或者是某控件的底图时，双击该参数进入选择界面，然后点击弹窗右上角×或者是右下角取消退出选择，进入可编辑状态，全选该参数内容，然后删除或者 Backspace 把参数内容替换成空，再 enter 确认。如下图例程：

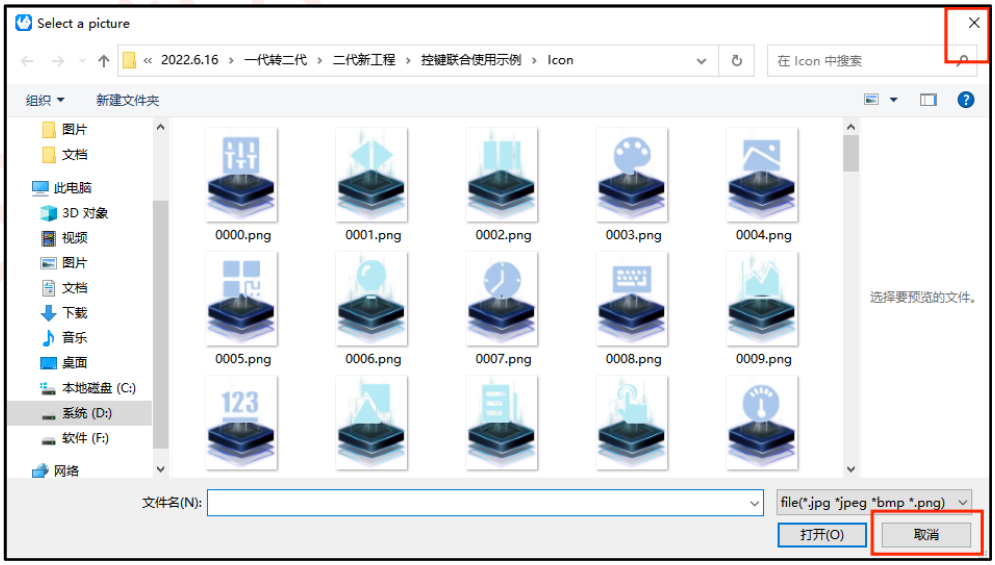
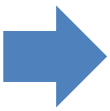


图 15-7：取消选择

Para	Data
Description	varAdj_3
X	687
Y	100
W	97
H	97
WriteAddr	0x7001
AdjStep	1
MinValue	60
MaxValue	60
DataType	uchar
AdjMode	+
Overflow	Loop
LongPress	Once
NormalIcon	0028.png
PressIcon	
EnableReport	Disable



Para	Data
Description	varAdj_3
X	687
Y	100
W	97
H	97
WriteAddr	0x7001
AdjStep	1
MinValue	60
MaxValue	60
DataType	uchar
AdjMode	+
Overflow	Loop
LongPress	Once
NormalIcon	
PressIcon	
EnableReport	Disable

图 15-8：删除参数内容

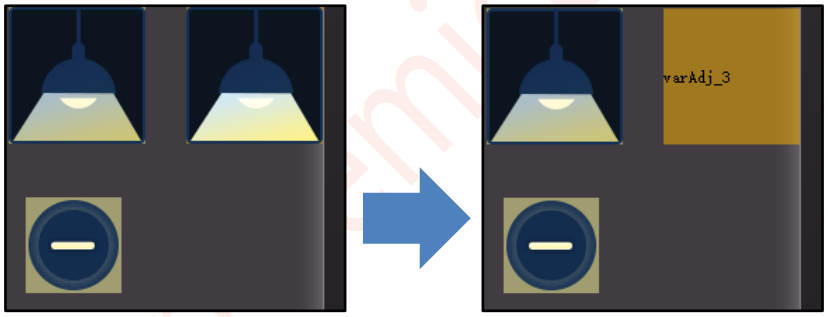


图 15-9：底图删除成功

15.10. 控件初值设置说明

多控件关联共用使同一个地址时，默认值设置得一致。例如两个同地址的字库数字，不能一个控件的 defaultText 是 123，另一个的是 1234。

15.11. 控件堆叠的说明

在编辑工程时，带有触控属性的控件不能相互堆叠（一定会出错），显示控件中只有同组 Icon 才可以叠加。

虚拟控件：是指一些带触控属性的控件，它们本身不需要添加底图或者是可以添加但不添加底图就能正常起作用，这种控件在串口屏屏幕上不显示，单击对应区域会起作用。虚拟控件包括不添加底图的按键、弹出窗体、变量调节和多变量操作，无法添加底图的数字键盘、英文键盘、中文键盘和键盘按键。

虚拟控件可以和显示控件 1 +1 式堆叠使用。

15.12. 控件超出屏幕范围的说明

所有控件在设置宽高时应当注意，增加了图片的控件其宽高会自适应图片的宽高，此时无法修改；没有添加图片的控件其宽高设置时不能超出屏幕范围。即**控件左上角坐标 X（Y） + 控件宽（高）≤ 屏幕宽（高）。**

15.13. 电脑屏幕分辨率导致软件字体显示不全的说明

由于不同电脑分辨率不同，有部分电脑会出现如下图的症状。此解决方法仅针对 WIN10 系统。

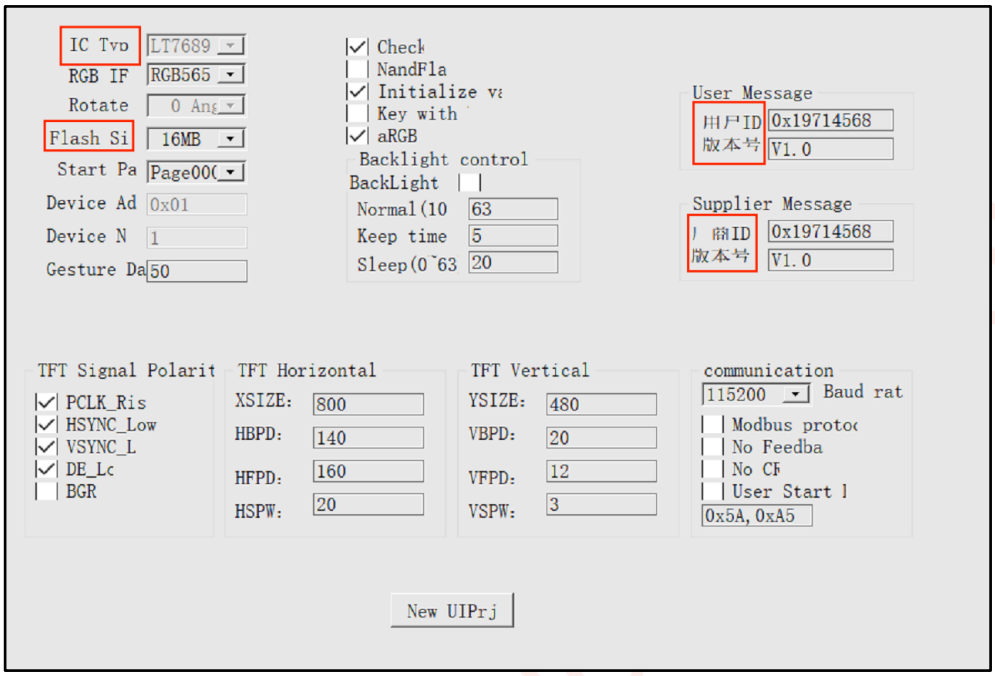


图 15-10：软件内字体显示不全

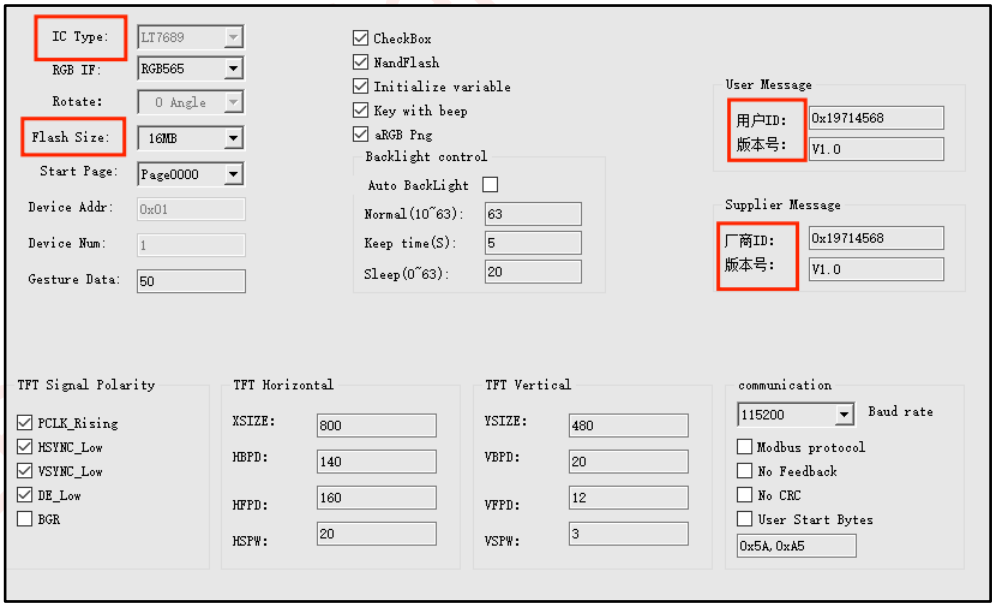


图 15-11：正常显示

解决方法步骤如下：

1、关闭软件，然后选中软件 exe 程序，单击右键，再点击属性。如下图所示

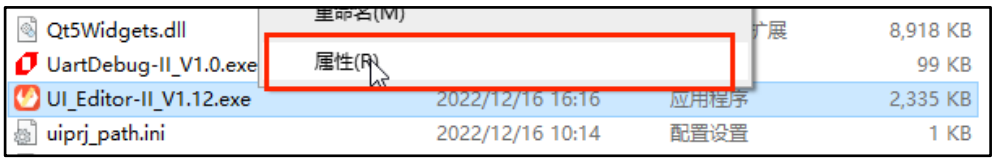


图 15-12：进入属性界面

2、点击“兼容性”，再点击“更改高 DPI 设置”，进入另一个弹窗。

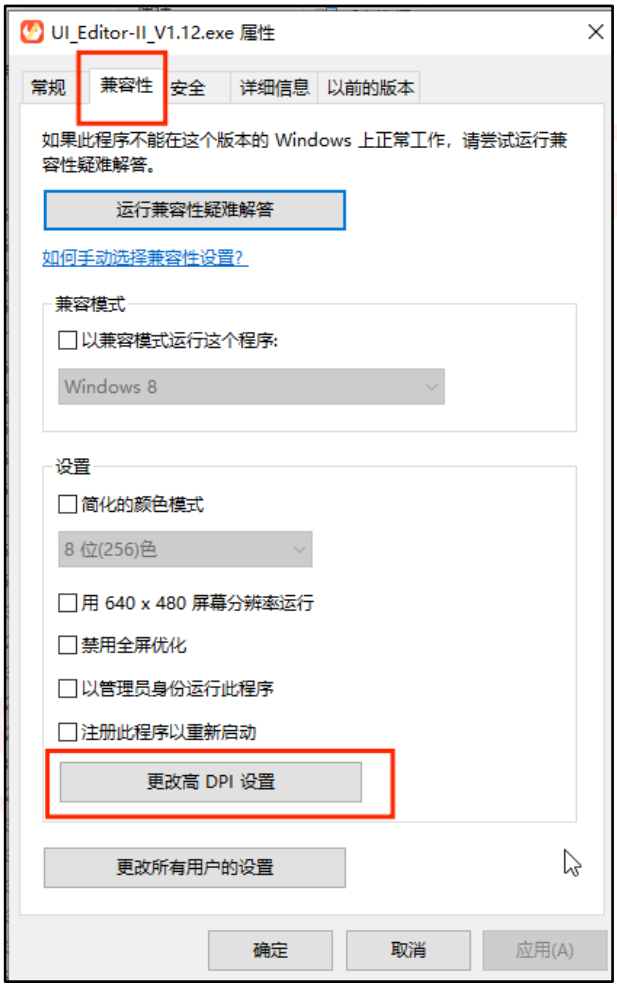


图 15-13：更改 DPI 设置

3、勾选“替代高 DPI 缩放行为”，点击下方箭头，选择“系统（增强）”，最后点击确认。

图 15-14：更改属性

4、上一步确认后，回到下图界面，点击应用再点击确认，完成修改。再次打开软件，字体正常显示。

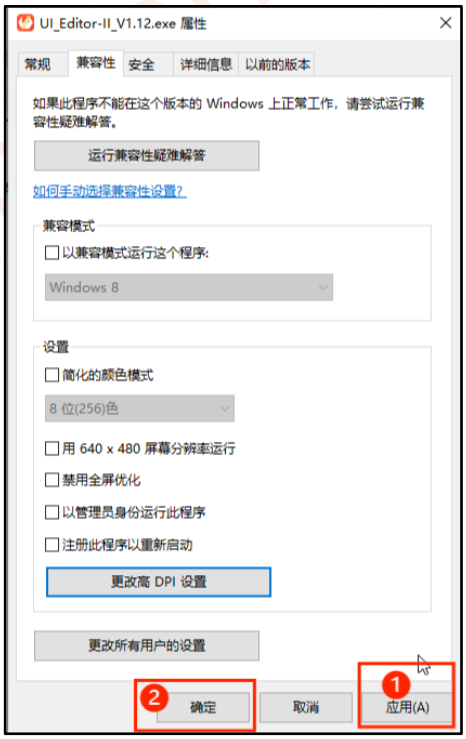


图 15-15：确认修改属性

15.14. UI_Editor-Lite 使用环境说明

建议用户的电脑安装 window10 及以上的系统，在使用软件编辑工程时，确保软件处于全屏模式（尤其是屏幕分辨率小的电脑屏幕）。

15.15. 各类素材和文件的命名说明

所有的素材、控件、页名称和工程名称等的命名都不能存在下列符号。对于小数点这个符号，名称中只能有一个作为格式后缀与名字的分隔符。

表 15-2：命名时禁止输入的符号

输入模式	EN	EN	EN	CN/EN	CN/EN	EN	EN	CN/EN	EN	EN
符号名称	反斜杠	斜杠/除号	冒号	星号/乘号	问号	小于号	大于号	竖线	句号/小数点	逗号
符号	\	/	:	*	?	<	>	|	.	,

15.16. 素材库说明

乐升半导体可以为用户提供一个素材库，里面包含了常用的图标、成组的图片数字素材和部分字库。用户如果需要可以咨询乐升半导体的工程师。

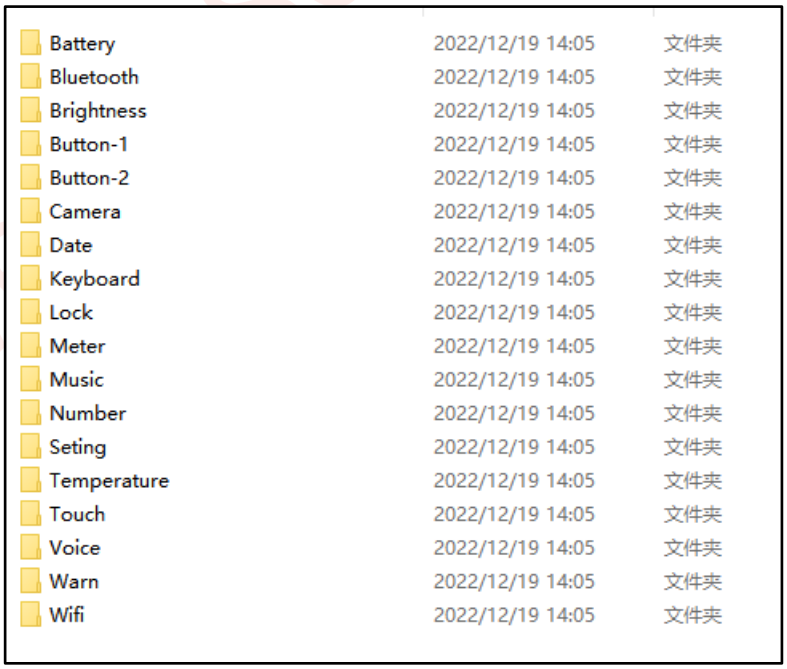


图 15-16：素材库

15.17. dataFormat 说明

15.17.1. 不同数据格式的结构说明

1、LT165x 支持的数据格式如下：

RGB565：上文已说明。

αRGB4444：上文已说明。

RGB565_zip：RGB565 的压缩格式，可以省下一部分空间。

αRGB4444_zip：aRGB4444 的压缩格式，可以省下一部分空间。

αRGB8565：每个像素由 24bit 的数据表示，数据结构如下表。
以下的数据格式均针对图片中的单个像素点。

RGB888：每个像素由 24bit 的数据表示，数据结构如下表。

表 15-3：RGB888 数据结构

数据格式	红色 Red	绿色 Green	蓝色 Blue
RGB888	bit 23~16	bit 15~8	bit 7~0
	R7~R0	G7~G0	B7~B0

RGB565：每个像素由 16bit 的数据表示，数据结构如下表。

表 15-4：RGB565 数据结构

数据格式	红色 Red	绿色 Green	蓝色 Blue
RGB565	bit 15~11	bit 10~5	bit 4~0
	R7~R3	G7~G2	B7~B3

αRGB4444：每个像素由 16bit 的数据表示，数据结构如下表。

表 15-5：αRGB4444 数据结构

数据格式	透明度 α	红色 Red	绿色 Green	蓝色 Blue
αRGB4444	bit 15~12	bit 11~8	bit 7~4	bit 3~0
	α3~α0	R7~R4	G7~G4	B7~B4

α3α2α1α0：0→0，1→2/32，2→4/32，3→6/32，4→8/32，……，12→24/32，13→26/32，14→28/32，15→100%.

表 15-6：αRGB8565 数据结构

数据格式	透明度 α	红色 Red	绿色 Green	蓝色 Blue
αRGB8565	bit 23~16	bit 15~11	bit 10~5	bit 4~0
	α 7~0	R7~R3	G7~G2	B7~B3

15.17.2. Icon 和 Gif 的 dataFormat 选择说明

dataFormat 的选择关系到 UI_Editor-Lite 软件编译工程产生 UartTFT-II_Flash.bin 时，对应图片的数据转换。

若控件的 dataFormat 参数留空，则控件关联的图片的数据转换跟随工程的图片格式（RGB Fomat）,原图为 bmp 和 jpg 格式其数据转换成 RGB565 或者 RGB888 格式，原图为 png 格式其数据转换成αRGB4444 格式。

若控件的 dataFormat 参数需要选择，则需要遵守以下规则：

1、**原图为 png 图：dataFormat 不能选择 RGB565 或 RGB888 及其压缩格式。**

2、**原图为 bmp 或 jpg 图：不能选择αRGB8565，Softpng 和αRGB4444 及其压缩格式。**

若多个 Icon 或 Gif 选择导入同样编号的素材且 dataFormat 需要选择不同的格式，则需要将这些素材复制出多份且重命名为不同的编号。这样做的原因是，软件在生成 UartTFT-II_Flash.bin 时，所有被调用到的素材都只生成一份数据，同一个编号的素材以最后一次转换的数据为准，在此之前转换的数据会被覆盖。

16. 附录

16.1. 附录 1 LT165 烧录说明

16.1.1.TFT 串口屏的升级更新概述

一个新的 165 芯片，需要依次烧录 **Bootloader.bin、MCU_Code.bin** 和 **UartTFT-II_Flash.bin** 三个文件，才能正常工作。

以 LT165A 芯片为例（其串口屏演示板（Demo Kit）见附录 3），烧录后 Bootloader.bin 和 MCU_Code.bin 文件会下载到 LT165A 芯片，UartTFT-II_Flash.bin 文件会下载到 SPI_Flash 芯片。

注：

1、**bootloader 程序：用于下载 MCU_Code.bin 和 UartTFT-II_Flash.bin 文件。**一般来说，只有需要更换下载方式的时候才会更换 bootloader 程序。

2、**MCU_Code.bin 文件或程序：UI_Editor-Lite 控件实现的基础支持程序**，用户可在此程序内编程，自定义一些基础程序不支持的操作。编译 MCU_Code 的程序产生，烧录至单片机的程序空间。

3、**UartTFT-II_Flash.bin 文件：用户素材和 UI_Editor-Lite 控件参数结合在一起的文件**，里面包含了工程用到的素材以及用户在 UI_Editor-Lite 上编辑的内容。通过 UI_Editor-Lite 编译 UI 工程产生，文件通常较大，几十至上百 MB 不等（看素材的大小），烧录至外部 SPI Flash 内。

乐升半导体每个串口屏控制芯片所支持的升级更新方式不同，请参考下表。用户在可选择合适的方式给对应的芯片升级 MCU_Code.bin 和 UartTFT-II_Flash.bin。

表 16-1：各芯片升级方式

型 号	TFT 接口	升级 Bin 文件	Uart 串口升级 (LT_Uart_GUI)	USB 接口升级	SD 卡升级
LT165	8Bit MCU	MCU_Code	V	-	-
		UartTFT-II_Flash	V	-	-

注：‘ -’ 表示不支持该功能，‘ V’ 表示支持。

16.1.2.LT165 的烧录说明

16.1.2.1.LT165 的 Bootloader 程序说明

注：该步骤为非必要步骤，更改烧录方式或者烧录程序出问题才需要下载 Bootloader 程序 。

LT165 有 1 个 bootloader 程序，名字为：

LT165_BSP_Bootloader_Vx.x：支持串口更新 MCU_Code.bin 和 UartTFT-II_Flash.bin 文件。

具体更新说明可在后续章节查看。

Bootloader 程序是烧录的前提条件。**对于从未烧录过程序的芯片第一次下载 Bootloader 需要用 LT_Programmer_Vx.xx 软件或者乐升的脱机烧录工具协助下载，而对于已经下载过 Bootloader 程序的芯片可以忽略以下内容，直接使用 LT_Uart_GUI_Vx.xx 软件进行 mcu 程序和 UI bin 文件的下载。**

1. 先到乐升官网 www.levetop.cn 的下载专区下载“**Bootloader 烧录软件**(LT_Programmer_Vxx.exe)”（V1.08 及以上版本），及进行解压缩，下载路径为：乐升官网→下载专区→开发软件/教学视频→串口屏开发软件→MCU 程序/Flash 数据更新软件。

2. 烧录板连接到电脑，执行“LT_Programmer_Vxx.exe”软件后点击“Input Files”导入要烧录的 bin 文件；

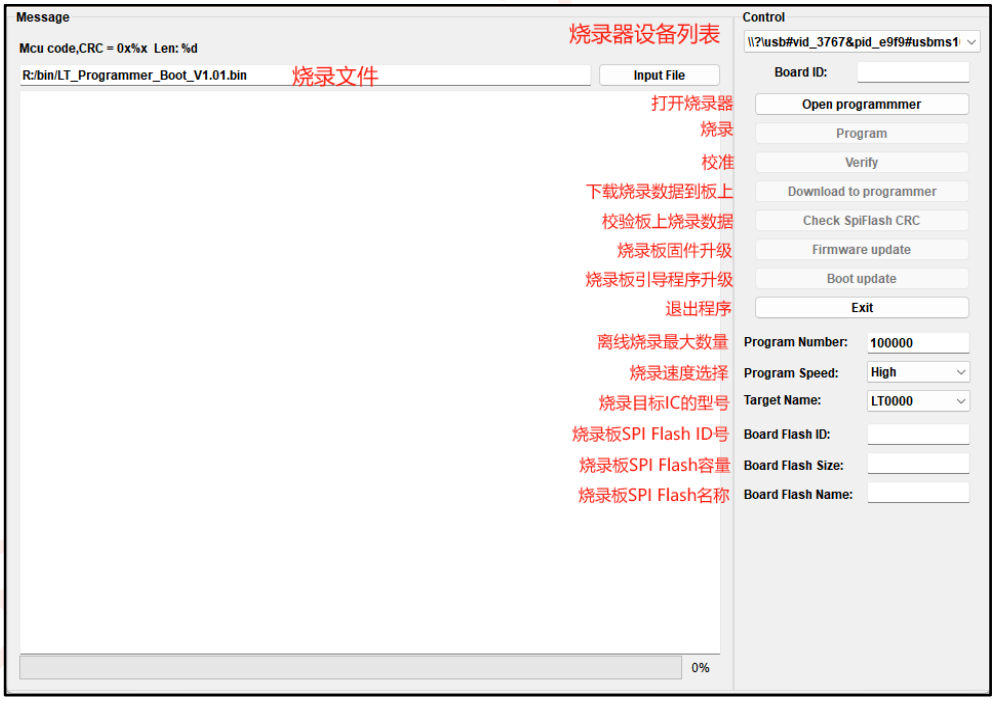


图 16-1：导入要烧录的 Bootloder 文件

3. 点击 Open programmer 连接烧录板；

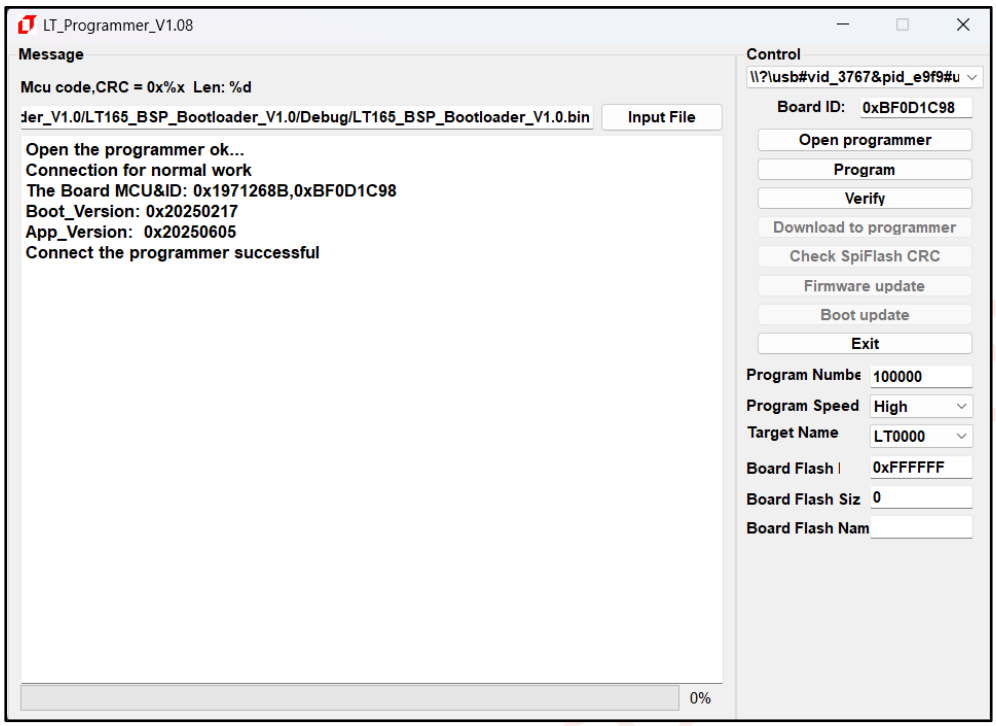


图 16-2：连接烧录板

4. 点击“Program ...”进行烧录，烧录软件会自动识别 IC，**不过要确保导入的烧录文件时对应 IC 的**，烧录成功如下图所示：

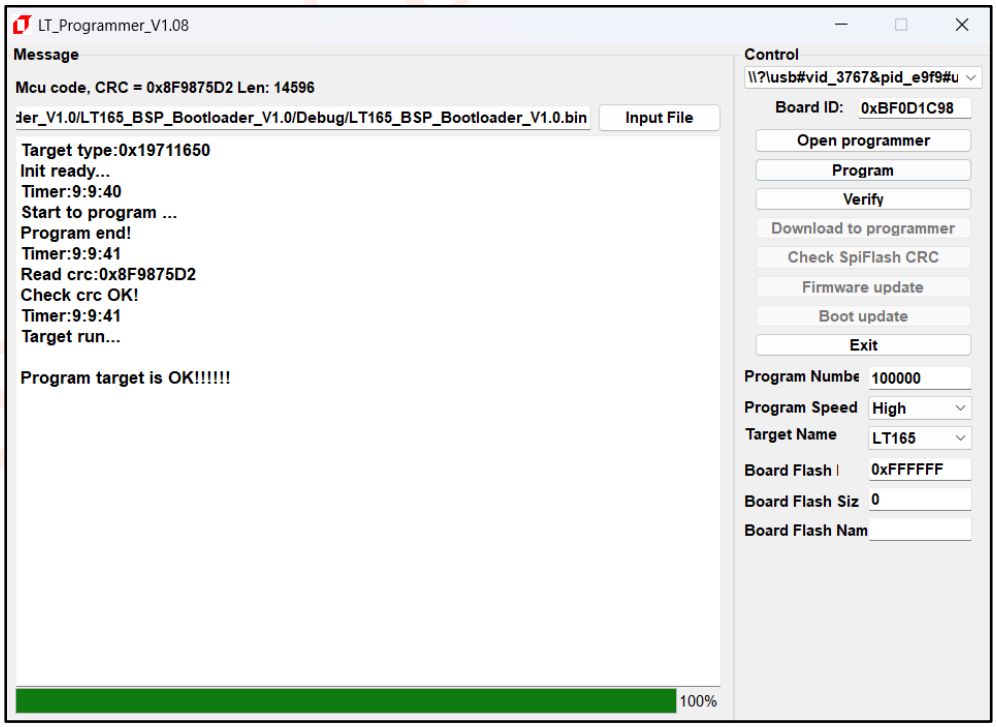


图 16-3：进行烧录与烧录完成

16.1.2.2. 下载更新 LT165 的 MCU_Code.bin 和 UartTFT_Flash.bin 文件

165 系列更新 MCU_Code 文件和 UI 文件只能通过串口进行更新，具体操作步骤如下：

1、接线说明，如下图所示，进行烧录前，**先将标注 1 处的 TXD1(busy)引脚和 GND 相连，然后通过标注 2 出 USB 口供电。待开发板进入烧录模式后，再通过标注 1 处 TX(TXD1)与 USB 转 TTL 模块的 RX 相连，RX(RXD1)与 USB 转 TTL 模块的 TX 相连，GND 与 USB 转 TTL 模块的 GND 相连，USB 转 TTL 模块的 USB 端与电脑连接。**

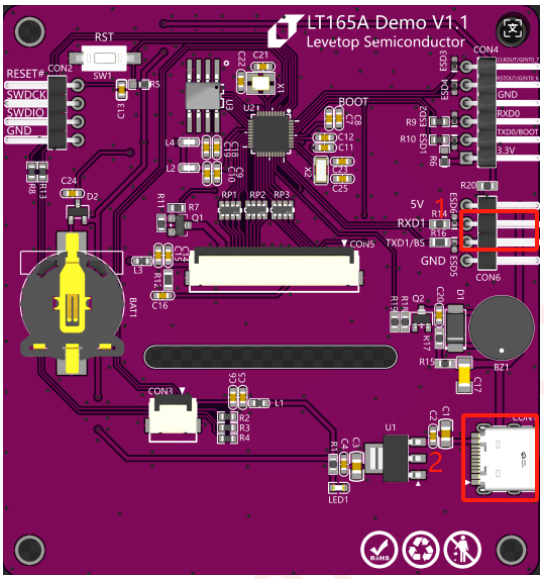


图 16-4：LT165 开发板示意图

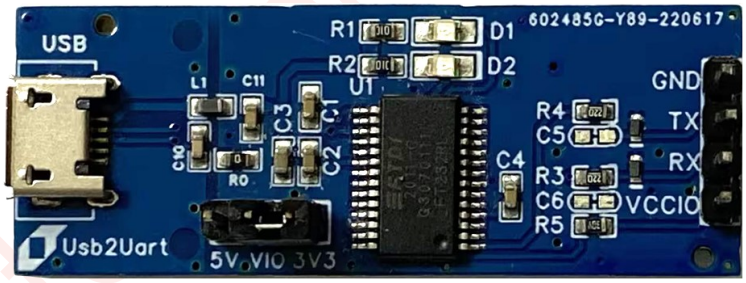


图 16-5：USB 转 TTL 模块

2、打开 LT_Uart_GUI_Vx.xx（3.48 以上版本）软件，如下图所示。

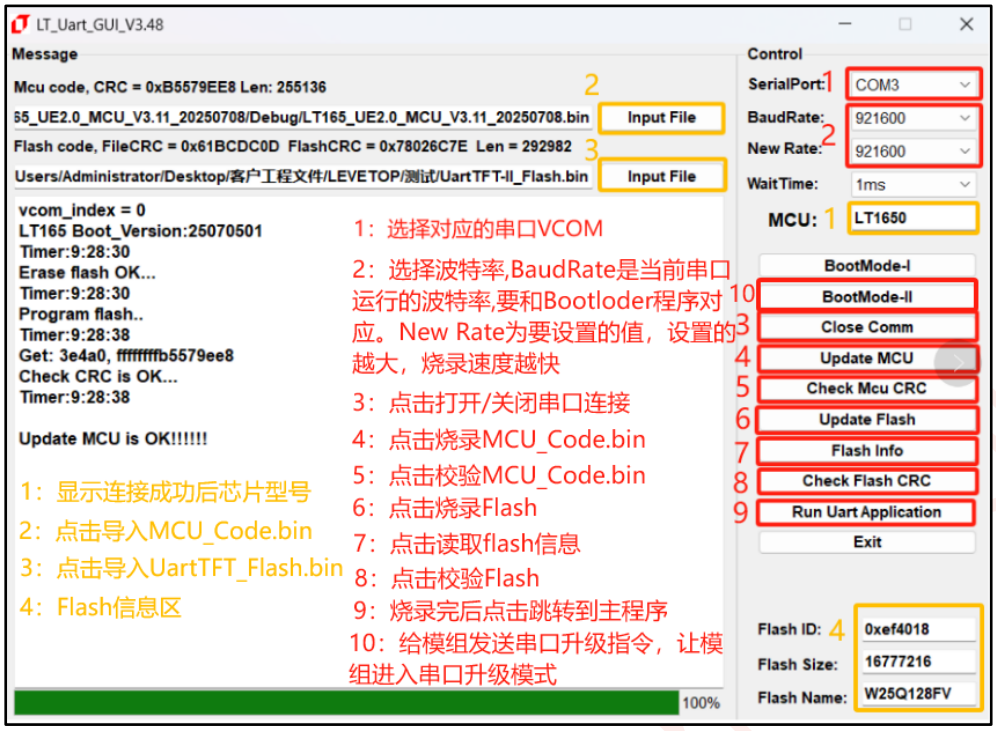


图 16-6：串口升级软件界面说明

3、打开 LT_Uart_GUI_Vxxx 软件后，检查 COM 口和 BaudRate 是否对应，点击 Open Comm。然后选择对应的端口，点击 Input File 添加需要烧录的 MCU_Code.bin 和 UartTFT_Flash.bin 工程文件，**需确保 BaudRate 为 115200**，再点击 Open Comm 打开端口连接。BaudRate 若不是 115200 则无法连接，建立连接后软件自动将下载速度调整至 921600 。然后点击软件右侧 Update MCU 或者 Update Flash 进行更新，如图 16-7 所示。更新完成后，如图 16-8 所示，最后点击 Run Uart Application 可跳转到主程序。

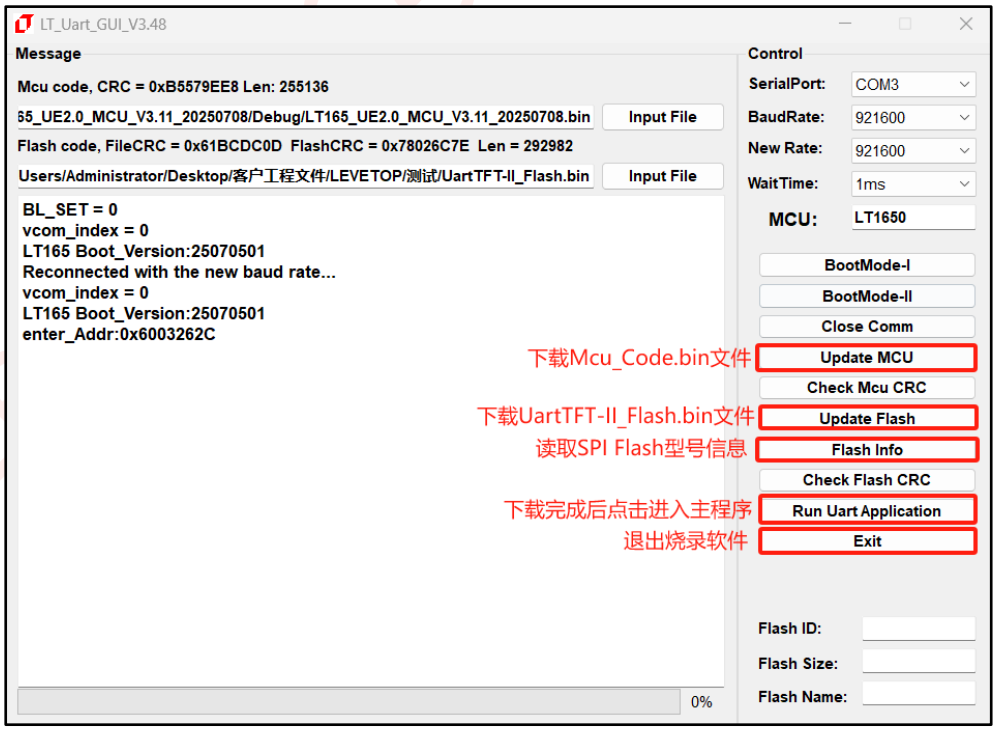


图 16-7：串口升级软件界面说明

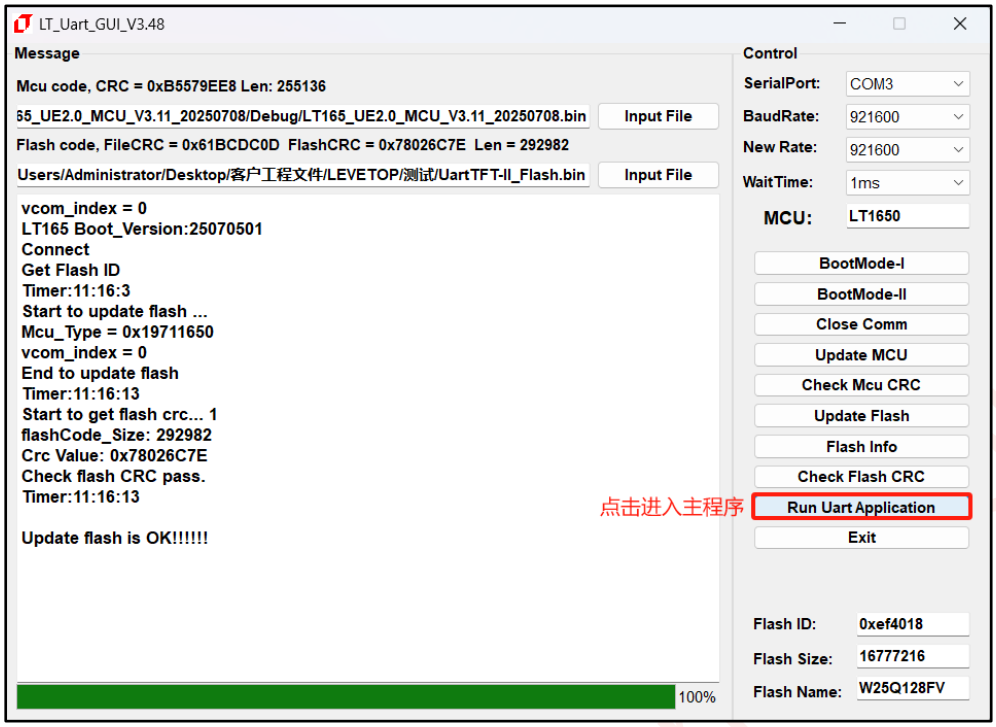


图 16-8：点击进入主程序

4、对于更新 Flash code 失败的情况，可以点击 Flash Info 后查看 Flash 信息区是否能显示全开发板上 flash 的信息 (包括 Flash ID、Flash Size、Flash Name)，如果无法显示完全部的信息，就需要在烧录软件的 Flash.ini 中加入 Flash ID，如果可以显示全但仍无法烧录则需要排查硬件问题。

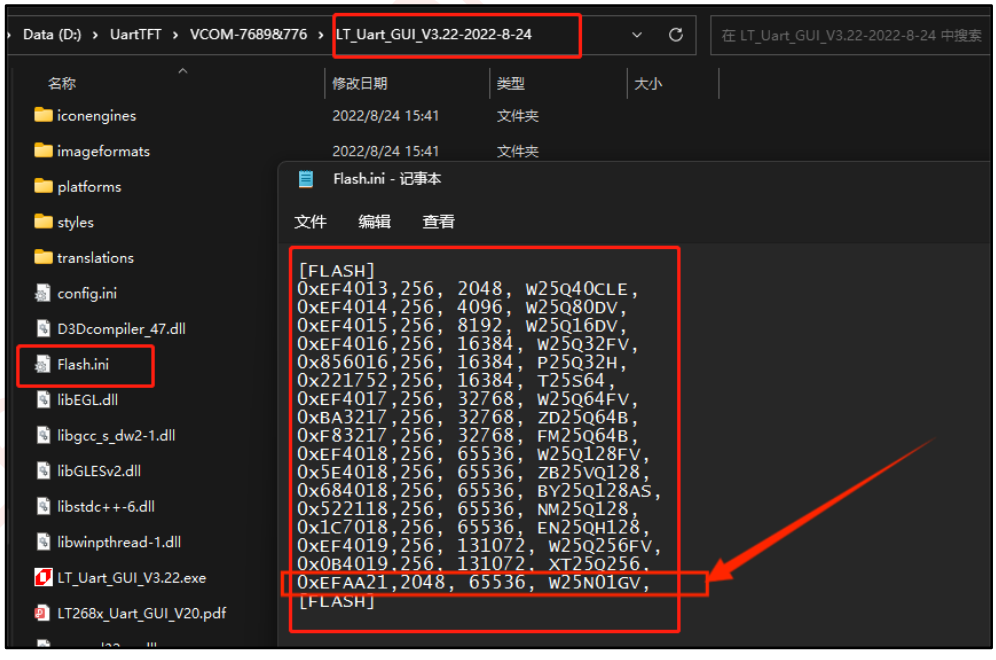


图 16-9：添加 Flash ID

以上图箭头所指的 Flash 信息为例，W25N01GV 是该 Flash 芯片的名字，0xEFAA21 是该 Flash 芯片的 ID，65536 为 Flash 芯片内可编程页的数量，2048 为每个可编程页可储存的数据量(单位 byte)。

16.2. 附录 2 UI_Editor-Lite 支持的 IC 及其参数限制

16.2.1.UI_Editor-Lite 支持的 IC 列举

IC 切换的位置如下图所示，通过设置界面的 IC Type 更改 LT165 系列 IC。

图 16-10：IC 选择

16.2.2.不同 IC 的参数限制

在使用 UI_Editor-Lite 基于不同 IC 设计工程时，会存在不同的限制，LT165 系列 IC 限制如下表所示，宽高相关参数的单位均为像素。

表 16-2：IC 类型及其参数限制表

参数 \ IC	LT165-MCU
用户地址范围	0x0000~0x200
显示 PNG 图片大小限制	W<= 768(Pixel)
环形触控/进度条素材	支持通过 Icon 的方式制作环形触控和进度条
模拟时钟素材	不支持模拟时钟
曲线控件面积大小限制	不支持曲线控件
RGB 格式支持	RGB565
键盘页素材大小限制	W<= 768(Pixel)
带滑动效果的滑动翻页	不支持

参数 \ IC	LT165-MCU
弹窗背景 降低亮度	不支持
页面底图 压缩功能	不支持
Icon 和 Gif 压缩功能	支持 RGB565_ZIP
编码器功能	支持

注：

1、W <= 768，W 指控件或者图片的宽度的分辨率。**该不等式表示 W 的像素小于 768 即可。**

2、 “-”表示没有该功能。

3、**LT165-MCU** 指 LT165 搭配 MCU 接口屏。

16.2.3.不同 IC 的单页控件数量限制

在使用 UI_Editor-Lite 设计工程时，我们会碰到在同一页内放置多个同类控件的情况，单页内不同控键可放置的数量有限制。如果需要突破该限制，则需要修改支持程序。各 IC 标准版支持程序（Mcu_Code）的单页控件数量限制如下表所示，仅供参考。

表 16-3：串口屏单页相应控件最大支持数量表

IC 型号 / 控件名称	上位机	LT165
按键（Button）	128	10
滑动菜单（Slidemenu）	64	-
弹窗（Popupbox）	64	3
变量调节（Variable Button）	256	4
多变量操作（Multi-Variable Button）	128	4
环形滑条（Circular touch）	32	4
滑条（Slider Bar）	32	4
键盘键值（SingleKey）	200	30
数字键盘（Numeric Keypad）	128	1
英文键盘（EN_KeyBooard）	128	-
中文键盘（CN_KeyBoard）	128	-
字符串显示（String_Label）	512	20
静态文本显示（Static_Text）	512	20
文字滚动（Text Scroll）	32	3
字库数字（Text Number Display）	512	30
图片数字（Graphics Number Display）	32	30
模拟时钟（Analog Clock）	8	-
数字时钟（Digital Clock）	16	4
动图（Gif）	32	4
二维码（QRCode）	16	-
音频（Audio Play）	32	1
条形进度条（Progress Bar）	32	4
环形进度条（Circular Progress Bar）	32	4
位元状态（Bit Status）	128	25
小图标（Icon）	128	25
曲线（Trend Graph）	12	-
编码器（Encoder）	8	1
计时器（Timer）	8	4
自动变量（Automatic variable）	64	4
指针（needle）	16	-

注：上表填了‘-’的格子说明其对应 IC 不支持对应控件。

16.2.4.不同 IC 的变量地址范围及寄存器地址说明

表 16-4：不同 IC 的变量地址范围及寄存器地址说明表

IC / 寄存器	LT165
用户地址 (writeAddr) 范围	0x0000~0x0200
换页寄存器	0x7000
背光寄存器	0x7001
时间寄存器	0x7002~0x7007
时间确定寄存器	0x7008
Wav 控制寄存器	0x700A
音量调节寄存器	0x700B
电阻屏校准寄存器	0x700C
键码触发寄存器	0x700D
自动背光控制寄存器	0x700E
休眠背光亮度控制寄存器	0x700F
背光休眠时间控制寄存器	0x7010
串口升级寄存器	0x7011
多国语言寄存器	0x703F

16.3. 附录 3 TFT 串口屏的演示板（Demo Kit）

乐升半导体 针对每个串口屏控制芯片都提供演示板（Demo Kit），如下图是乐升半导体的 LT165ATFT 串口屏演示板，可以搭配标准 40Pin 或是 50Pin TFT 屏。

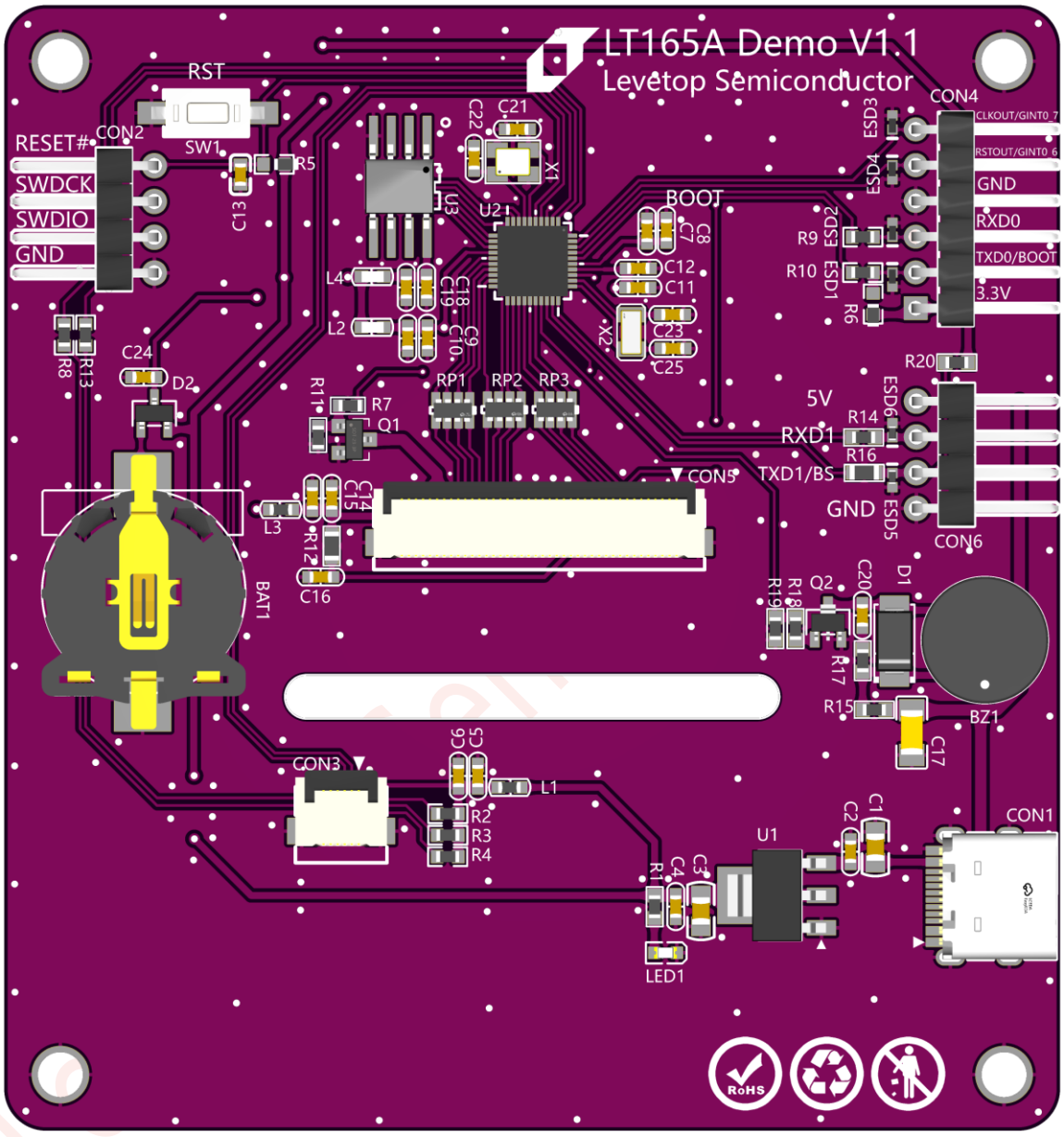


图 16-11：LT165A 串口屏的开发演示板（Demo Kit）

乐升半导体提供的演示板内都已经含有开发程序，串口屏主板上的 SPI Flash 也烧录有演示图片、Gif 动画、字库、Wav 音乐文件等所组合成的 Bin 文件，相关 Demo 文件可以前往深圳市乐升半导体有限公司 (levetop.cn) 官网的下载专区内下载，或与业务人员联系。

16.4. 附录 4 串口屏开发流程

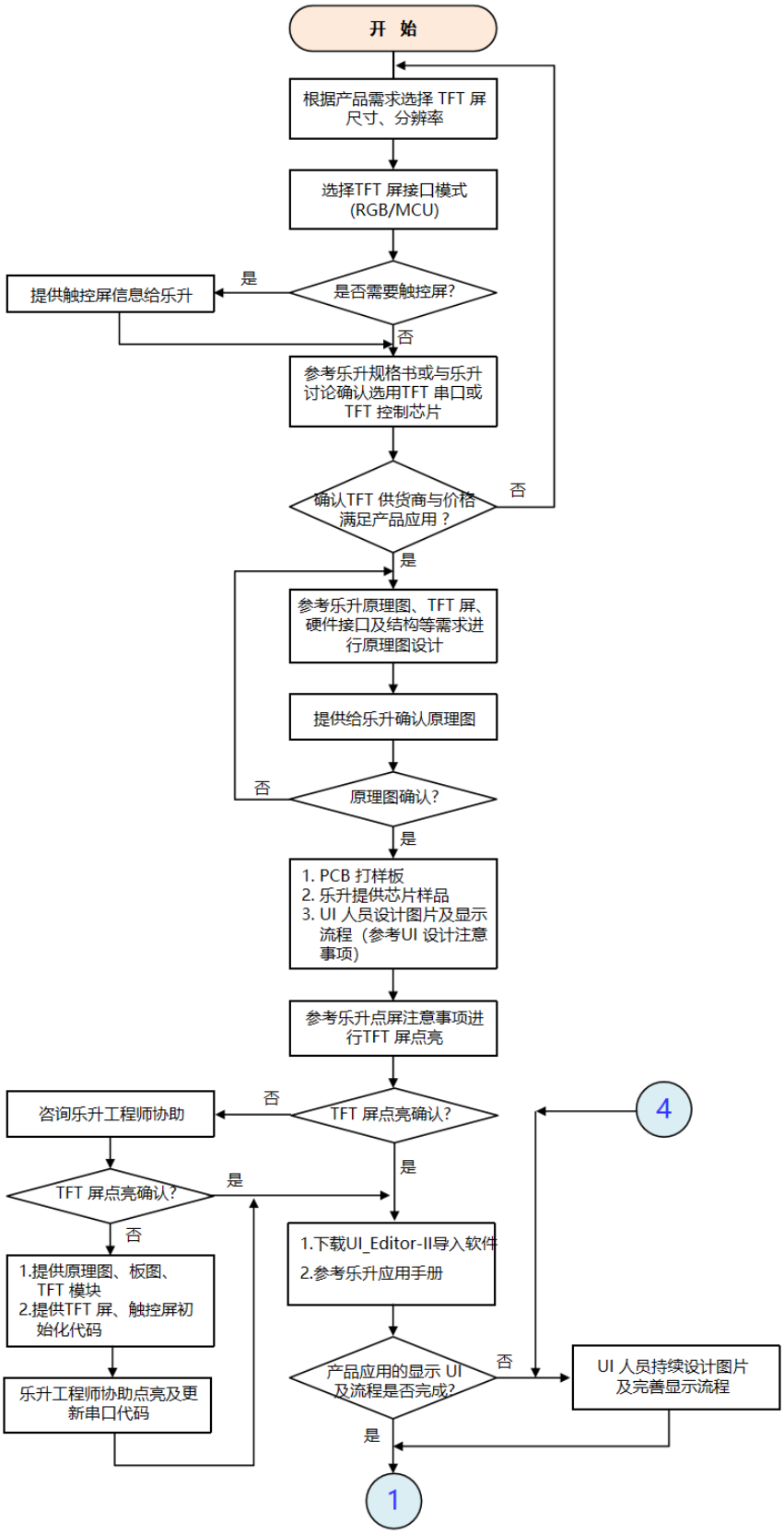


图 16-12：开发流程（1）

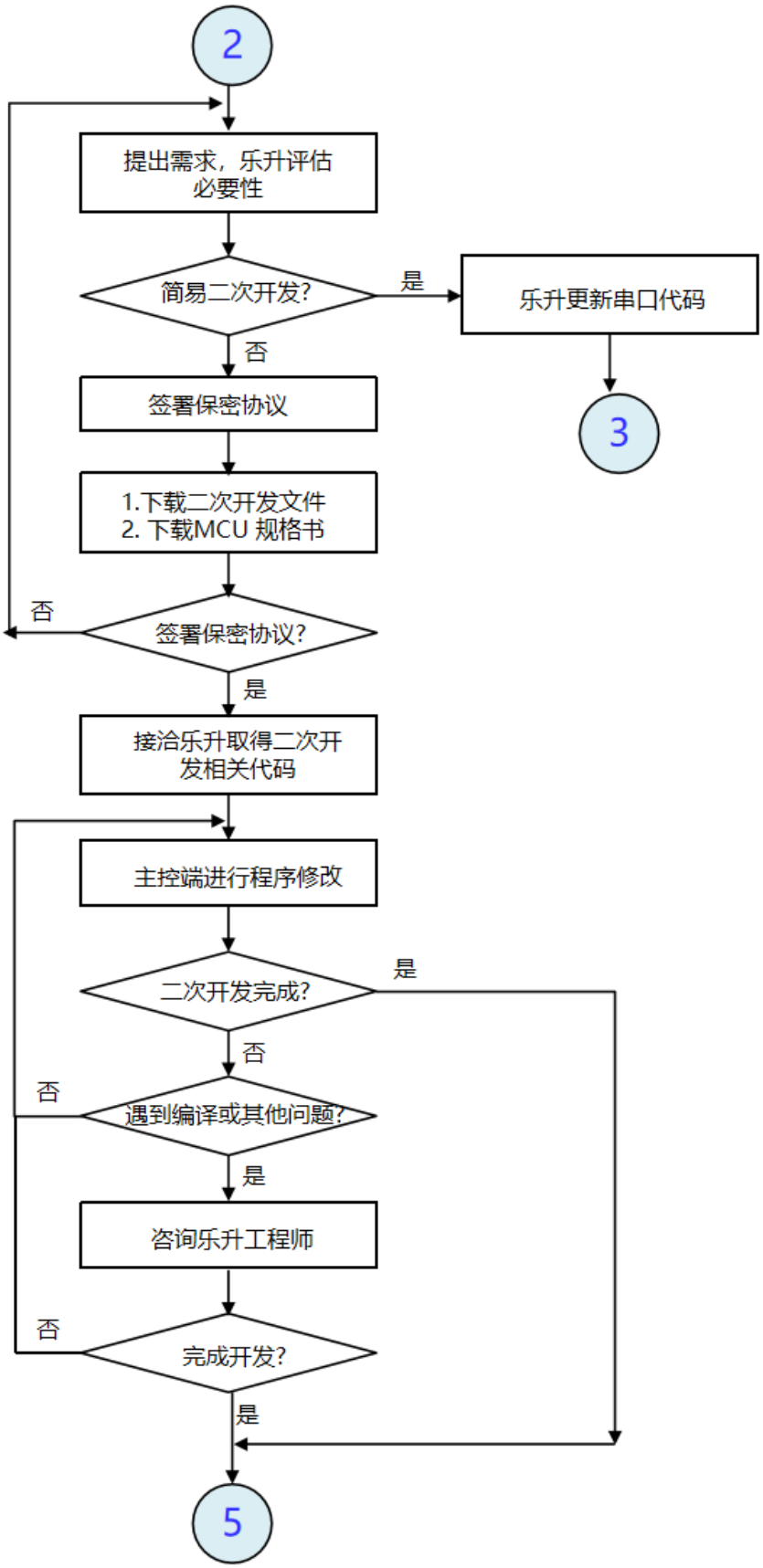


图 16-13：开发流程（2）

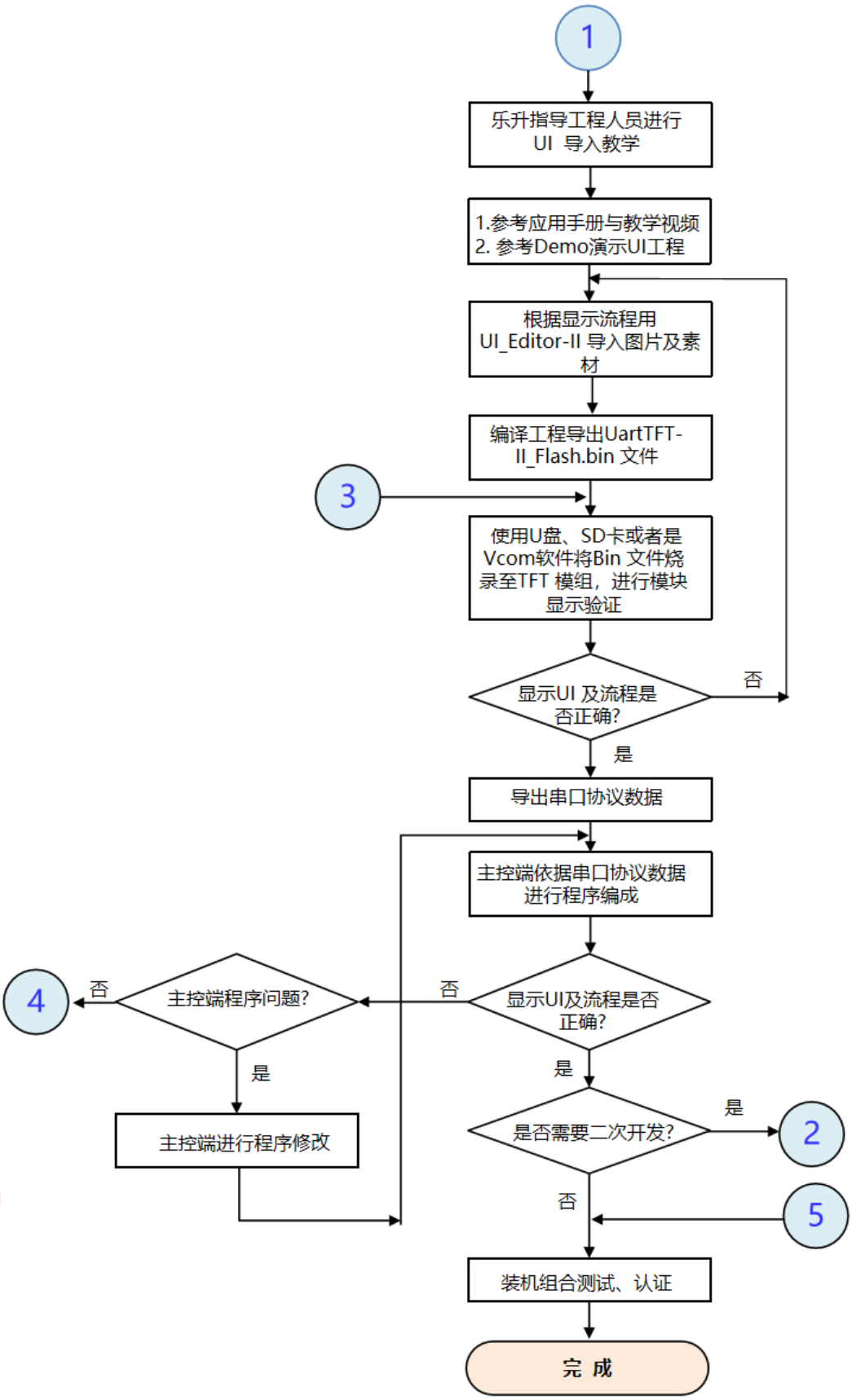


图 16-14：开发流程（3）

16.5. 附录 5 一代与二代串口通信的差异

第一代和第二代的串口通信有所差异的，第一代的串口通信是采用指令模式，所有动作都赋予定义好的固定串口指令，而第二代串口通信采用变量控制模式，变量变量来控制所要显示的画面，第二代串口屏 UI_Editor-Lite 与第一代的串口通信与指令差异如下：

表 16-5：串口通信与指令差异

模式	UI_Editor	UI_Editor-Lite
串口通信	由“帧头 + 串口指令 + CRC + 帧尾”构成，帧尾可修改。	由“帧头 + 长度 + 读/写指令码 + 变量数据+ CRC”构成，帧头可修改，支持 Modbus，I2C。
串口指令	功能与串口指令对应，单个功能收指令序号限制，最多为 256 条。	串口指令只进行变量读写操作，串口与功能不一一对应，单个功能数目不受限制。

16.6. 附录 6 二代串口屏可实现效果

表 16-6：第二代串口屏可实现的效果

No.	功 能	说 明
1	无控件触发弹窗警告	使用指令：弹窗指令 + 键码触发寄存器。
2	数字输入	可使用数字键盘，在输入执行后直接显示在用户的 UI 输入框内。
3	背光控制功能	可滑动设计好的进度条/圆环进度条 UI 实时控制背光。 可由主控发出指令增量调节控制背光。 可使用弹窗弹出 UI 窗口来进行背光控制；使用指令：弹窗指令 + 进度条/圆环进度条/增量 + 多变量控制。
4	弹窗选择功能	使用指令：弹窗指令 + 多变量控制。 使用弹窗将需要选择的功能平铺展开，点击对应的图案后，返回主界面可将选到的 Icon 在主界面显示。 使用弹窗指令，弹出菜单选择，通过滑动去选择。
5	日期/时间调整	如同弹窗选择功能，可滑动日期/时间设置的弹窗去设置日期与时间。
6	进度条关联数据	进度条关联数据显示，可显示百分比、温度等变量数据及代表符号。
7	Icon 变化	变量调节控制 Icon 循环切换。

16.7. 附录 7 OTA 差分升级说明

16.7.1.概述

为了改善用串口进行 OTA 升级时升级速度慢的问题，乐升半导体设计出了差分升级的升级方式。这种升级方式，会在编译的时候给 UI 工程的每一类素材划定存储范围，并且在 UI 工程的设置界面用户可以根据需求在已有素材划定的存储范围的基础上，给每一类素材设定预留的升级空间。这样在开发后期如需要更改、删减、替换 UI 工程的素材则只要重新工程生成 bin 文件。这时将 UI 工程的旧的 bin 文件和 UI 工程新生成的 bin 文件导入乐升半导体研发的差分对比软件，该软件就可以对比分析出新、旧两个 bin 文件的差异，并将差异部分的数据生成以.diff 为后缀的文件。由于此文件只有两个 bin 文件差异部分的数据，所以数据量很小，得到的 diff 文件可以通过 USB 或 uart 用乐升半导体研发的软件 LT_Uart_GUI_VX.XX(注意需要 3.48 以上版本)将.diff 后缀的文件更新到 SPI flash 中，**也可以通过 WIFI 蓝牙或由客户主板 MCU 通过串口更新到 SPI Flash 中来完成成 UI 工程的升级。**

16.7.2.OTA 差分升级流程说明

1. 使用 UI-Editor 生成工程更改前和工程更改后的 bin 文件。
2. 使用乐升半导体研发的 Diff_OTA_Tool_VX.XX 软件对比新、旧两个 bin 文件的差异，并生成差异部分的数据文件。
3. 使用乐升半导体研发的 LT_Uart_GUI_VX.XX(注意需要 3.48 以上版本)将.diff 后缀的文件更新到外挂 flash 中即可完成 UI 工程的升级。

16.7.3.软件使用说明

16.7.3.1.UI-Editor 差分升级部分使用说明

1. 如果需要 OTA 差分升级的 UI 工程，要先在 UI-Editor 中设置 OTA 升级地址管理，如下图所示：

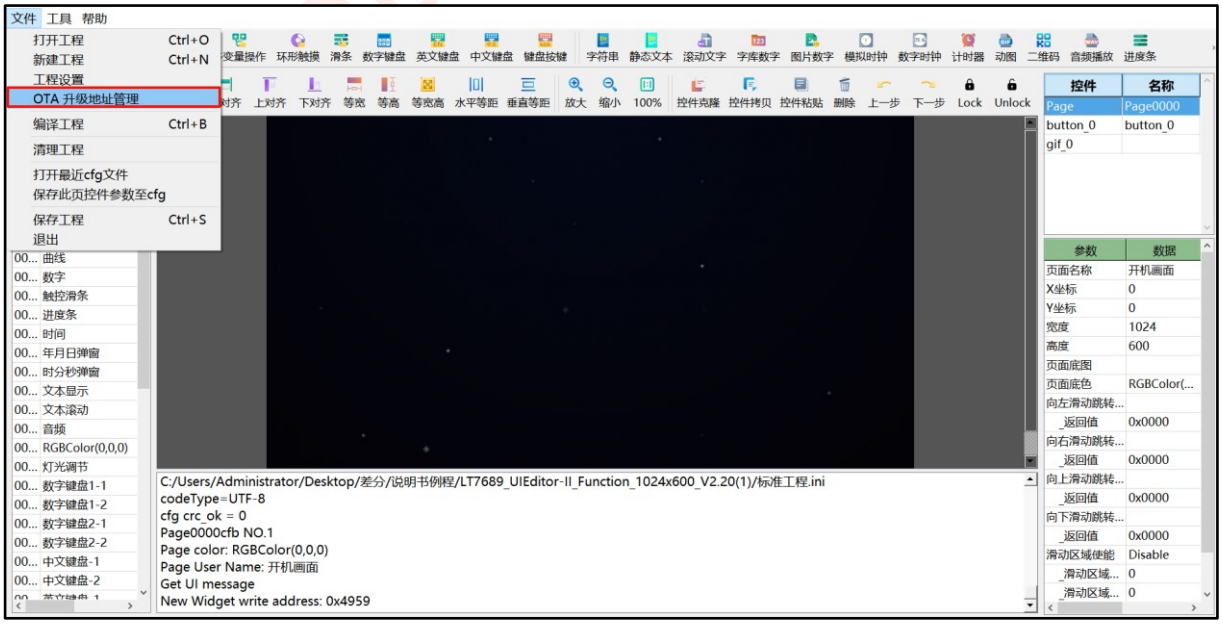


图 16-15：在 UI 工程找到 OTA 升级地址管理

2. 打开 OTA 升级地址管理，并勾选左上角 OTA Update Addr，如下图所示：第一列是每种素材数据的开始地址；第二列是该类素材实际的数据长度(单位 byte)；第三列是该类素材所要预留的空间，方便后续添加同类素材；第一列和第二列由程序自动生成，第三列需要人工设置，默认是 1024K。这个地址设置最好在工程差不多完成时再设置，因为这时架构大致稳定，后续添加素材不多，预留的空间比较好设置了。

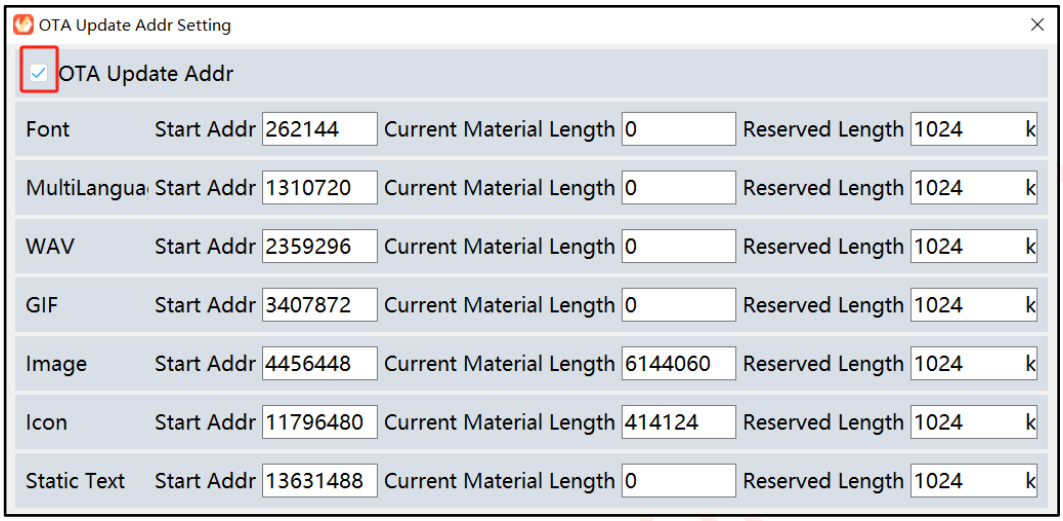


图 16-16：勾选 OTA Update Addr

3. 编译之后就可以根据后期需要升级的素材大小给每一部分控件的素材预留出升级空间(初始每部分素材会自动预留 1024k 的升级空间，可手动根据需求更改）。需要注意的是预留的空间必须是 128k 的倍数，如果输入的数值不是 128k 的倍数，UI-Editor 会自动向上取 128k 倍数的空间（如果在工程设置中勾选了 GBK 编码添加则会自动向上取 256k 倍数的空间）。设定好升级空间之后需再次进行编译，此时编译生成的 bin 文件作为旧的 bin 文件烧录到 SPI flash 中。设定升级空间的位置，如下图所示：

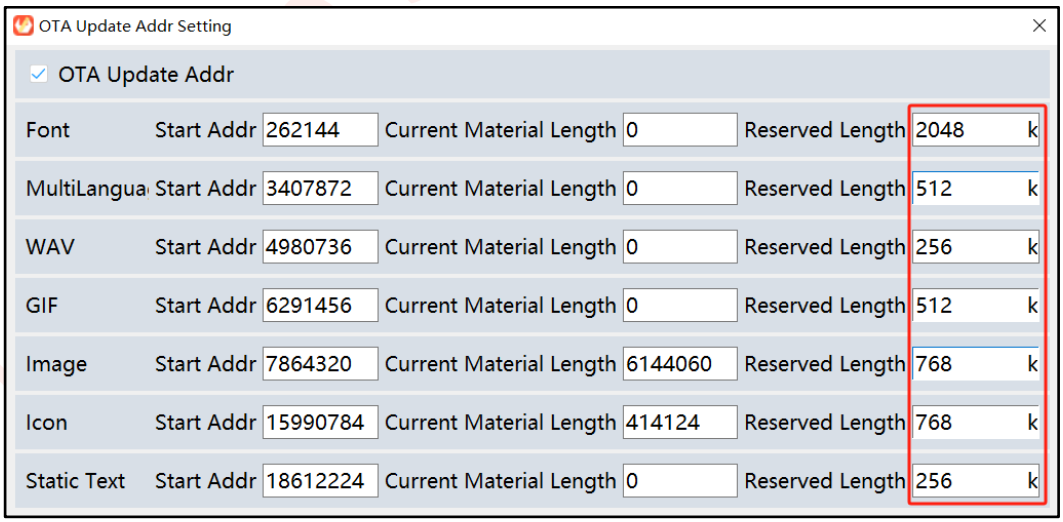


图 16-17：设置升级预留空间

4. 根据需求对 UI 工程进行修改，此时对 UI 工程的变化可能有，控件的位置改变、控件的地址改变、控件的数量改变、素材的增加、减少以及替换等。需要说明的是对于控件的位置、地址、数量等信息全部是保留在生成 bin 文件的前 256k 数据之中，如果控件的位置、地址、数量等信息改变则前面 256k 数据就会发生改变并被记

录在差分对比后的.diff 文件中。而素材的增加减少则是会保存在之前步骤设定的升级预留空间中。注意素材的增加不能超出设定的升级预留空间的大小且增加的素材的编号必须往后增加不能在前面或者中间插入，否则会出现错误。修改好 UI 工程后再次编译工程作为差分升级的新工程（注意 OTA 升级地址管理左上角的 OTA Update Addr 要勾选上）。

5. 需要注意的是 UI 工程两次生成 bin 文件即前面提到的旧、新两个 bin 文件大小一定是一样的（因为旧的 bin 文件虽然没有新添加的素材但是预留了升级空间，而新的 bin 文件新增的素材是放置在预留的空间之中的，bin 文件的大小指的是所占空间的大小，所以新、旧两个 bin 文件总的文件大小是一样的），可以以此来作为新增素材有没有超出预留空间的判断。

16.7.3.2. 差分对比软件 Diff_OTA_Tool 使用说明

1. 差分文件的数据结构如下图所示：

A. 总结构

32bytes 结构头
Diff数据 Block0
Diff数据 Block1
Diff数据 Block2
Diff数据 Blockn

图 16-18：差分文件总结构

B. 结构头（里面的数据皆为高 8bits 在前）

32bytes结构头	byte0	0x44	字符“DOTA”是文件标识		
	byte1	0x4F			
	byte2	0x54			
	byte3	0x41			
	byte4		Diff Block数据的长度		数据高8bits在前
	byte5				
	byte6				
	byte7				
	byte8		Diff Block数据的CRC值		
	byte9				
	byte10				
	byte11				
	byte12		Diff Block的数量		
	byte13				
	byte14				
	byte15		升级原文件的长度		
	byte16				
	byte17				
	byte18				
	byte19				
	byte20		升级文件的CRC值		
	byte21				
	byte22				
	byte23				
	byte24		升级文件的UI数据的CRC值		
	byte25				
	byte26				
	byte27				
	byte28		文件头前28bytes的CRC值		
	byte29				
	byte30				
	byte31				

图 16-19：差分升级文件的 32byte 数据结构

C. Diff 数据 Block 的结构：16bytes 结构头高 8bits 在前，后续的数据按顺序读取即可

Block_0	byte0		Block数据的地址	
	byte1			
	byte2			
	byte3			
	byte4		Block数据的长度	
	byte5			
	byte6			
	byte7			
	byte8		Block数据的CRC值	
	byte9			
	byte10			
	byte11			
	byte12		Block数据的CRC值反码	
	byte13			
	byte14			
	byte15			
	DATA		Block数据	

图 16-20：每块 block 的 16byte 数据结构

2. 打开 Diff_OTA_Tool 软件，分别将 UI 工程生成的旧的 bin 文件以及新的 bin 文件导入 Diff_OTA_Tool 中，需注意的是，input new code 处要导入的是新的 bin 文件。而 input old code 处要导入的是旧的 bin 文件如图所示：

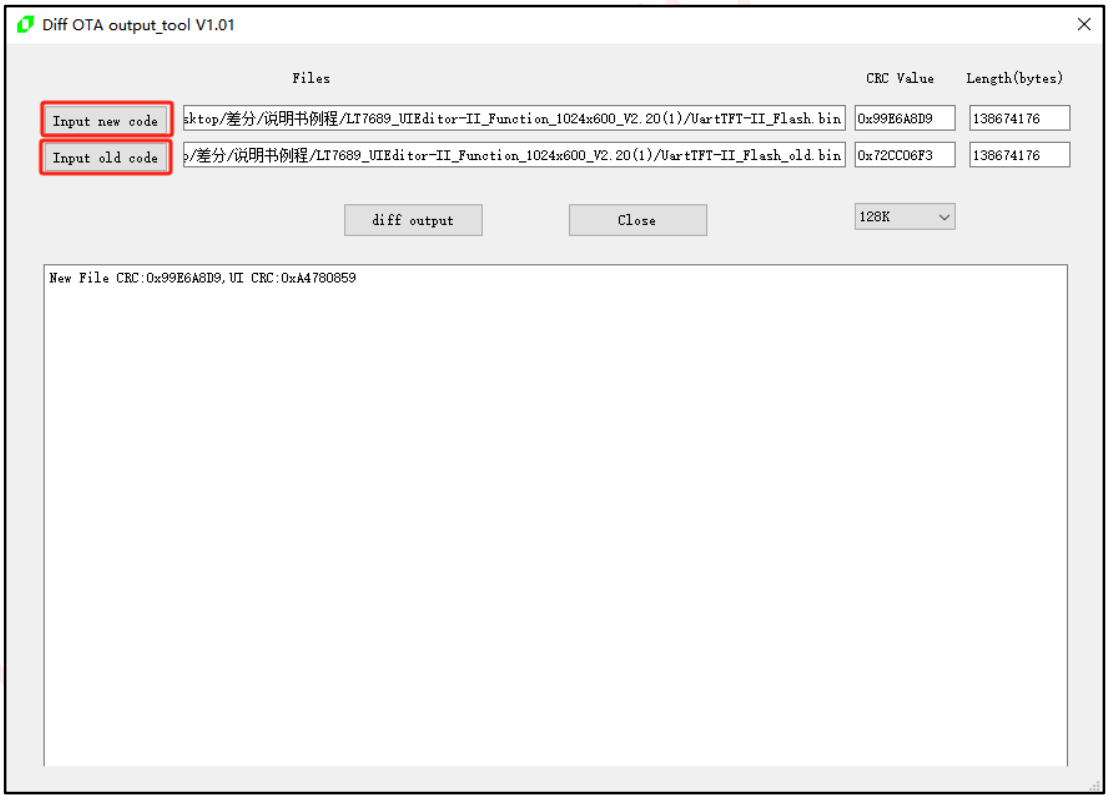


图 16-21：在差分对比软件中导入 bin 文件

3. 导入 bin 文件后，点击 diff output 即可对两个文件进行差分对比，如下图所示：

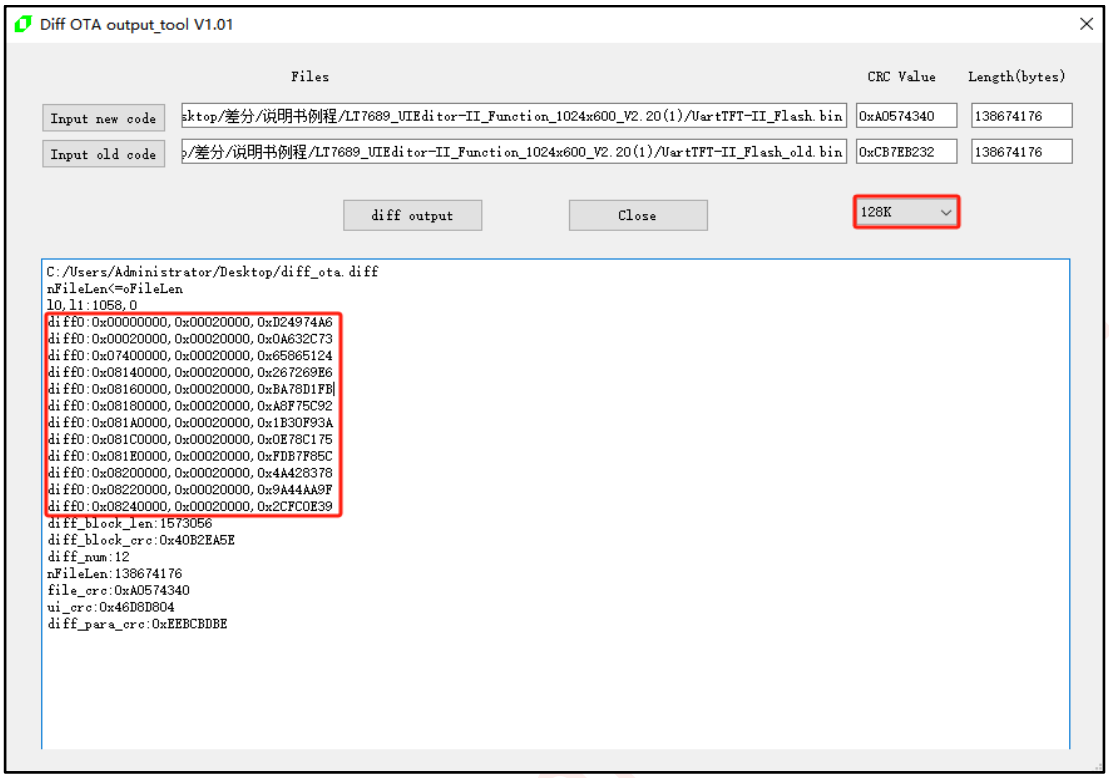


图 16-22：差分对比软件比较出的差异信息

需要注意的是上图 3-7 中右上角红框里面的 128k 意思是：将新、旧 bin 文件的数据从 0 地址开始每 128k 分成一包做一次差分对比，如果在新、旧文件的对比中这包数据有差异，则就会在图 3-7 左下角的红框中显示出来。对于 NandFlash 可以选择每 128k 数据作为一包进行一次对比，而对于 NorFlash 则可以选择 4k 或者 64k 数据作为一包进行一次对比。如上图所示的差异信息中比较出有多少行的数据不同实际就是新、旧 bin 文件有这么多包数据不同。

4. 点击 diff output 后会生成一个后缀名为.diff 的文件，指定文件的生成目录后，点击保存即可得到差分升级的数据文件（注意重复生成此文件时，最好改变文件名称，不要直接替换，否则文件可能叠加在一起）。如下图所示：

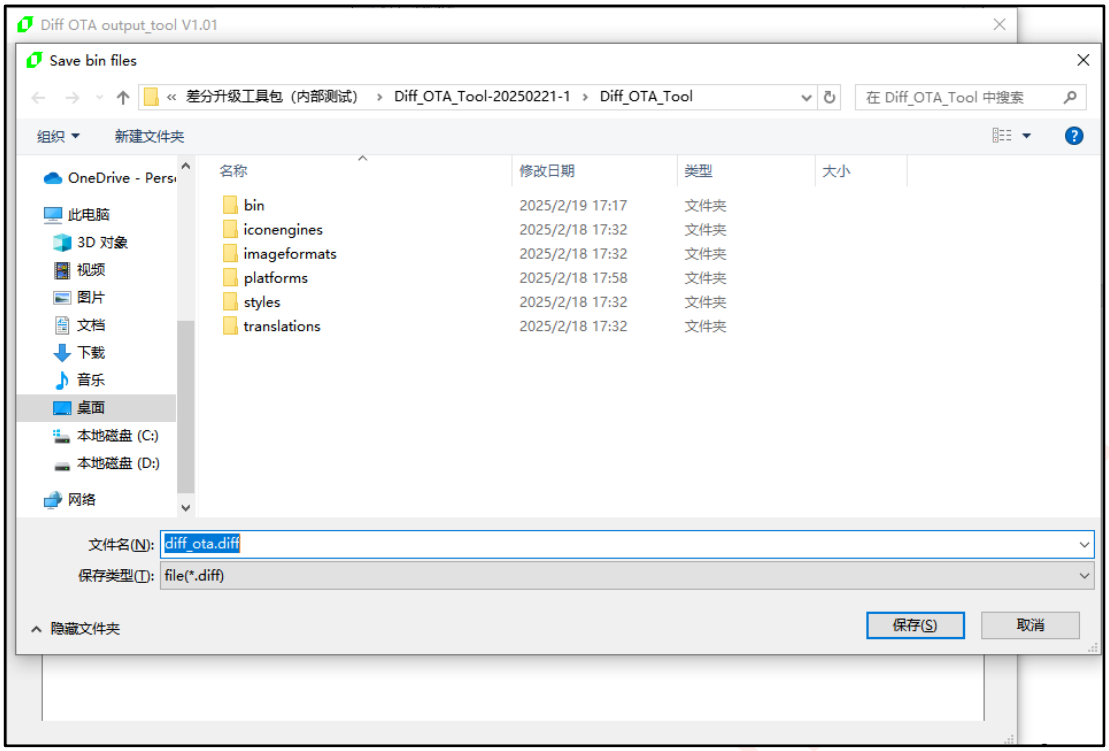


图 16-23：生成后缀名为.diff 的数据文件

16.7.3.3. 差分数据更新软件 LT_Uart_GUI 使用说明

经过上述步骤现在已经获取到了差分升级的数据文件，现在只需将此文件烧录到外挂 Flash 中即可完成 UI 升级。烧录后缀名为.diff 的差分升级数据文件需要用到 LT_Uart_GUI_VX.XX（注意需要 3.48 以上版本）这款软件（其实烧录差分升级数据文件和通过 USB 或串口烧录普通的 UartTFT-II_Flash.bin 流程一样）具体可查看 UI-Editor 的使用说明书附录 1 LT165 烧录说明，下文进行简要说明。

1. 参照附录 1 LT165 烧录说明让开发板进入烧录模式，并接好串口 1，如下图所示标号 1 处为串口 1，通过标号 2 处对开发板进行供电。

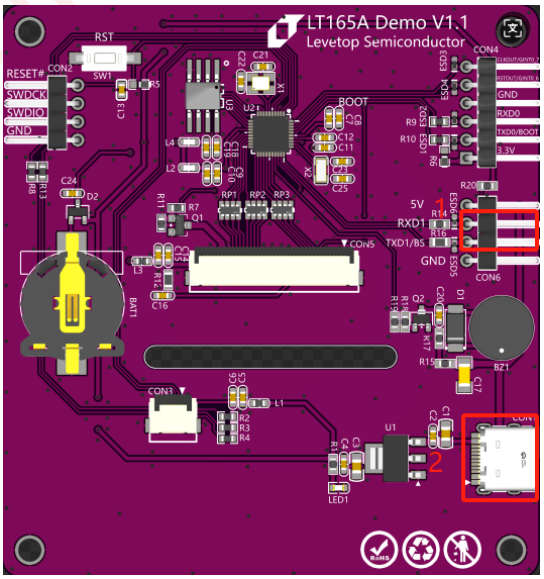


图 16-24：导入.diff 文件

2. 打开 LT_Uart_GUI_VX.XX 软件，点击 Input File 在 FlashCode 选项框中导入上述生成的.diff 文件如下图：

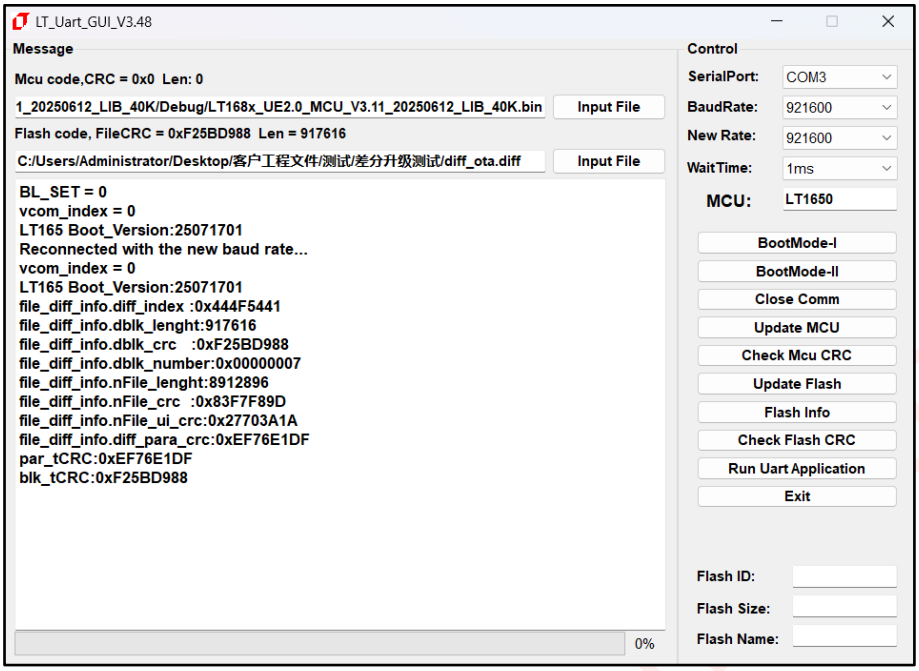


图 16-25：导入.diff 文件

3. 点击 Update Flash 进行烧录，烧录完成会有提示信息如下图所示：

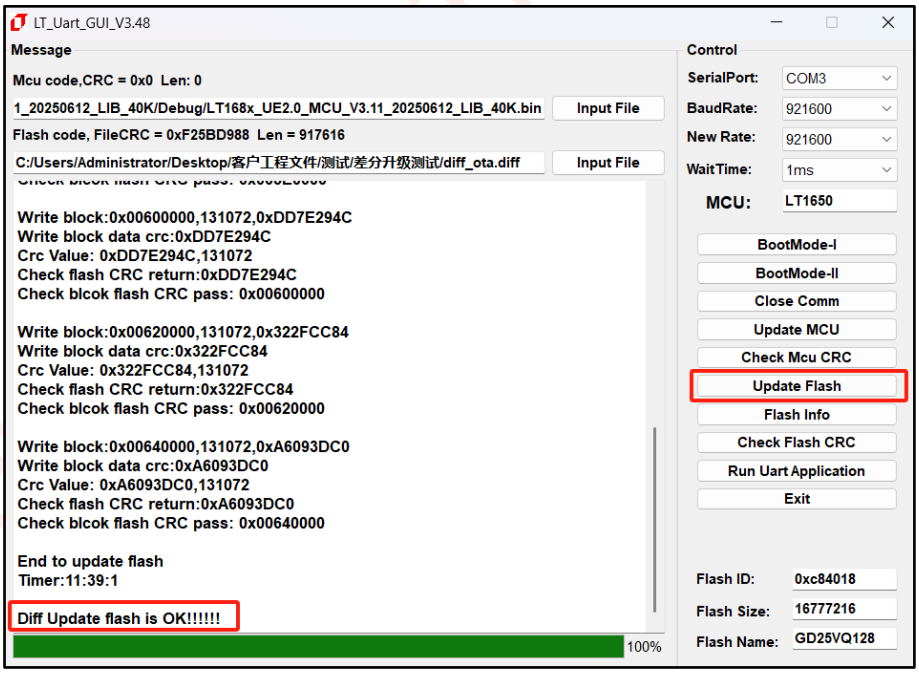


图 16-26：点击 Update Flash 进行烧录