



LT268B

TFT 串口屏方案

Serial Uart TFT Panel Solution

应用手册

Application Note

V1.1

www.levetop.cn

Levetop Semiconductor Co., Ltd.

目 录

1. 概述	1
1.1 LT268B 基本介绍	1
1.2 TFT 串口屏的软硬件架构.....	2
1.3 LT268B 串口屏原理图.....	5
2. 串口指令	6
2.1 串口屏指令集.....	6
2.2 主控端与 TFT 串口屏协议表	7
2.3 RS-232(UART) 通讯协议	10
2.4 主控端发送指令的范例.....	12
3. 图文 UI 编辑器 (UI_Editor.exe)	15
3.1 UI_Editor 界面的简介	15
3.1.1 使用 UI_Editor 的设计流程.....	20
3.1.2 新建工程和背景图的设定	21
3.1.3 显示图的设定.....	24
3.1.4 界面的编辑与调试	27
3.1.5 工程编译和 BIN 文件生成.....	30
3.2 显示内容的设定	32
3.2.1 单张及多张图片(80h)	32
3.2.2 卷动及循环卷动图片(D8h, D9h).....	33
3.2.3 循环显示重叠图片(81h)	35
3.2.4 显示 GIF 动画图片(88h).....	36
3.2.5 图片式文字的显示(80h)	37
3.2.6 图片式数字的显示(90h)	38
3.2.7 使用字库显示文字(C0h~C3h)	40
3.2.8 显示控件功能的图片(A0h).....	41
3.2.9 开机画面(9Ah)	43
3.2.10 进度条图(B0h).....	46
3.2.11 画直线(E0h)	50
3.2.12 基本几何绘图.....	51
3.2.13 表格制作(F6h)	55
3.3 其他功能	56

3.3.1	背光控制指令	56
3.3.2	起始控制指令	57
3.3.3	电阻屏控制指令	58
3.3.4	串口屏侦测指令	59
3.3.5	字库文件的说明	60
3.4	串口发送指令	61
3.4.1	SPI Flash 的结构	68
3.4.2	Userinfo.bin (128K) 更新	68
3.5	UI_Editor 范例下载	69
4.	图文整合编译器(UartTFT_Tool.exe)	71
4.1	编辑指令设定文件	73
4.1.1	信息设定 [INFO]	73
4.1.2	指令参数设定 [USERCMD]	74
4.2	指令参数的设定	76
4.2.1	固定图片设定及显示指令	76
4.2.2	动态图片设定及显示指令	78
4.2.3	控件图片设定及显示指令	81
4.2.4	图片式的数字显示指令	85
4.2.5	使用字库的文字设定及显示指令	88
4.2.6	画点指令	90
4.2.7	画圆形/椭圆形指令	91
4.2.8	画直线指令	94
4.2.9	画矩形/圆角矩形指令	95
4.2.10	画三角形指令	98
4.2.11	画圆柱状体、画表格指令	100
4.2.12	开机设定指令	102
4.2.13	合并指令	103
4.2.14	图片卷动设定及显示指令	104
4.2.15	环形绘图设定及显示指令	106
4.2.16	进度条设定及显示指令	107
4.2.17	电阻屏控制指令	109
4.2.18	背光控制指令	110
4.2.19	串口屏侦测指令	111
4.2.20	二维码 (QR-Code) 设定及显示指令	113
4.2.21	声音控制指令	114
4.2.22	时钟控制指令	115

4.2.23 指令总表	116
4.3 使用 UartTFT_Tool 的设计流程	120
4.4 图文整合编译器操作说明	122
4.4.1 TFT 串口指令编译.....	123
4.4.2 制作图片的 Bin 文件.....	130
4.4.3 制作 GIF 檔的 Bin 文件.....	135
4.4.4 制作字库的 Bin 文件.....	140
4.4.4.1 全字库制作.....	140
4.4.4.2 自定义字库制作.....	143
4.4.5 制作 Wav 檔的 Bin 文件.....	146
4.4.5.1 音频文件转 WAV.....	146
4.4.5.2 制作 WAV Bin 文件	151
4.4.5.3 典型 PWM 音频驱动电路	154
4.4.6 Bin 文件的结合.....	155
5. 串口通讯软件(UartDebug.exe).....	158
5.1 打开 UartDebug 软件注意事项	159
5.2 UartDebug 的功能简介	161
5.3 仿真指令的加载	162
5.4 仿真指令的发送	165
5.5 发送循环仿真指令	167
5.6 指令列表的更改与保存.....	170
5.7 清除指令框指令与接收框数据.....	173
5.8 加载与更新 UserInfo.bin 文件	175
6. MCU 码与 Flash 更新.....	178
6.1 LT268B 的主程序更新.....	178
6.2 USB 更新外部的 SPI Flash	182
6.3 SD 卡更新外部的 SPI Flash.....	185
7. 版本记录.....	186
8. 版权说明.....	187

1. 概述

本应用手册主要在说明 LT268B 小尺寸 Uart 串口屏控制芯片的应用，及介绍串口屏开发工具 - [图文整合编译器 \(UartTFT_Tool.exe\)](#) 和 [图文 UI 编辑器 \(UI_Editor.exe\)](#) 的使用方式，让 TFT 屏厂的制造业者或是系统端的客户能够依据功能需求快速地规划及完成其产品在 TFT 屏的显示应用，避免为了处理 TFT 显示画面进行冗长的程序开发。

1.1 LT268B 基本介绍

LT268B 是针对小尺寸 MCU 屏所设计的 Uart 串口屏控制芯片。其内部采用 32bit M4 核心架构，主要的功能就是提供 Uart 串口通讯，让主控端 MCU 透过简易的指令就能轻易的将要显示到 TFT 屏的内容传递给小尺寸 MCU 屏上的 TFT 驱动器 (Driver)，LT268B 内部硬件及程序提供图形处理功能，能够提升 TFT 显示效率，及降低主控端 MCU 处理图形显示的时间，LT268B 支持小尺寸 MCU 接口的 TFT 屏，通常是 3.5" 以下、显示分辨率为 480*320 (HVGA) 以内，提供 SPI 或 8 位的 MCU 接口。

LT268B 内部的主频可达 120MHz，含有 256K bytes Flash、128K bytes SRAM，除了提供 Uart 串口通讯，也提供一 SPI Flash 接口，外接的 SPI Flash 可以用来储存图片、动画、字库等信息。LT268B 内的 MCU 程序已经包含了 [乐升半导体](#) 的串口协议，可以配合 [乐升半导体](#) 开发的 PC 上位机软件 ([UI_Editor](#) / [UartTFT_Tool](#))，直接在电脑上进行产品的 UI 显示接口开发，除了提升显示效率外，也大幅缩短 TFT 显示功能的开发周期。LT268B 内建串口指令功能包括图片显示、GIF 动画显示、循环图片显示、开机画面显示、进度条显示、文字字符串显示、二维码产生，及几何图形显示如画线、画圆、画三角形、画矩形等功能。LT268B 内的 MCU 程序除了串口协议外也含有升级程序，可使用 USB 接口及通讯软件，对内部 MCU 程序及 SPI Flash 数据进行更新，也可以透过 SD 卡更新 SPI Flash 内的数据，详细请参考第 6 章说明。

LT268B 的显示功能非常适合用在有小尺寸 TFT-LCD 屏的电子产品上，或是原使用单色屏而想进行升级的产品，如各式小家电、智能家电、工业控制板、电子仪器、医疗设备、小型检测设备等产品。下图为 LT268B 的应用方块图：

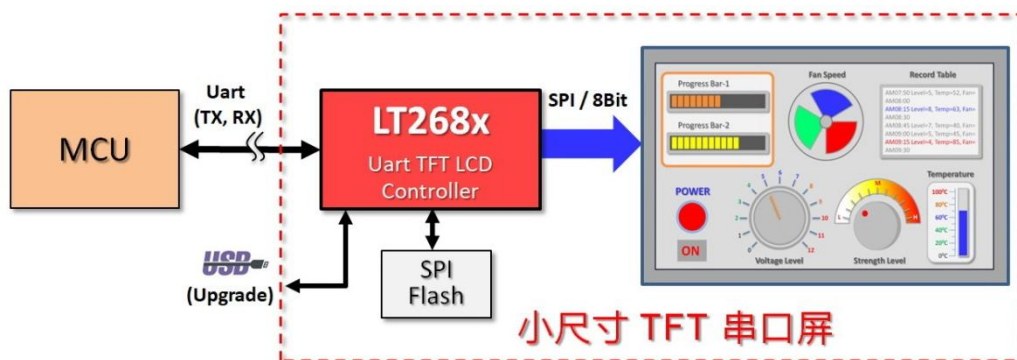


图 1-1: LT268B 应用方块图

1.2 TFT 串口屏的软硬件架构

所谓的 TFT 串口屏就是在 TFT 显示模块上加上 MCU 及 TFT 控制器, 该 MCU 负责接收主控端主板送来的串口 (Uart) 指令, 然后依据这些定义好的指令去显示出图片或是动画, 主控端主板上的 MCU 不需要为了繁琐的图片显示去编写复杂的程序, 因此 TFT 串口屏实际上就是一种指令屏的架构。

TFT 串口屏对主控端主要是透过 RS232 或是 RS485 接口来通讯, 如果主控端与 TFT 串口屏的距离很近 (~30cm 内), 可以将主控端 MCU 的 Uart 输出输入口直接接到 LT268B 串口屏上的 Uart 输出输入口, 如下图示意图:

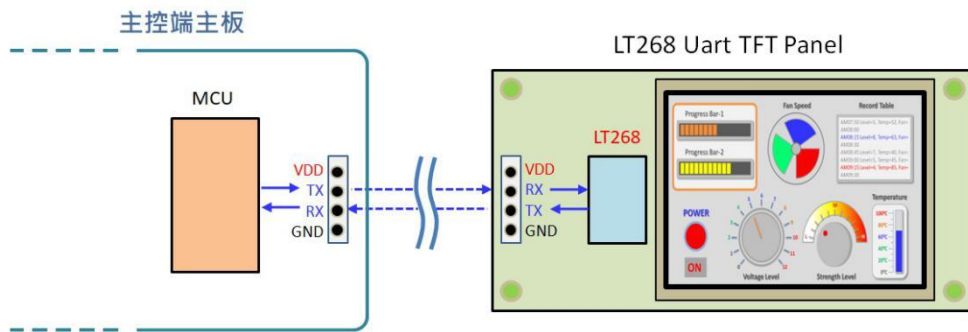


图 1-2: 主控端 MCU 的 Uart 与 LT268B 串口屏的 Uart 连接示意图

如果要达到较远距离的通讯效果, 通常需要加上 RS232 或是 RS485 的专用驱动芯片, 下图为主控端与 LT268B 串口屏的 RS232 驱动 IC 接口示意图:

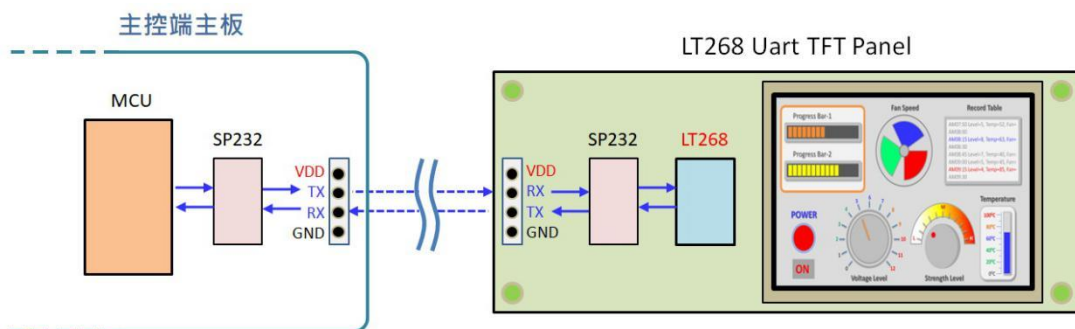


图 1-3: 主控端 MCU 与 RS232 驱动 IC 的接口示意图

在使用 LT268B 串口屏之前必须要用上位机软件做开发, 乐升半导体提供了 [图文整合编译器 \(UartTFT_Tool.exe\)](#) 及 [图文 UI 编辑器 \(UI_Editor.exe\)](#) 两种上位机软件, 两者都可以单独对 LT268B 的 TFT 串口屏进行设置及显示功能的开发, 上位机软件开发时会使用到的图片、文字、动画等信息产生 Bin 档, 开发者可以透过 USB 及使用 [LT268B_ISP_Vxx.exe](#) 程序 (参考第 6 章)、或是专用的 SPI Flash 烧录器将 Bin 档烧录到 SPI Flash 内, 然后透过 USB 转 Uart (RS232) 的控制线对 TFT 串口屏进行模拟, 也就是做 TFT 屏显示画面的前期验证。

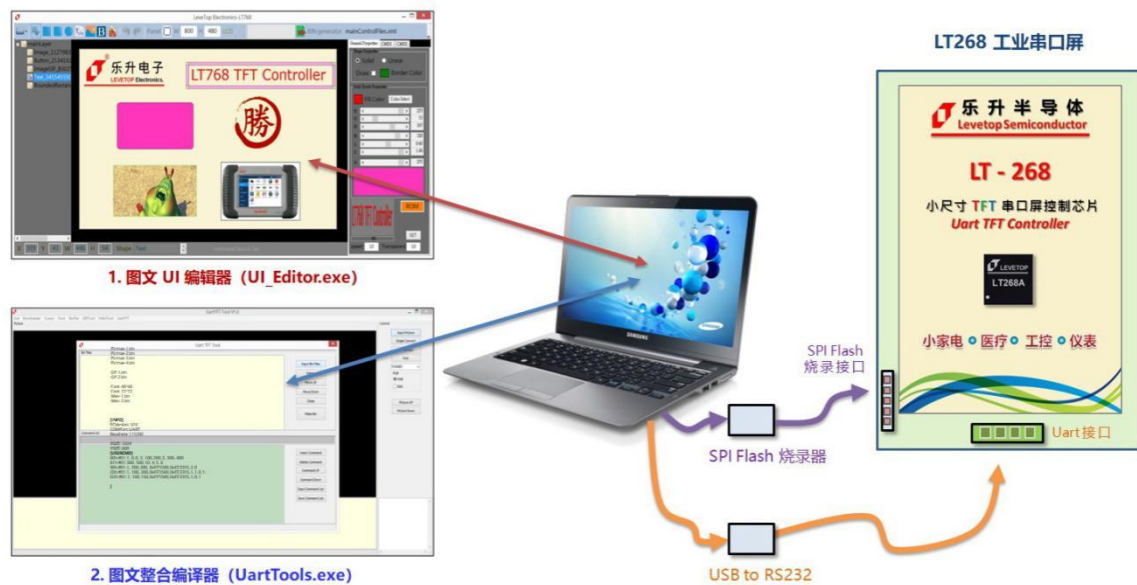


图 1-4：使用上位机软件开发的示意图

上位机软件会依据图片出现的顺序及方式产生指令格式，而前面所说的模拟就是以电脑透过 USB 转 Uart 控制线替代主控端主控发出指令，让开发者在上位机软件上做前期验证，如果上位机软件发出的指令格式都能在 TFT 屏上显示及达到开发者所要的效果，那么主控端最终就在其 MCU 程序上植入这些指令格式，在想要显示图片时送出对应的指令。下图 1-5 为主控板与 LT268B 串口屏连接的示意图：

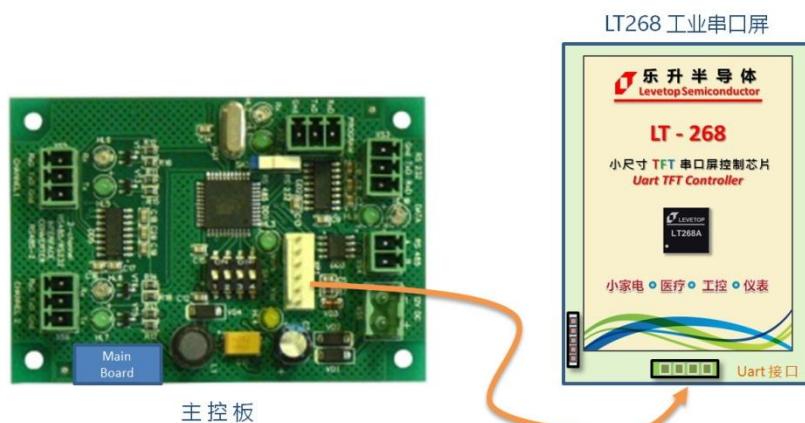


图 1-5：主控端主控板与 TFT 串口屏连接

图文整合编译器 (UartTFT_Tool.exe) 对每个 TFT 串口屏的显示动作都有一个固定的指令，例如 80h 就是显示图片的指令，UartTFT_Tool 会将使用的图片给予编号，在进行编译后会将所有图片、文字、动画等信息产生 Bin 档，开发者将 Bin 档烧录到 SPI Flash 内后，于验证的时候当电脑送出 80h、00h 那么 TFT 串口屏就会显示第一张图片，送出 80h、01h 就会显示第二张图片，当 UartTFT_Tool 发出的指令格式都能在 TFT 屏上显示及达到开发者所要的效果，就可以实地将主控端连接到 TFT 串口屏(如上图 1-5)，而主控端 MCU 程序送出 0xAA(Start)、80h、00h、1Bh(CRC1)、98h(CRC2)、0xE4(End1)、0x1B(End2)、0x11(End3)、0xEE(End4) 指令后，TFT 串口屏就会显示第一张图片，同时回传信息 0xAA(Start)、80h、

00h、00h、1Bh(CRC1)、98h(CRC2))、0xE4(End1)、0x1B(End2)、0x11(End3)、0xEE(End4) 给主控端，确认整个指令握手协议完成，如下图：

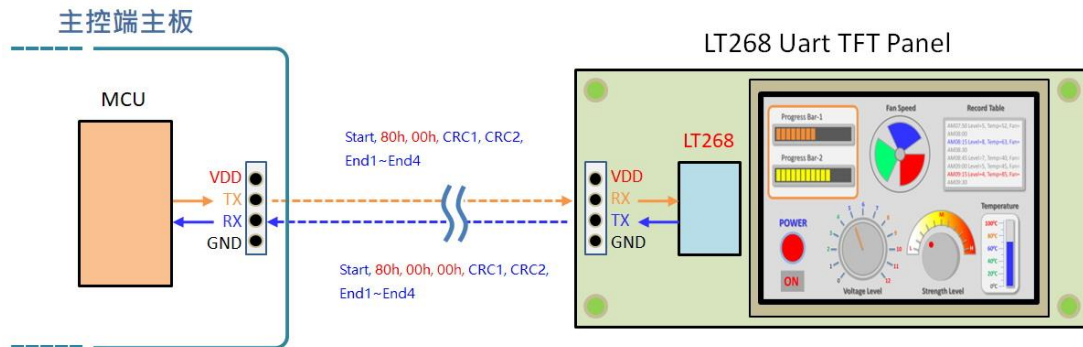


图 1-6：串口屏的指令协议范例一

当主控端 MCU 程序送出 0xAA(Start)、80h、01h、0Bh(CRC1)、B9h(CRC2)、0xE4(End1)、0x1B(End2)、0x11(End3)、0xEE(End4) 指令后，TFT 串口屏就会显示第二张图片，同时回传信息 0xAA(Start)、80h、01h、00h、1Bh(CRC1)、98h(CRC2)、0xE4(End1)、0x1B(End2)、0x11(End3)、0xEE(End4) 给主控端，确认整个握手协议完成，如下图：

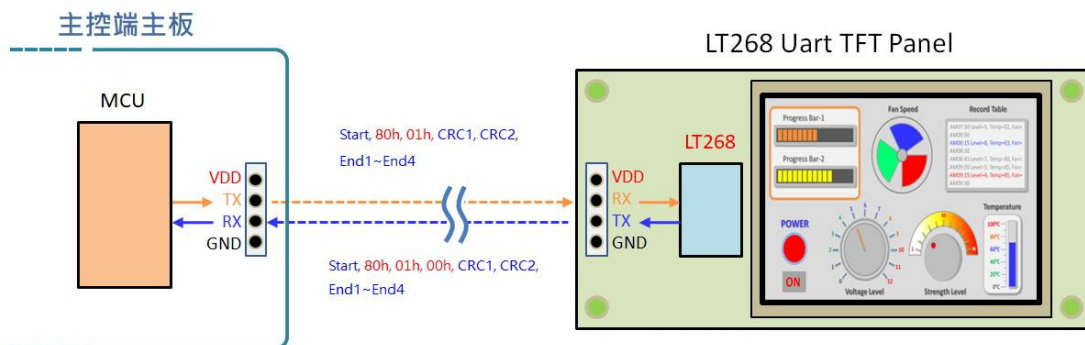


图 1-7：串口屏的指令协议范例二

为了确保主控端与 TFT 串口屏之间的数据传递正确，主控端 MCU 程序送出的指令还要加上 1 个 Byte 的 **起始码** (固定为 0xAA)、2 个 Byte 的 **CRC 码**、4 个 Byte 的 **结束码** (固定为 0x E4、0x 1B、0x 11、0x EE)，而 TFT 串口屏收到信息或是完成指令后会回传后信息给主控端的 MCU，主控端与串口屏的指令握手协议表请参考手册后面第 2.2 节。有关 2 个字节 CRC 的产生方式请参考手册后面第 2.3 节。

而 LT268B 的 TFT 串口屏还支持用 USB 接口更新，可以用 USB 接口对 LT268B 内部核心主程序或是 SPI Flash 进行数据更新，请参考下一节的原理图及手册后面第 6 章说明。

1.3 LT268B 串口屏原理图

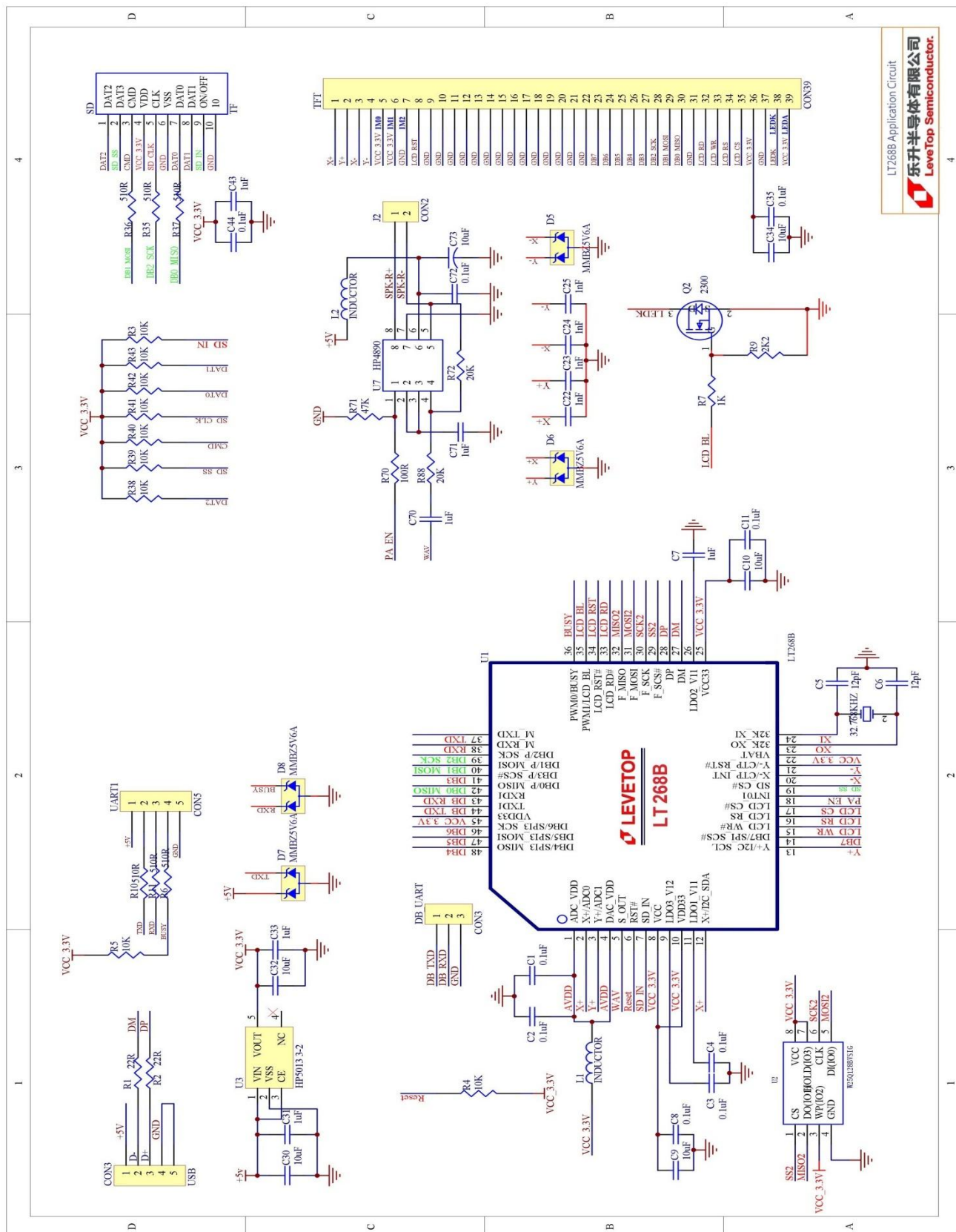


图 1-8: LT268B 参考原理图

LT268B_UartTFT_AP Note_CH / V1.1

2. 串口指令

为了让主控端的系统或是主板就能够透过 UART、SPI 等串口轻易的在 TFT 屏上显示图片或是信息，在 LT268B 串口屏上规划了一个串口指令集，透过定义好的指令码配合指令参数去改变 TFT 屏上的画面，乐升半导体提供 2 种串口屏的开发工具：[图文整合编译器 \(UartTFT_Tool.exe\)](#) 及 [图文 UI 编辑器 \(UI_Editor.exe\)](#)，使用者可以选择任一种进行 TFT 显示屏的方案开发，同时这 2 种串口屏的开发工具都可以实时模拟 TFT 显示屏的显示效果，做前期的验证。

2.1 串口屏指令集

LT268B 支持的 TFT 串口屏指令，包括图片静态显示、图片动态显示、文字显示、几何图形等等，如下表 2-1 所示。

表 2-1：LT268B 串口屏指令集

主功能	细项功能	指令码 (1Byte)	主功能	细项功能	指令码 (1Byte)
显示图片	单张/多张图片	80h, 8Ah, 8Fh	几何图形	画点	DFh
	循环拨放	81h, 84h		直线	E0h
	GIF 动画	88h, 89h		空心圆形	E1h
	弹出图片	D8h		实心圆形	E2h
	循环卷动	D9h, DBh		带框实心圆形	E3h
	数字图片	90h, 91h		空心椭圆	E4h
显示控件 图片	显示单一控件图片	A0h		实心椭圆	E5h
	取消单一控件图片	A1h		带框实心椭圆	E6h
	虚拟控件	A2h		空心矩形	E7h
	取消虚拟控件	A3h		实心矩形	E8h
指标与造图	进度条指标图	B0h		带框矩形	E9h
	环形指标图	DCh		空心圆角矩形	EAh
	二维码生成	98h		实心圆角矩形	EBh
显示字库	字库-1~4	C0h, C1h, C2h, C3h		带框圆角矩形	ECh
背光亮度	设置亮度	BAh		空心三角形	EDh
	On/Off	BCh		实心三角形	EEh
设定时钟	设定时钟	8Ch		带框三角形	EFh
	读取时钟	8Dh		圆柱体	F4h
开机指令、 合并指令	开机指令	9Ah		表格视窗	F6h
Wav 檔	播放	B8h	电阻屏校验	电阻屏校验指令	8Bh
	停止	B9h	串口屏侦测	联机检查	BEh
				版本侦测	BFh

2.2 主控端与 TFT 串口屏协议表

表 2-2：主控端与 TFT 串口屏协议表

主功能	细项功能	主控端发送 (TFT 串口屏接收)						主控端接收 (TFT 串口屏发送)					
		起始码 (1Bytes)	指令码 (1Byte)	序号 (1Byte)	指令参数	CRC 码 (2Bytes)	结束码 (4Bytes)	起始码 (1Bytes)	指令码 (1Byte)	序号 (1Byte)	信息码/ 反馈码 (1Bytes)	CRC 码 (2Bytes)	结束码 (4Bytes)
显示图片	单张/ 多张图片	Start	80h	nn		CRC	End	Start	80h	nn	信息码	CRC	End
	单张/ 多张图片	Start	8Ah	nn		CRC	End	Start	8Ah	nn	信息码	CRC	End
	单张图片	Start	8Fh	nn	X, Y, PNG, Pnn	CRC	End	Start	8Fh	nn	信息码	CRC	End
	循环拨放	Start	81h	nn		CRC	End	Start	81h	nn	信息码	CRC	End
	取消循环 拨放	Start	84h	nn		CRC	End	Start	84h	nn	信息码	CRC	End
	GIF 动画	Start	88h	nn		CRC	End	Start	88h	nn	信息码	CRC	End
	取消 GIF 动 画	Start	89h	nn		CRC	End	Start	89h	nn	信息码	CRC	End
	弹出图片	Start	D8h	nn		CRC	End	Start	D8h	nn	信息码	CRC	End
	循环卷动	Start	D9h	nn		CRC	End	Start	D9h	nn	信息码	CRC	End
	取消循环卷 动	Start	DBh	nn		CRC	End	Start	DBh	nn	信息码	CRC	End
	数字图片 1	Start	90h	nn	ddd.d	CRC	End	Start	90h	nn	信息码	CRC	End
	数字图片 2	Start	91h	nn	ddd.d	CRC	End	Start	91h	nn	信息码	CRC	End
显示控件图片	显示单一控 件图片	Start	A0h	nn		CRC	End	Start	A0h	nn	信息码	CRC	End
		按下控件图片时						Start	A0h	nn	31h	CRC	End
		放开控件图片时						Start	A0h	nn	30h	CRC	End
	取消单一控 件图片	Start	A1h	nn		CRC	End	Start	A1h	nn	信息码	CRC	End
	虚拟控件	Start	A2h	nn		CRC	End	Start	A2h	nn	信息码	CRC	End
		按下控件图片时						Start	A2h	nn	31h	CRC	End
		放开控件图片时						Start	A2h	nn	30h	CRC	End
	取消虚拟控 件	Start	A3h	nn		CRC	End	Start	A3h	nn	信息码	CRC	End
指标与造图	进度条 指标图	Start	B0h	nn	Value (2 Bytes)	CRC	End	Start	B0h	nn	信息码	CRC	End
	环形指标图	Start	DCh	nn	S_Angle, A_Angle	CRC	End	Start	DCh	nn	信息码	CRC	End
	二维码生成	Start	98h	nn	字符串	CRC	End	Start	98h	nn	信息码	CRC	End

表 2-2：主控端与 TFT 串口屏协议表（续）

主功能	细项功能	主控端发送 (TFT 串口屏接收)						主控端接收 (TFT 串口屏发送)					
		起始码 (1Bytes)	指令码 (1Byte)	序号 (1Byte)	指令参数	CRC 码 (2Bytes)	结束码 (4Bytes)	起始码 (1Bytes)	指令码 (1Byte)	序号 (1Byte)	信息码/ 反馈码 (1Bytes)	CRC 码 (2Bytes)	结束码 (4Bytes)
显示字符串	字库-1	Start	C0h	nn	字符串 String	CRC	End	Start	C0h	nn	信息码	CRC	End
	字库-2	Start	C1h	nn	字符串 String	CRC	End	Start	C1h	nn	信息码	CRC	End
	字库-3	Start	C2h	nn	字符串 String	CRC	End	Start	C2h	nn	信息码	CRC	End
	字库-4	Start	C3h	nn	字符串 String	CRC	End	Start	C3h	nn	信息码	CRC	End
背光亮度	设置亮度	Start	BAh		BL (00~0Fh)	CRC	End	Start	BAh	BL (00~0Fh)	信息码	CRC	End
	On/Off	Start	BCh		00 或 01	CRC	End	Start	BCh	00 或 01	信息码	CRC	End
Wav 檔	播放	Start	B8h		REP(Bit7) + WAV 编号	CRC	End	Start	B8h	REP(Bit7) + WAV 编号	信息码	CRC	End
	停止	Start	B9h			CRC	End	Start	B9h	00	信息码	CRC	End
开机指令	开机指令	Start	9Ah	00		CRC	End	Start	9Ah	00	信息码	CRC	End
合并指令	合并指令	Start	9Ah	nn		CRC	End	Start	9Ah	nn	信息码	CRC	End
设定时钟	设定时钟	Start	8Ch		Y, M, D, H, M, S, W (7 Bytes)	CRC	End	Start	8Ch	00	信息码	CRC	End
	读取时钟	Start	8Dh			CRC	End	Start	8Dh	Y, M, D, H, M, S, W (8)	信息码	CRC	End
电阻屏校验	电阻屏校验	Start	8Bh			CRC	End	Start	8Bh	00	信息码	CRC	End
串口屏 侦测	联机检查	Start	BEh			CRC	End	Start	BEh	00	5Ah, or 55h	CRC	End
	版本检查	Start	BFh			CRC	End	Start	BFh	MCU Code(5Byte s) + Module Info. (42)	信息码	CRC	End

表 2-2：主控端与 TFT 串口屏协议表（续）

主功能	细项功能	主控端发送 (TFT 串口屏接收)						主控端接收 (TFT 串口屏发送)					
		起始码 (1Bytes)	指令码 (1Byte)	序号 (1Byte)	指令参数	CRC 码 (2Bytes)	结束码 (4Bytes)	起始码 (1Bytes)	指令码 (1Byte)	序号 (1Byte)	信息码/ 反馈码 (1Bytes)	CRC 码 (2Bytes)	结束码 (4Bytes)
几何图形	画点	Start	DFh	nn		CRC	End	Start	DFh	nn	信息码	CRC	End
	直线	Start	E0h	nn		CRC	End	Start	E0h	nn	信息码	CRC	End
	空心圆形	Start	E1h	nn		CRC	End	Start	E1h	nn	信息码	CRC	End
	实心圆形	Start	E2h	nn		CRC	End	Start	E2h	nn	信息码	CRC	End
	带框实心圆形	Start	E3h	nn		CRC	End	Start	E3h	nn	信息码	CRC	End
	空心椭圆	Start	E4h	nn		CRC	End	Start	E4h	nn	信息码	CRC	End
	实心椭圆	Start	E5h	nn		CRC	End	Start	E5h	nn	信息码	CRC	End
	带框实心椭圆	Start	E6h	nn		CRC	End	Start	E6h	nn	信息码	CRC	End
	空心矩形	Start	E7h	nn		CRC	End	Start	E7h	nn	信息码	CRC	End
	实心矩形	Start	E8h	nn		CRC	End	Start	E8h	nn	信息码	CRC	End
	带框矩形	Start	E9h	nn		CRC	End	Start	E9h	nn	信息码	CRC	End
	空心圆角矩形	Start	EAh	nn		CRC	End	Start	EAh	nn	信息码	CRC	End
	实心圆角矩形	Start	EBh	nn		CRC	End	Start	EBh	nn	信息码	CRC	End
	带框圆角矩形	Start	ECh	nn		CRC	End	Start	ECh	nn	信息码	CRC	End
	空心三角形	Start	EDh	nn		CRC	End	Start	EDh	nn	信息码	CRC	End
	实心三角形	Start	EEh	nn		CRC	End	Start	EEh	nn	信息码	CRC	End
	带框三角形	Start	EFh	nn		CRC	End	Start	EFh	nn	信息码	CRC	End
	圆柱体	Start	F4h	nn		CRC	End	Start	F4h	nn	信息码	CRC	End
	表格视窗	Start	F6h	nn		CRC	End	Start	F6h	nn	信息码	CRC	End

2.3 RS-232(UART) 通讯协议

主控端的系统或是主板透过 UART 串口传递显示指令给 LT268B 串口屏时，除了 **指令码**、**序号**、**指令参数** 外还要加上 1 个 Byte 的 **起始码** (固定为 0xAA)、2 个 Byte 的 **CRC 码**、4 个 Byte 的 **结束码** (固定为 0xE4、0x1B、0x11、0xEE)，指令信息如下表：

表 2-3：串口屏接收的指令信息

起始码	指令码	序号	指令参数	CRC 码	结束码
0xAA (1 Byte)	1 Byte	1 Byte	n Bytes	2 Bytes	0xE4、0x1B、0x11、0xEE (4 Bytes)

CRC 码协议如下：

```

chkSum = Rx_CRC_CCITT(txBuf,txLen);
txBuf[txLen++] = (chkSum>>8)&0xFF;
txBuf[txLen++] =  chkSum&0xFF;

unsigned int Rx_CRC_CCITT(unsigned char *puchMsg, unsigned int usDataLen)
{
    unsigned char i = 0;
    unsigned short wCRcIn = 0x0000;
    unsigned short wCPoly = 0x1021;
    unsigned char wChar = 0;

    while (usDataLen--)
    {
        wChar = *(puchMsg++);
        wCRcIn ^= (wChar << 8);
        for(i = 0; i < 8; i++)
        {
            if (wCRcIn & 0x8000)
                wCRcIn = (wCRcIn << 1) ^ wCPoly;
            else
                wCRcIn = wCRcIn << 1;
        }
    }
    return (wCRcIn);
}

```

串口屏在收到主控端的系统或是主板指令后会通常会响应 10 个 Byte 信息，包括 **起始码**、**指令码**、**序号**、**信息码**、**CRC 码**、**结束码**，第一个 Byte 是**起始码**，然后是传回所收到的指令，第三个是**序号**，第四个传回串口屏执行结果的**信息码**，第五、六个是 **CRC 码**，最后是 4 个 Bytes 的 **结束码**：

表 2-4：串口屏反馈的信息

起始码	指令码	序号	信息码	CRC 码	结束码
0xAA (1 Byte)	1 Byte	一般指令 (1 Byte) BFh 指令 (47 Bytes)	1Byte 0x00: 执行完该指令 0x01: 串口指令参数错误 0x02: 不存在该指令 0x03: 指令 Flash 配置溢出 0x04: CRC 码校正错误 0x05: Flash 数据异常 BEh 指令: 0x5A: Ready 0x55: Not Ready A0h 控件指令: 0x31: 按下控件 0x30: 放开控件	2 Bytes	0xE4、0x1B、 0x11、0xEE (4 Bytes)

在串口屏反馈的信息结构中，序号在某些指令也代表不同的意思，如设置亮度 BAh 指令其序号代表背光亮亮度、版本检查 BFh 指令其序号有 47 个 Bytes 代表串口屏信息。

2.4 主控端发送指令的范例

以下是主控端的 MCU 透过 UART 串口传递显示命令给 LT268B TFT 串口屏的程序范例，本程序以传送显示第一张图片（80h、00h）为范例，程序内会自动加入 0xAA 起始码、2 个 Byte 的 CRC 码及 4 个 Byte 的 结束码：

```
int main (void)
{
    char c[] = "80 00";          //发送 80h 指令, 和 00 序号
    Send(c);

    while(1);
}

unsigned short Rx_CRC_CCITT(unsigned char *puchMsg, unsigned int usDataLen) // 生成 CRC
{
    unsigned char i = 0;
    unsigned short wCRCCin = 0x0000;
    unsigned short wCPoly = 0x1021;
    unsigned char wChar = 0;

    while (usDataLen--)
    {
        wChar = *(puchMsg++);
        wCRCCin ^= (wChar << 8);
        for(i = 0; i < 8; i++)
        {
            if (wCRCCin & 0x8000)
                wCRCCin = (wCRCCin << 1) ^ wCPoly;
            else
                wCRCCin = wCRCCin << 1;
        }
    }
    return (wCRCCin);
}

void Send(char *c)                //发送命令函数
{
    unsigned char Sendbuff[100]={0};
    unsigned short Send_CRC = 0;
    unsigned char C_flag = 0;      //判断是否在" "中

    int i = 0, j = 0;

    if(((c[0]>=0x30 && c[0]<=0x39) || (c[0]>=0x41 && c[0]<=0x5A)) || ((c[1]>=0x30 && c[1]<=0x39) || (c[1]>=0x41 && c[1]<=0x5A))) //只有第一个和第二个字符是有效的才是命令
    {
        while(c[i] != '\0')
        {
            if(c[i] != ' ')                //排除空格，非空格可以进入
            {
                if(c[i] == '"')            //当" "中的字符 以 ASCII 输出，不需要输出" 号，
            }
        }
    }
}
```

```

{
    C_flag++;
    i++;
}
if(C_flag == 1)
{
    if(c[i] != '"')
    {
        Sendbuff[j] = c[i];          //ASCII 直接输出
        i++;
        j++;
    }

}
else if(C_flag == 2)                //第二次遇到"
{
    C_flag = 0;
    i++;
}
if(C_flag == 0)
{
    if(c[i] == '/') break;

    if(c[i]>=0x30 && c[i]<=0x39) //0~9
    {
        Sendbuff[j] = ((c[i] - 0x30)<< 4 );
        i++;

        if(c[i]>=0x30 && c[i]<=0x39)
        {
            Sendbuff[j] += (c[i] - 0x30);
            i++;
            j++;
        }
        else if(c[i]>=0x41 && c[i]<=0x5A)
        {
            Sendbuff[j] += (c[i] - 0x37);
            i++;
            j++;
        }
    }

}

else if(c[i]>=0x41 && c[i]<=0x5A) //A~Z
{
    Sendbuff[j] = ((c[i] - 0x37)<< 4 );
    i++;

    if(c[i]>=0x30 && c[i]<=0x39)
    {
        Sendbuff[j] += (c[i] - 0x30);
        i++;
        j++;
    }
    else if(c[i]>=0x41 && c[i]<=0x5A)
    {
        Sendbuff[j] += (c[i] - 0x37);
        i++;
        j++;
    }
}

```

```

        }

    }

    }
    }
    else i++;
}
Sendbuff[j] = '\0';
// printf("%s\r\n",Sendbuff);

/*****CRC 和针头针尾*****/

Send_CRC = Rx_CRC_CCITT(Sendbuff,j);

Sendbuff[j] = Send_CRC>>8 & 0xff;
Sendbuff[j+1] = Send_CRC & 0xff;

for(i = 0;i<j+2;i++)
    Sendbuff[j+2-i]=Sendbuff[j+1-i];           //挪位

Sendbuff[0] = 0xAA;
Sendbuff[j+3] = 0xE4;
Sendbuff[j+4] = 0x1B;
Sendbuff[j+5] = 0x11;
Sendbuff[j+6] = 0xEE;

j+=7;

for(i = 0; i < j; i++)
{
    USART_DATA(USART0) = (uint8_t) Sendbuff[i];
    while(usart_flag_get(USART0, USART_FLAG_TBE)==0){};    //循环发送,直到发送完毕
}

}
}

```

3. 图文 UI 编辑器 (UI_Editor.exe)

3.1 UI_Editor 界面的简介

UI_Editor.exe 是乐升半导体提供的一款以串口屏为对象的**图文 UI 编译器**。它的功能是根据客户的需求，将串口屏要用到的图片、文字、配置数据等信息打包生成 BIN 档。客户可以使用 UI_Editor 简单、快捷的制作 UI 界面，之后将生成的 BIN 文档烧录到 SPI Flash 中。











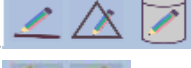


注意：UI_Editor 是在 **Microsoft.NET Framework 4.6.2** 的环境中编写出来的，所以电脑系统必须安装 Microsoft.NETFramework4.6.2 才能正常使用。

UI_Editor 的界面由各种按钮和屏幕框组成如，下图所示：



图 3-1: UI_Editor 界面

所有的 UI 设计都在屏幕框内完成，用户根据需求选用不同的功能实现设计。其中 268 支持的功能键详细如下：

1.  添加图片按钮
2.  添加控件按钮
3.  添加 GIF 图按钮
4.  添加文字按钮（图片）
5.  添加数字按钮（图片）
6.  添加文字按钮（字库）
7.  添加表格
8.  添加圆环
9.  添加进度条按钮
10.  分别是画矩形、画圆角矩形、画圆、画椭圆
11.  分别是画线、画三角形、画圆柱体
12.  分别是撤回操作和恢复操作按钮（当无任何操作时初始画面是  ）

与 UI_Editor 工具同级的有几个文件夹，它们的作用如下图所示。

- ★ FONT 文件夹用来存放需要使用的字库。
- ★ PICFILE 文件夹可用来先存放需要使用到的图片文件。
- ★ PROJECT 文件夹备份着每次 Save 和 Build 的工程文件。

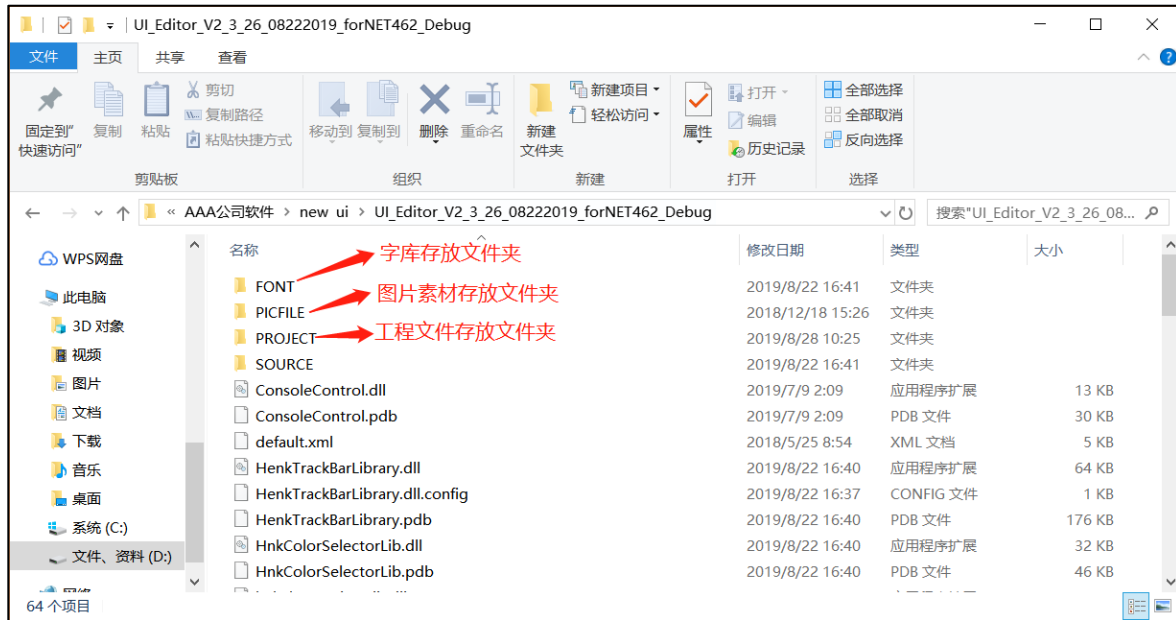


图 3-2: UI_Editor 工具同级文件目录

PROJECT 文件夹下级的工程文件里有几个文件夹，它们的作用如下图所示。

- ★ BINFILE 文件夹存放着编译好的 BIN 文件,需要烧录的 **UserInfo** 和 **UartTFT_Flash** 就存放在此处。
- ★ COMMANDFILE 文件夹存放着工程储存文件。
- ★ PICFILE 文件夹存放着编译后的图片文件。
- ★ SRCPIC 文件夹存放着编译前的图片。

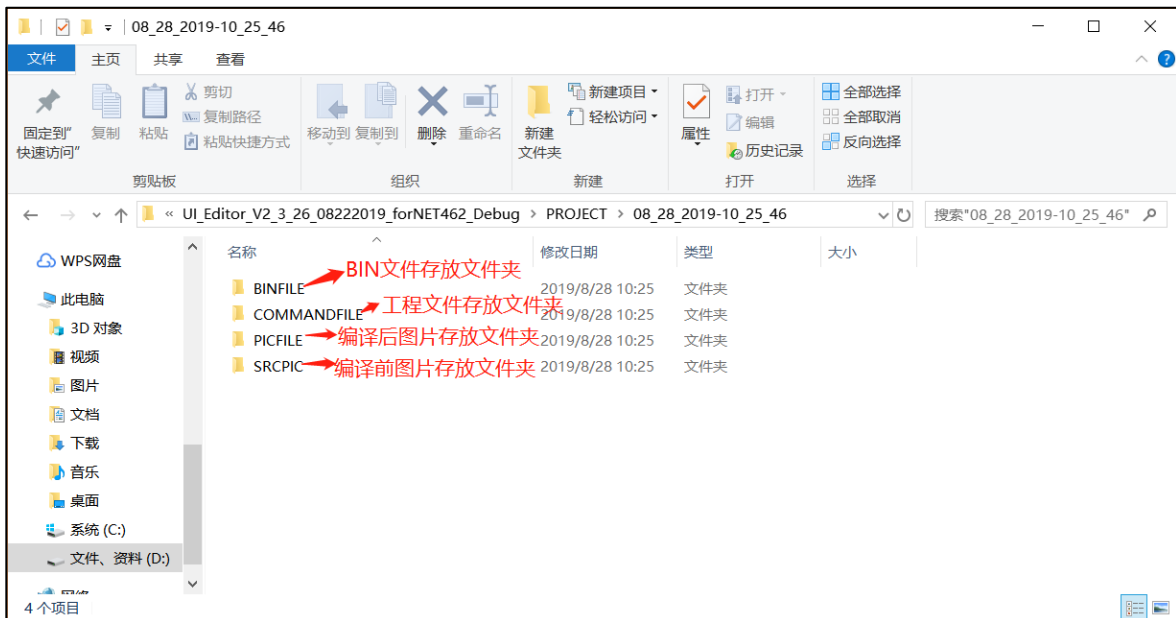


图 3-3: PROJECT 文件夹下级的工程文件目录

在菜单按钮里，有 New Project、load、save、Project Folders 四个按钮。

- ★ New Project: 创建新工程
- ★ Load: 装载工程文件
- ★ Save: 保存当前工程（除了菜单有 Save 按钮，软件界面也有一个 Save 按钮）。
- ★ Project Folders: 打开工程存放文件夹。



图 3-4: UI_Editor 菜单选项

按 save 按钮会把工程以 mainControlFiles.xml 文件保存在 PROJECT 下级中以时间命名的 COMMANDFILE 文件夹里。使用 Load 功能在 PROJECT 下级找到对应时间的文件夹里 COMMANDFILE 文件夹的 mainControlFiles.xml 文件，就可以重新加载工程。

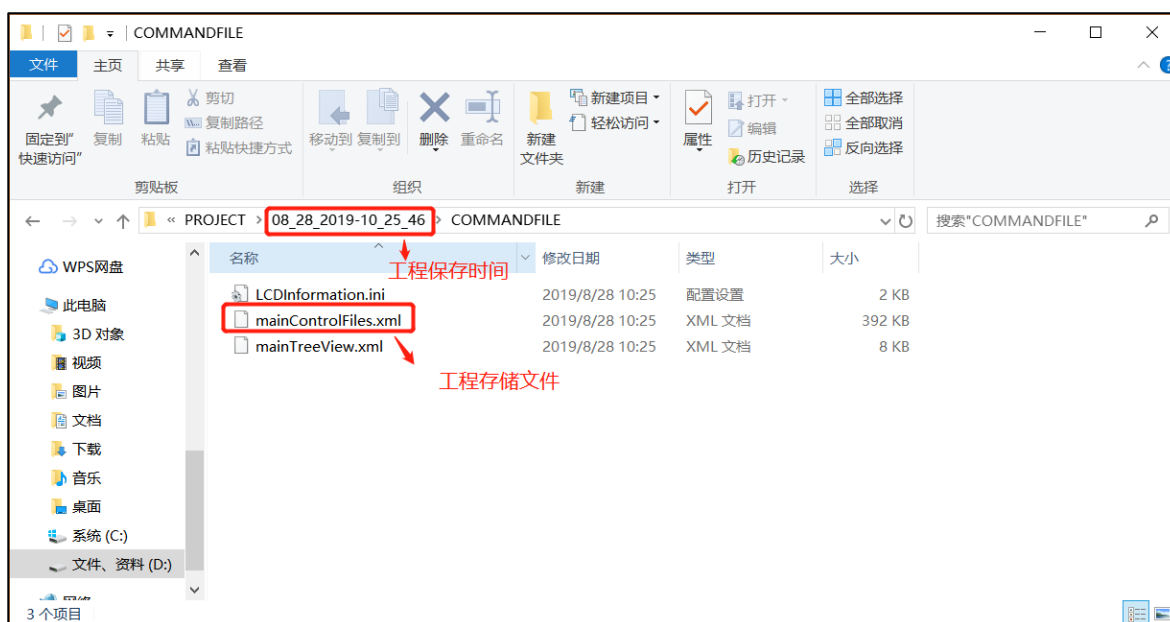


图 3-5: UI_Editor 重装载工程文件

3.1.1 使用 UI_Editor 的设计流程

下图为用图文 UI 编译器 (UI_Editor.exe) 开发的详细流程图, 用户也可以至乐升半导体网站下载 UI_Editor 的 LT268B 范例来操作, 将更快速的了解开发模式。同时建议用户先依据所需功能及 TFT 屏幕大小准备好素材, 因为这些显示图片、动画档、文字库等是存放在 SPI Flash 内, 数据量都不小, 而 SPI Flash 的烧录所需时间较长, 应该用 UI_Editor 串口屏仿真器先做前期验证, 尽量避免开发中反复对 SPI Flash 进行 UartTFT_Flash.bin 档的烧写, 以免延误开发效率。UI_Editor 图文 UI 编译器及 UI_Editor 串口屏仿真器软件可以自 [乐升半导体 官网“串口屏上位机软件”](#) 下载专区下载。

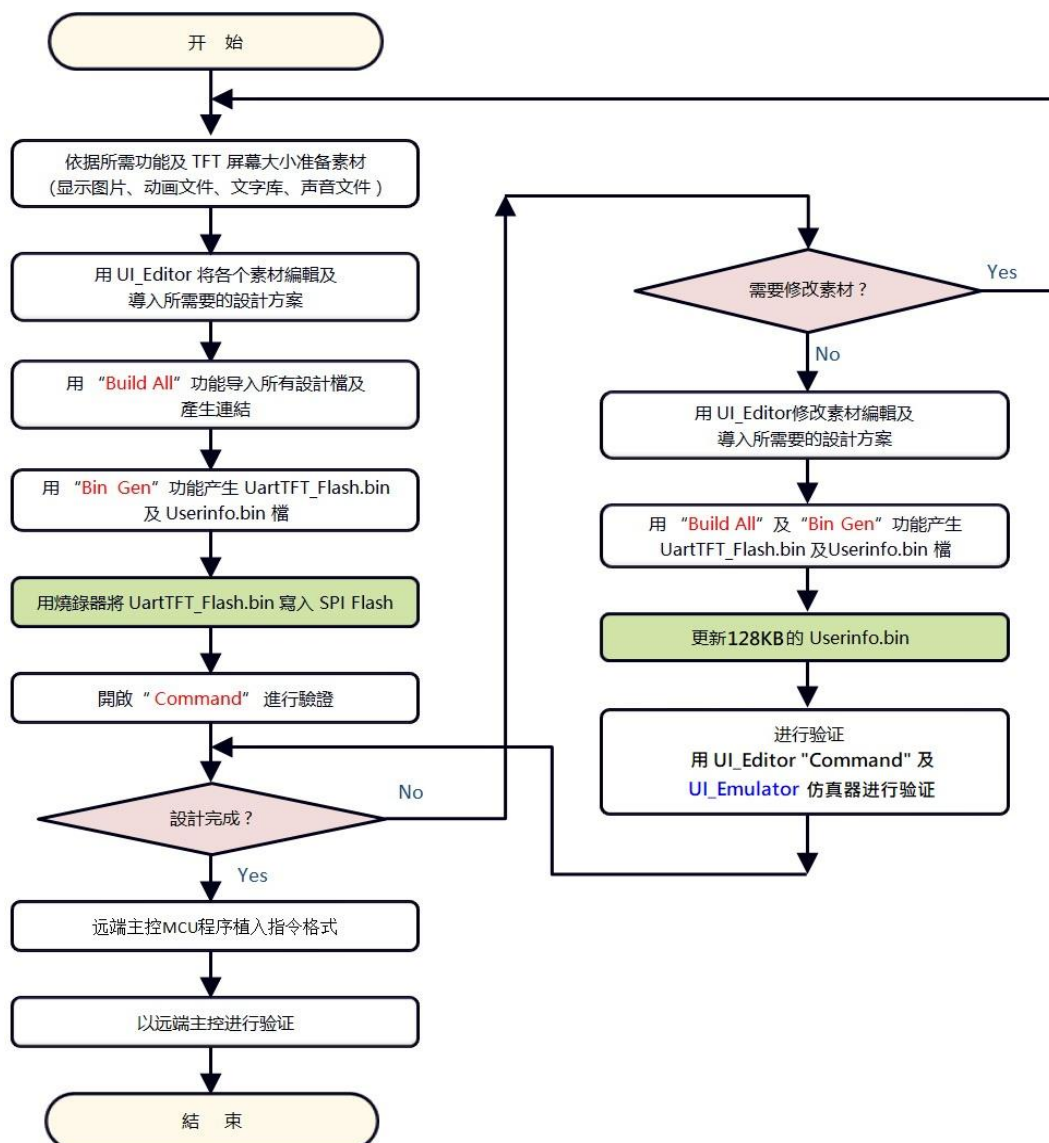


图 3-6: 使用 UI_Editor 的设计流程

有关 LT268B 使用 UI_Editor 的环境做开发, 使用者可以自本公司网址(www.levetop.cn)下载 LT268B 的 UI_Editor 演示案例 “LT268B_UI_Editor_Demo_240x320.rar” 来进行操作。

3.1.2 新建工程和背景图的设定

先打开 UI_Editor。



图 3-7: 打开的 UI_Editor

新建工程有以下两种方式：

方式一：在菜单中选择“New Project”，根据需要修改屏幕分辨率（修改之后请按回车键确认）。
点击“屏幕框选择按钮”，双击框内任意位置改变主图层图片，如下图所示：



图 3-8: 新建工程方式一

方式二：点击“初始化屏幕按钮”，直接选择在主图层显示的图片，之后根据需要修改屏幕分辨率（修改之后请按回车键确认），如下图所示：



图 3-9：新建工程方式二

通过修改分辨率可以改变屏幕的长、宽。注意，每个框中的数字修改后需要按回车键来确认，否则无法修改。例如要设定 2.4 寸 240X320 分辨率的屏幕，则将 W 修改为 240 按下回车键确认，再将 H 修改为 320 按下回车键确认。进行 UI 设计时请确认分辨率正确。



图 3-10：设置屏幕分辨率

点击“屏幕框选择按钮”后，整个屏幕框可以被任意拖动，可放在视野适合位置。再次点击该按钮后，即取消屏幕框选择功能，无法修改主图层背景图和拖动屏幕框。



图 3-11：拖动屏幕框到任意位置



图 3-12：取消了屏幕框选择功能

3.1.3 显示图的设定

初始化屏幕框之后，屏幕框默认显示的是主图层。我们可以通过在项目栏中主图层选项中右键，选择 Add New Page 来增加次图层。



图 3-13：增加图层

如果所建图层只需与主图层共享一张图片，这时在此界面点击“取消”即可。

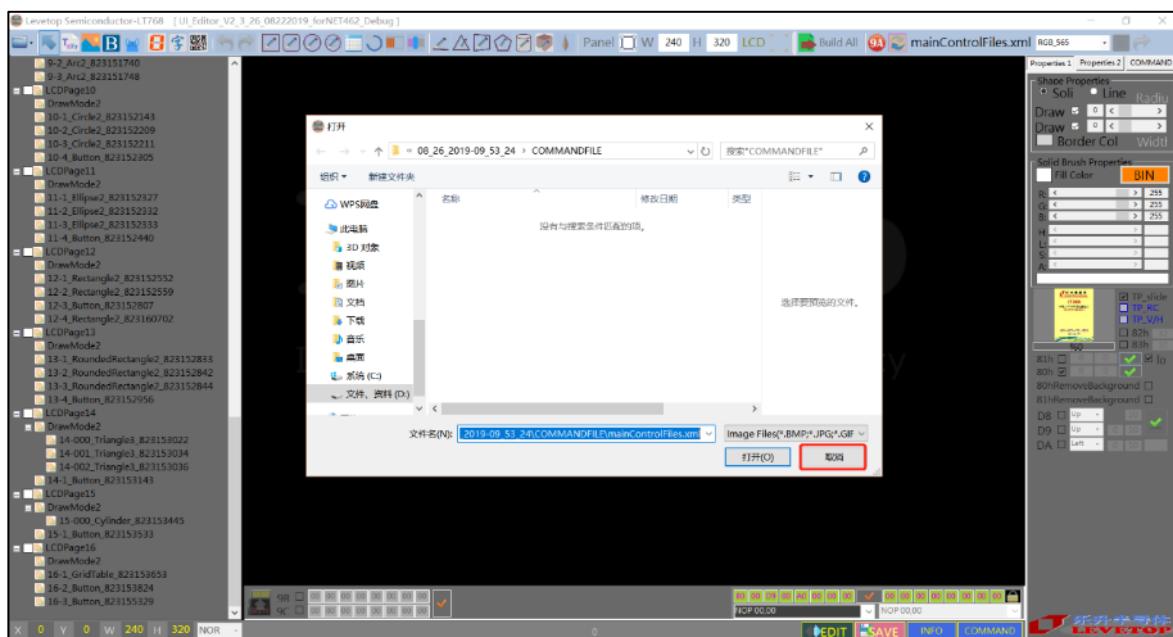


图 3-14：新建次图层与主图层共享

如果所建次图层需要新建一张独立的图片，选择图片后“打开”。

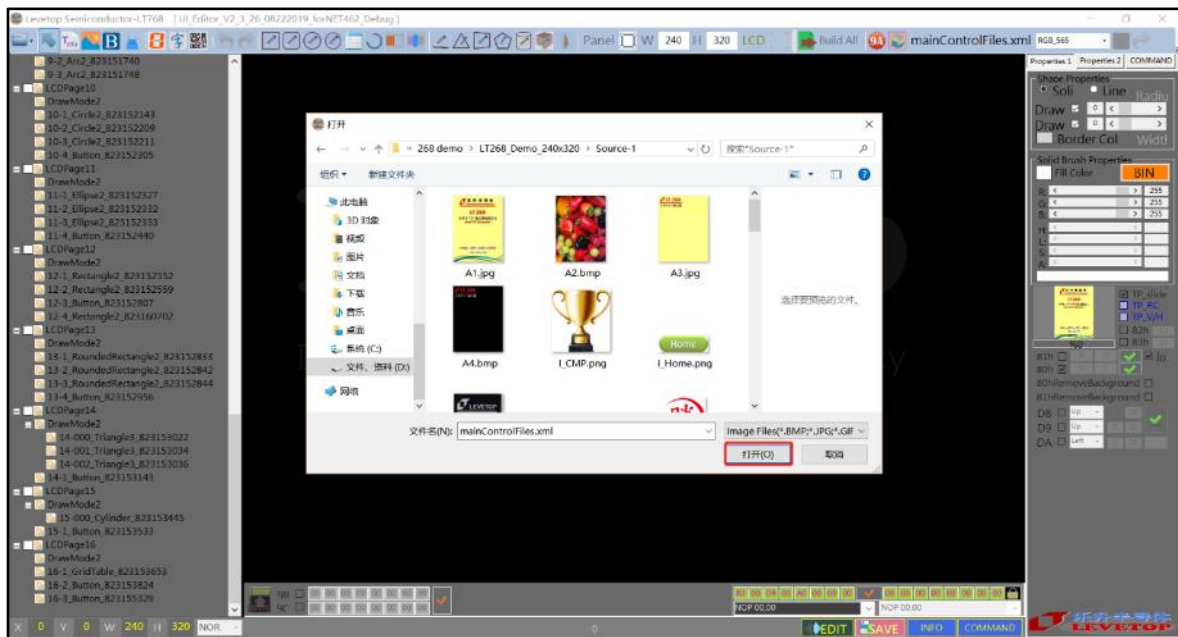


图 3-15: 新建独立的图层

如果需要变更次图层的背景图，先勾选次图层，点击该图层的“DrawMode”，再点击该图层的“LCDPage”，接着双击缩略的显示框，即可根据需求来修改次图层的背景图。



图 3-16: 变更次图层背景

当存在多个图层的时候,可以通过勾选图层前面的小方格来显示和隐藏当前图层。在当前的图层,右击该图层可对该图层进行删除,右键选中该图层下的图片或控件也可进行删除操作。



图 3-17: 删除或移动图层

3.1.4 界面的编辑与调试

如下图所示，鼠标左键选中控件，点击右键，选择“Remove”执行删除控件操作。也可以在界面左边，右键选中该图层下的图片或控件进行删除操作。

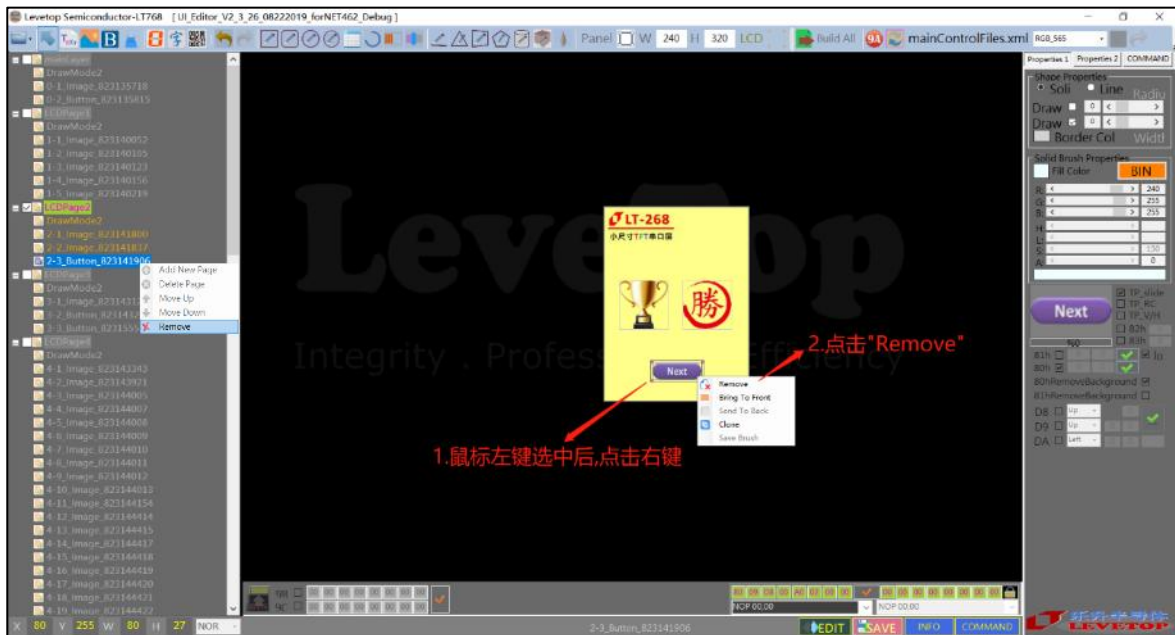


图 3-18：删除操作

复制控件的方式有以下两种：

方式一：如下图所示，“Clone”功能默认是异位置复制，先鼠标左键选中控件，再右键后选择“Clone”就会产生一个相同但异位置的控件。



图 3-19：异位置复制操作

如下图所示，复制前先双击“Clone 按钮”的灰色框，此时状态变为同位置复制，选中要复制的控件，点击右键后选择“Clone”就会产生一个等大小、同位置的控件。



图 3-20：同位置复制操作

方式二：“Copy”功能，首先，鼠标左键选中控件，再右键选择“Copy”。在当前图层，通过鼠标右键呼出选项，选择“Paste”来复制控件。也可以勾选另一个图层，点击“Draw Mode”，再通过“Paste”达到异图层复制。

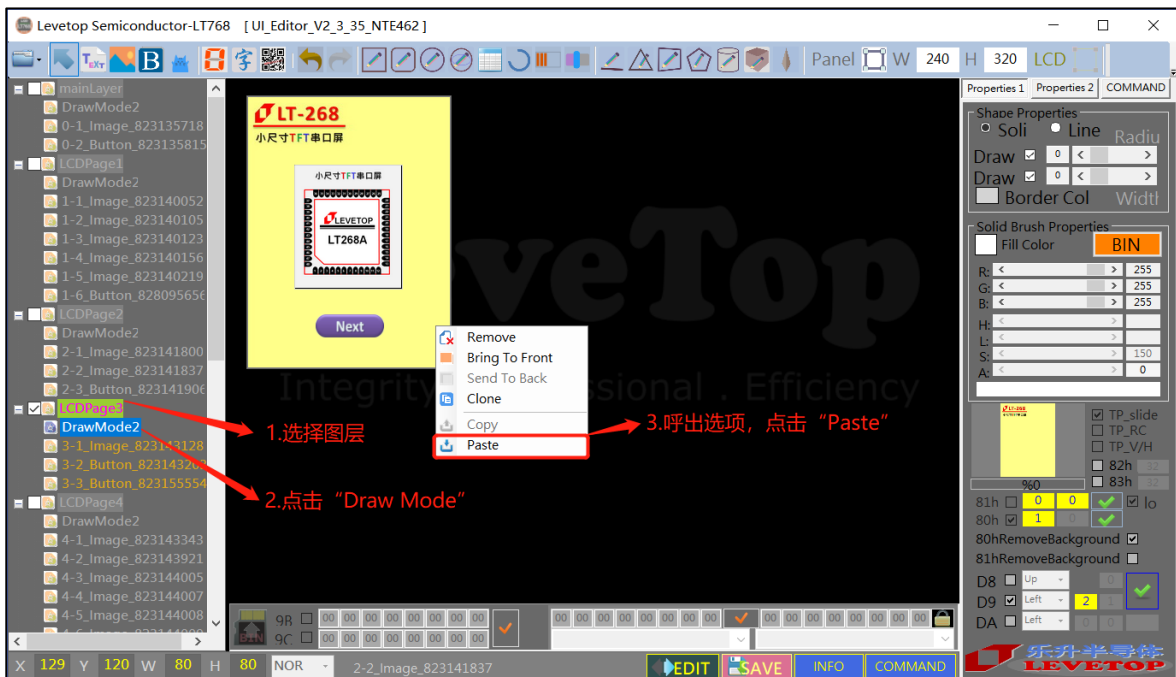


图 3-21：异图层复制

如下图所示，表示当前处在正常编辑的状态下，所有的添加、删除、编辑功能都能正常执行。



图 3-22：编辑状态

如下图所示，表示当前处在调试的状态下，此时不能对界面进行编辑，选中对应的控件点击右键，此时会弹出相关的指令，点击相应的指令即为发送该串口指令，要想重新对界面进行编辑，再次点击下图红色框的这个图标即可。



图 3-23：调试状态

3.1.5 工程编译和 BIN 文件生成

完成了整个 UI 画面的设计后，按下工程编译按钮（包含工程保存、编译后素材保存）：



图 3-24：工程编译



图 3-25：工程编译成功

等待工程编译成功之后，再按 BIN 按钮可以生成 BIN 文件到对应工程的 BINFILE 文件夹下。生成 BIN 文件成功，如下图：

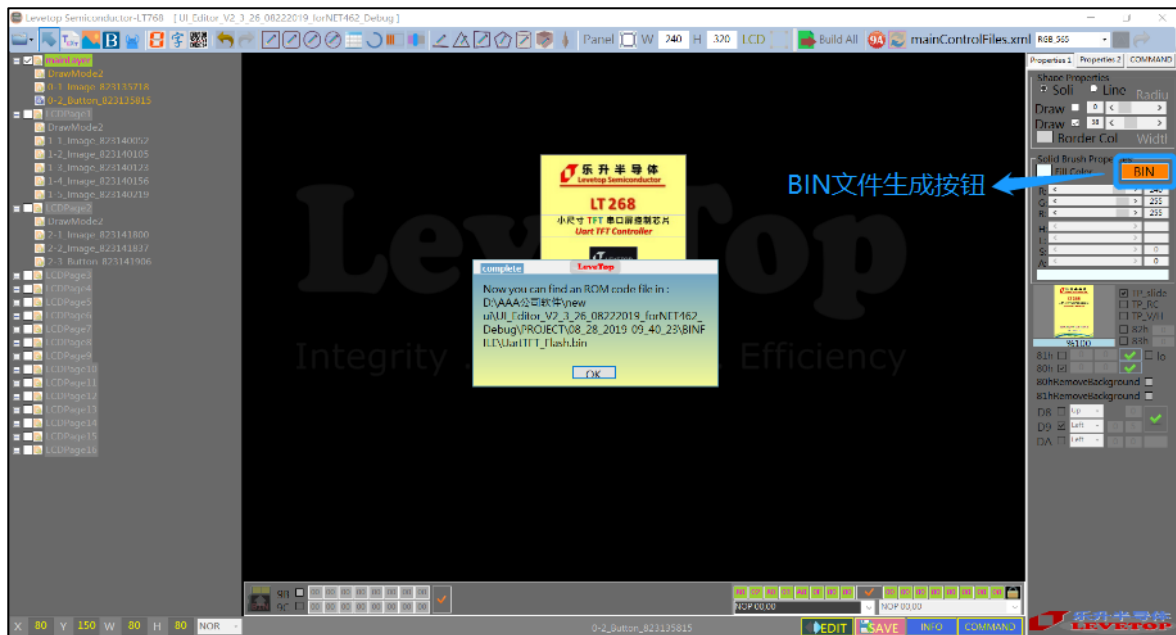


图 3-26：BIN 文件生成


若想打开刚刚生成 BIN 文件所在路径，可以点击  按钮，位置如下图：



图 3-27：BIN 文件生成

3.2 显示内容的设定

3.2.1 单张及多张图片(80h)

先点击图片添加按钮，在屏幕框中拖动，出现一个四角带点的白线框。这个白线框就是图片的大小。松开手后，在弹出的资源管理器中选中一张图片。图片的大小可以通过拉动白线框的大小来改变。如果想更换图片可以在图片被选中的前提下，双击图片来更改。如果图片不能被选中（没有出现四角白线框），可以单击项目栏中的项目来选中对应的图片对象。成功添加图片之后，可以看到右边操作栏中有 80h 指令的功能框。

显示单张图片：如果想在发送指令后单独显示一张图片，则可以不设置组号（默认为 0）。

显示多张图片：要是想要用一个指令显示多张图片，就设定相同组号（组号 ≥ 1 ），按确认键完成设定。软件会将相同组号的图片形成一个组合，组合的图片会一起显示。

如果图片是 PNG 的话可以勾选 PNG 背景框选项（80hRemoveBackground）来去掉 PNG 图片的背景。



图 3-28：设定显示多张图片

3.2.2 卷动及循环卷动图片(D8h, D9h)

卷动图片(D8h): 效果是从某一个地方按时间间隔逐渐显示一张图片。在添加图片之后, 勾选 D8h 操作栏中的小方格, 对图片开启卷动功能。之后可以选择图片卷动的方向。有四个选项, 分别是: 向上卷动、向下卷动、向左卷动、向右卷动。选择一个自己需要的方向后, 填写卷动的出现速度 (单位为 10ms), 最后按确认按钮。

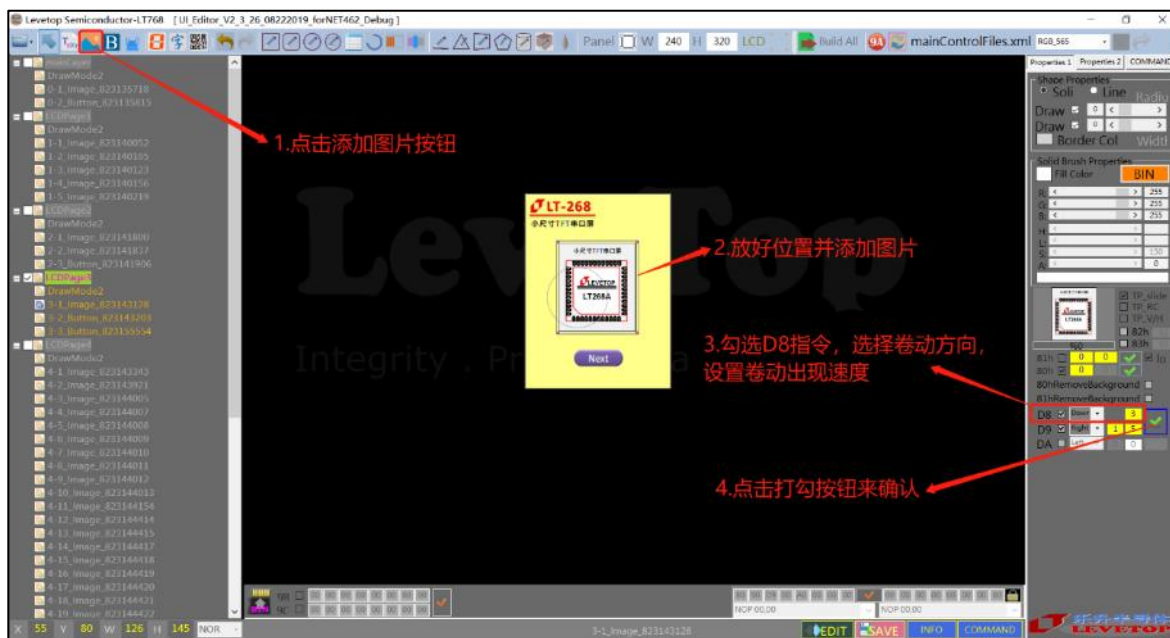


图 3-29: 设定卷动出现图片

循环卷动图片(D9h): 是在卷动图片的基础上, 可以实现一张图片维持卷动的动画, 或者多张图片持续的顺序卷动。具体的操作方法和上面的设定卷动出现图片的基本类似。

单张图片循环卷动, 则要该张图片的 D9h 指令操作栏中设定卷动方向、组号、卷动速度。
(此时, 单张图片独占一个组号)

多张图片循环卷动, 则要在每张图片的 D9h 指令操作栏中设定卷动方向、相同组号、卷动速度, 其中卷动方向、时间间隔可以设置不同值。如图所示:



图 3-30：设定第一张滚动出现图片



图 3-31：设定第二张滚动出现图片

3.2.3 循环显示重叠图片(81h)

在添加了图片之后，可以设定循环显示图片的组号和显示之间的时间间隔。其中时间间隔是以 10ms 为单位。假如，图片 A、B、C 要以 100ms 时间间隔循环显示，需要设置 3 张图片为同一组号，时间间隔只需要在第一张图片 A 处设置即可。

详细设置，在第一张图片 A 的 81h 指令操作框处设定 1 和 100，按下确定键；在图片 B 的 81h 指令操作框处设定 1 和 0，按下确定键；在图片 C 的 81h 指令操作框处设定 1 和 0，按下确定键。图片 A、B、C 就会形成一个组合 1，在接收到对应的指令后，循环显示。其中如果图片是 PNG 的话可以勾选去掉 PNG 背景框 (81hRemoveBackground) 来去掉 PNG 图片的背景。



图 3-32：设定循环显示重叠图片

3.2.4 显示 GIF 动画图片(88h)

按 GIF 添加按钮，从资源管理器中选一张 GIF 图片。在右边的操作栏中可以找到 GIF 演示窗。在小演示窗口的右下角的数据框是 GIF 图播放间隔的设定框（以 10ms 为单位）。此时 **81h** 指令的时间间隔、确认按钮变为了 **GIF** 的时间间隔、确认按钮。


设定时间间隔，再选择是要一直循环还是循环一次，按确认按钮，即可完成设定。注意，修改了时间间隔后，要按确认按钮来完成设定。（一直循环表示重复播放 GIF 动画，循环一次表示播放一遍 GIF 动画后静止在最开始的画面。）



图 3-33：设定显示 GIF 动画

3.2.5 图片式文字的显示(80h)

图片式显示的意思是用图片来显示文字，处理的对象是图片。是在 UI_Editor 上将文字生成为图片，在 TFT 串口屏中以图片形式显示文字。这里的文字是指电脑上可以输入的一切文字，包括中文、英文、数字和符号。这种文字显示只适用于显示固定不变的文字。

点击 UI_Editor 中添加  文字的按钮，是在屏幕框中拉一个框，松开按键后会弹出一个设置框。在框中输入要显示的文字。可以根据需要选择心仪的字体和字号，可以改变字体和背景的颜色，还可以选择背景是否透明。注意选择了背景透明，则背景颜色设置失效。确认设置后，要调节显示框的大小使全部文字得以显示。BackColor 勾选代表不透明，不勾选代表透明。

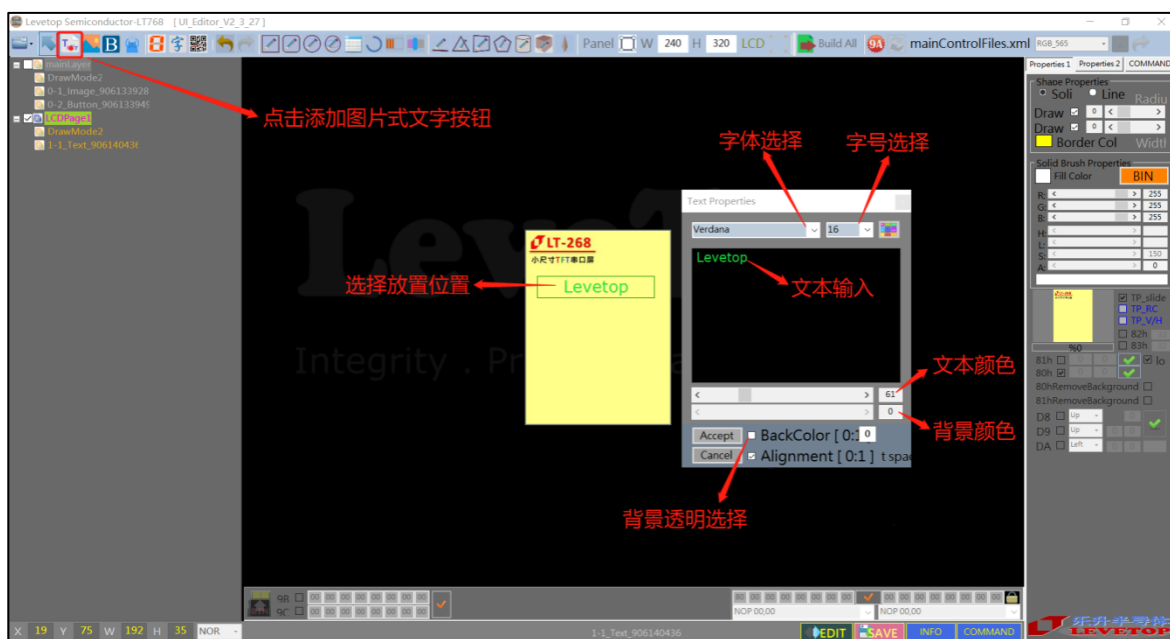



图 3-34：图片式文字的显示

3.2.6 图片式数字的显示(90h)

图片式的数字显示是指利用提前制作好的数字图片，根据串口接受到的数据，分析得到应该显示的数字。所以这种数字显示具有多字体和动态显示的优点。

点击 UI_Editor 中的  按钮。在屏幕框中拉一个框，松开按键会弹出一个设置框。这个设置框和上面的提到的设置框有不同。这个设置框中的文字不能改变，同时可以设置数字的对齐方式。Alignment 勾选代表左对齐，不勾选代表右对齐。BackColor 勾选代表背景不透明，不勾选代表背景透明。

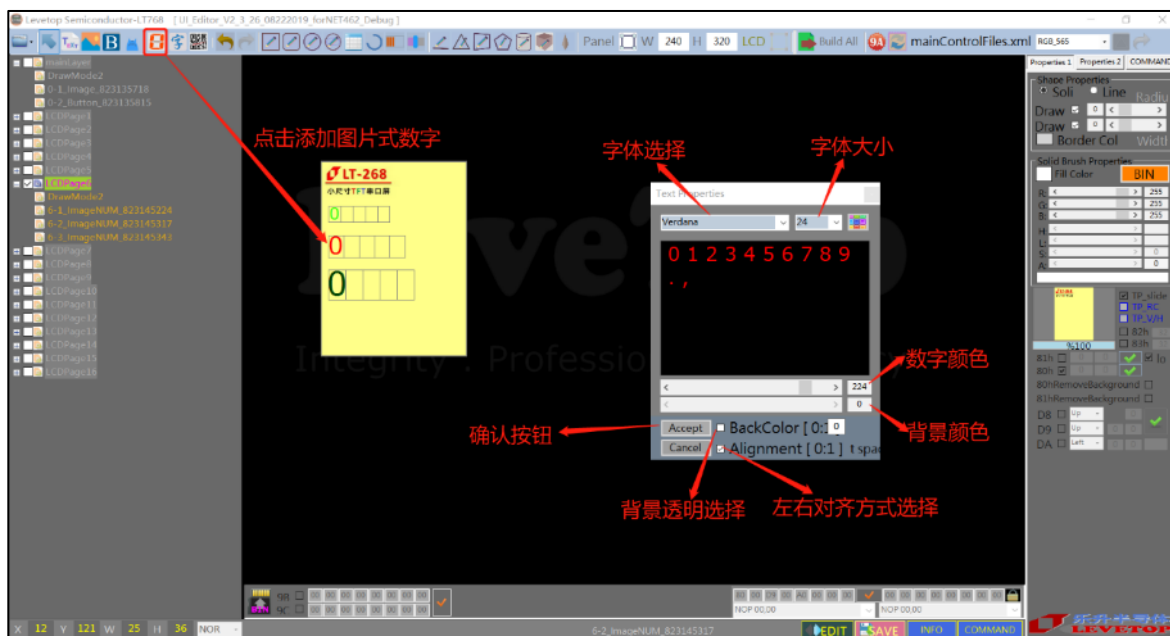


图 3-35：图片式的数字显示设置窗口

确认设置后，要调整文字框到刚好显示一个数字。如下图所示，每一个小框代表一个数字所占用的面积大小，只需要调整第一个数字框的大小，其他数字框也会跟着改变。在生成 BIN 文件后，用串口和 TFT 串口屏通信的时发送要显示的数字就可以在 TFT 串口屏屏幕中显示出来。

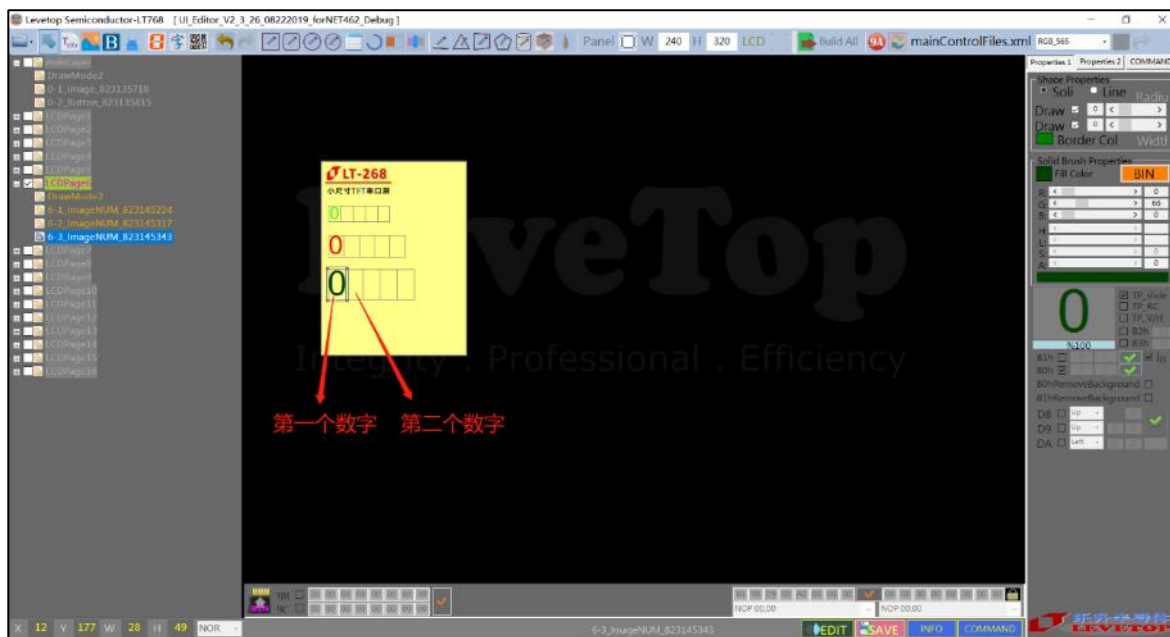


图 3-36：调整图片式数字显示的字体框

3.2.7 使用字库显示文字(C0h~C3h)

动态的中文、英文数字显示是 LT268B 利用内部或者 Flash 里的字库直接显示。其中 C0-C1 字库可以显示数字、英文、中文。C2-C3 字库仅可以显示中文。

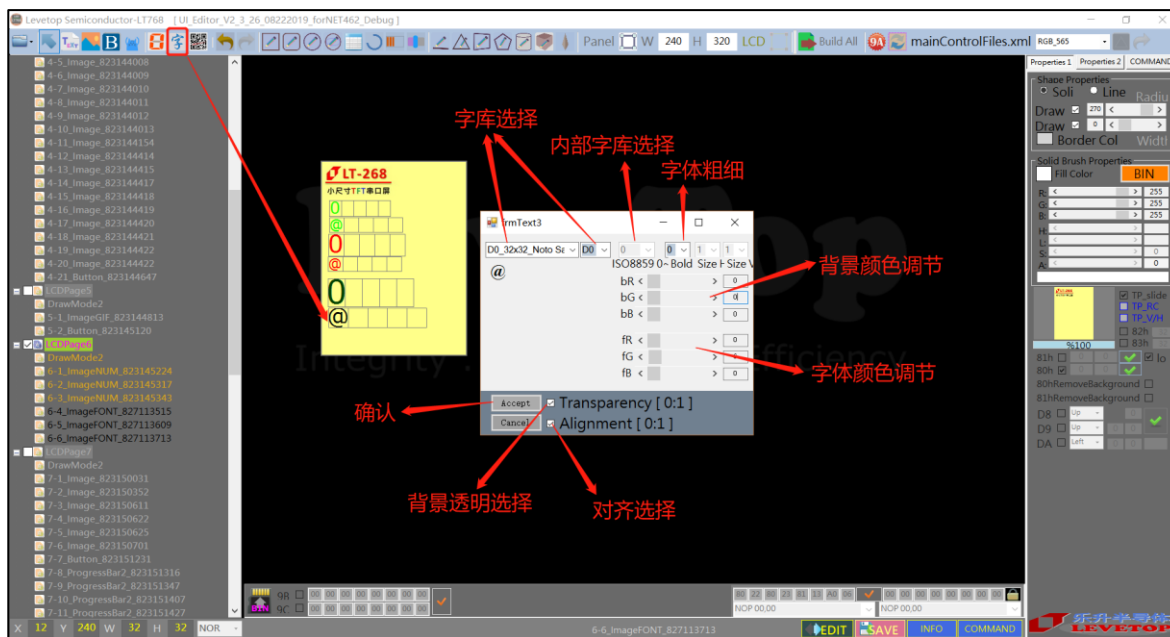


图 3-37：字库显示文字设置

点击文字(字库)添加按钮，点击一下屏幕框，会弹出设置框，框中有相关的各种设置，包括背景颜色、字体颜色等。注意：C0 到 C3 的字库不可以加粗字体。

背景透明的设置则和前面的不同，Transparency 被勾选了表示背景透明，不勾选代表背景不透明。Alignment 被勾选代表对齐，不勾选代表不对齐。

设置完成后，文字框中小框的大小不能被改变。在生成 BIN 文件后，用串口和 TFT 串口屏通信时发送要显示的文字就可以在 TFT 串口屏屏幕上显示出来。

3.2.8 显示控件功能的图片(A0h)

可以通过添加按钮控件来实现按键功能，即在屏幕上显示一个按钮图片，触摸到这个图片的时候，执行一系列动作，如显示图片、显示 GIF 等。

先点击按键添加按钮，在屏幕框中点击、拖动生成白线框，在资源管理器中选一张图片作为按钮图片（必须添加图片）。选中按钮控件，点击第一组指令输入处（两格组成一个指令），在下面的指令可选列表中选择要执行的指令。设置好后点击确认设置按钮即可。指令可选列表是由当前工程中其他已生成动作指令组成的。例如在设定按键功能前，已经设定了显示一张图片和一张 GIF 图的指令。则指令列表中会有这两个指令。特别说明，不同按键的设定方法一样。如果控件图片是 32 位 PNG 格式，希望去掉背景，则可以勾选 80h 指令后的去背景框。



图 3-38：设定显示控件功能的图片



图 3-39：设定控件按下后要执行的指令



图 3-40：确认设定显示控件的指令

3.2.9 开机画面(9Ah)

开机画面设定是用来选择那些控件在开机时被预先执行，而不需要主控端先发送指令，以下示例假设构建的控件已经都存在：

1. 先点选左上角 “mainLayer” ，此时会出现 8 组在开机时可预先选择要执行的指令。



图 3-41：点选左上角 “mainLay

2. 只要点选任一组数字，即会显示黄色框背景与红色字体，再根据如下图所示位置点选并自动上拉展开 Command 群组，此时即可选择每一组 Command 的指令组合。



图 3-42：选择每一组 Command 的指令组合

3. 开机画面设定如下如所选的指令 (UserCommand)

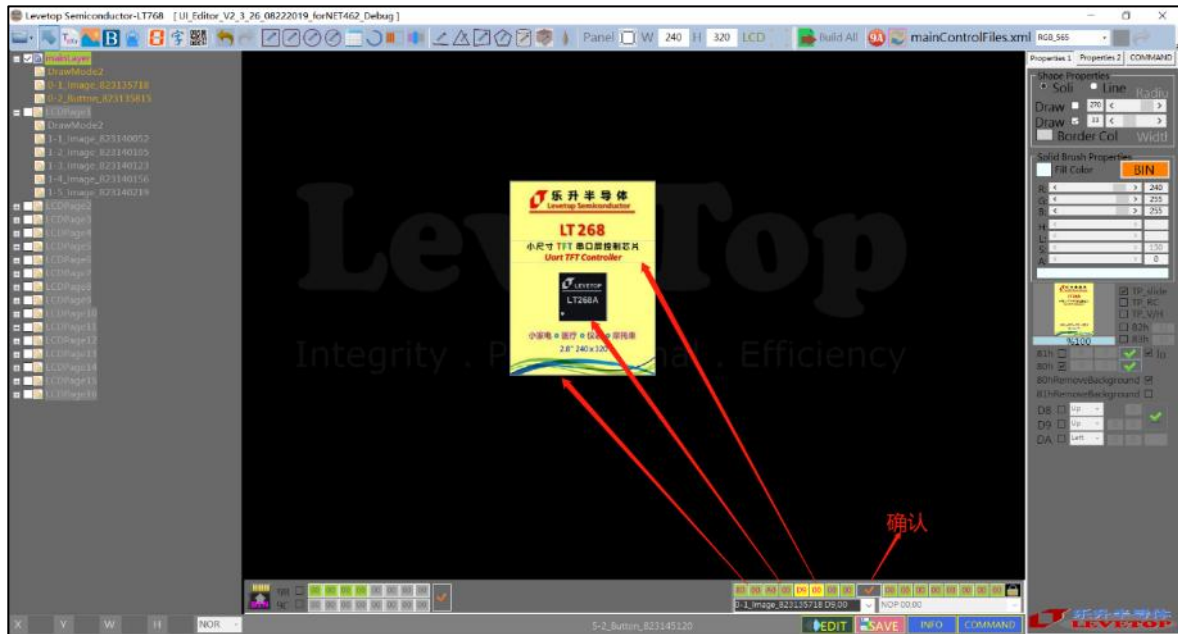


图 3-43：开机画面所选的指令

4. 选择完毕后请再点选确认图标。

9A 指令中除了 9A 00 为开机指令外，也可以用来设定执行多组的指令，例如新增加 9A 01、9A 02 等等之后的使用者指令，这些指令码主要是让使用者可以直接从串口输出 9A 01、9A 02 等等指令，控制 UI_Editor 编辑时预先设定要呼叫的其他指令，范例如下图所示（实际于 LCD 并不会显示 9A 图标）：



图 3-44：以 9A 指令来设定执行多组

3.2.10 进度条图(B0h)

此进度条功能与指针图类似，由主控端传来信息改变进度条显示的长短：



图 3-45：点选进度条图标

1. 点选进度条图标，并在屏幕任意位置拖拉移动显示出进度条，此时可针对功能需求透过调色盘调整进度条背景与进度条移动时的颜色。



图 3-46：设定进度条背景与进度条移动时的颜色

2. 若要显示进度背景图，要勾选进度条背景透明，再添加进度条背景图片，如下图所示：



图 3-47：设定进度条背景图片

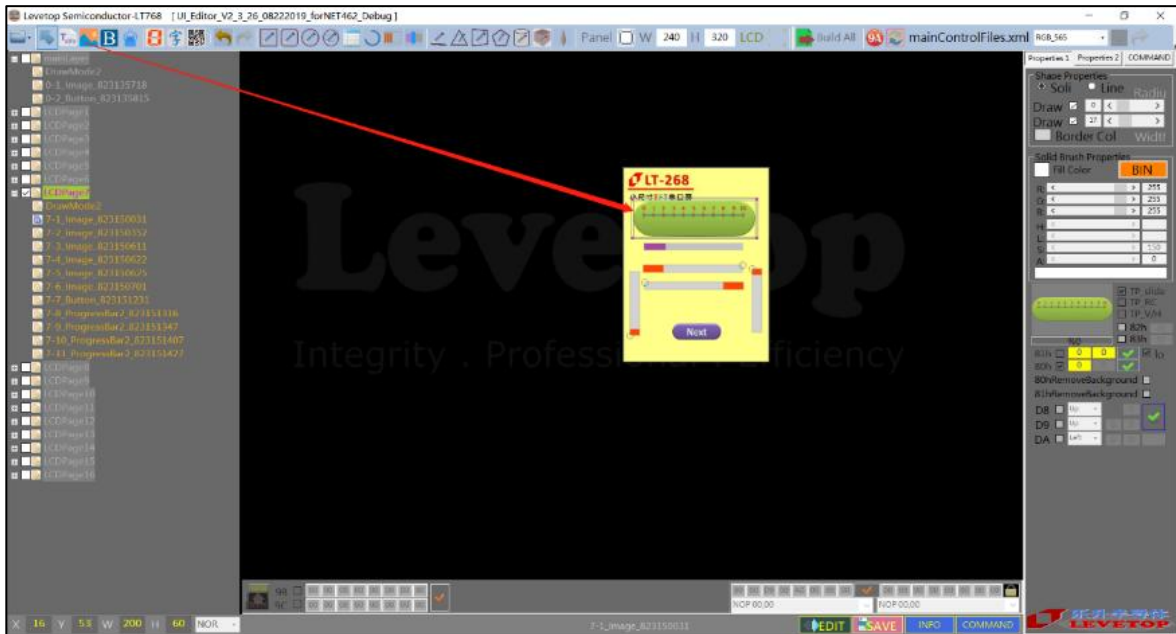


图 3-48：设定背景图片

3. 请针对功能需求将进度条与背景图片进行对齐调整，如下图所示：



图 3-49：调整进度条背景图片显示位置

4. 选择进度条背景图片 Command, 请先点选进度条以确保是要设定进度条背景图片模式,

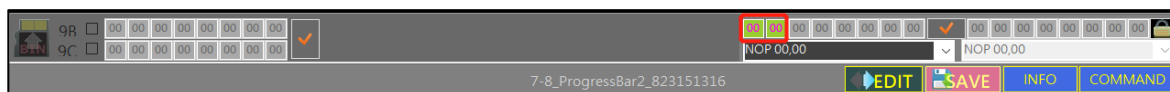


图 3-50: 选择进度条背景图片 Command

5. 点选第一组数据 (00 00 任意点选一个, 数据背景变成黄色)

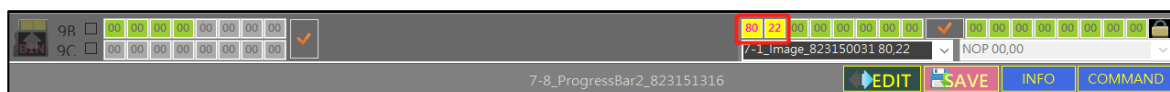


图 3-51: 点选进度条背景图片模式数据

6. 接着点选屏幕下方 Command 选择窗口, 如下图所示, 再展开 Command 列表:

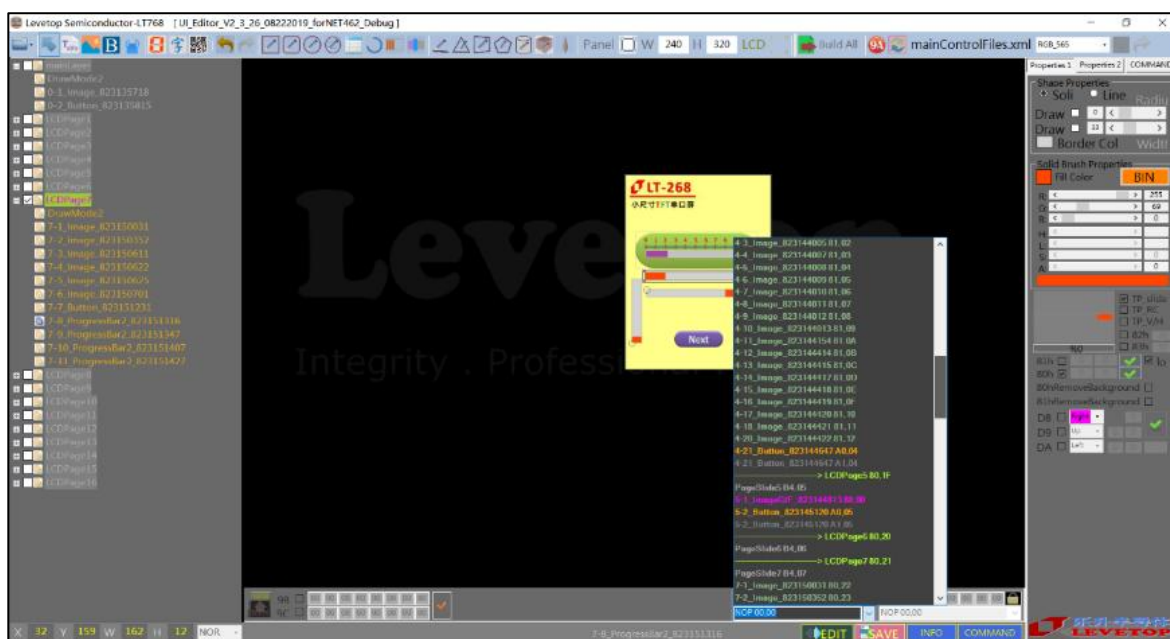


图 3-52: 选择背景图片

7. 并选择上一步骤图片的编号即可，最后要记得再点选确定图标。

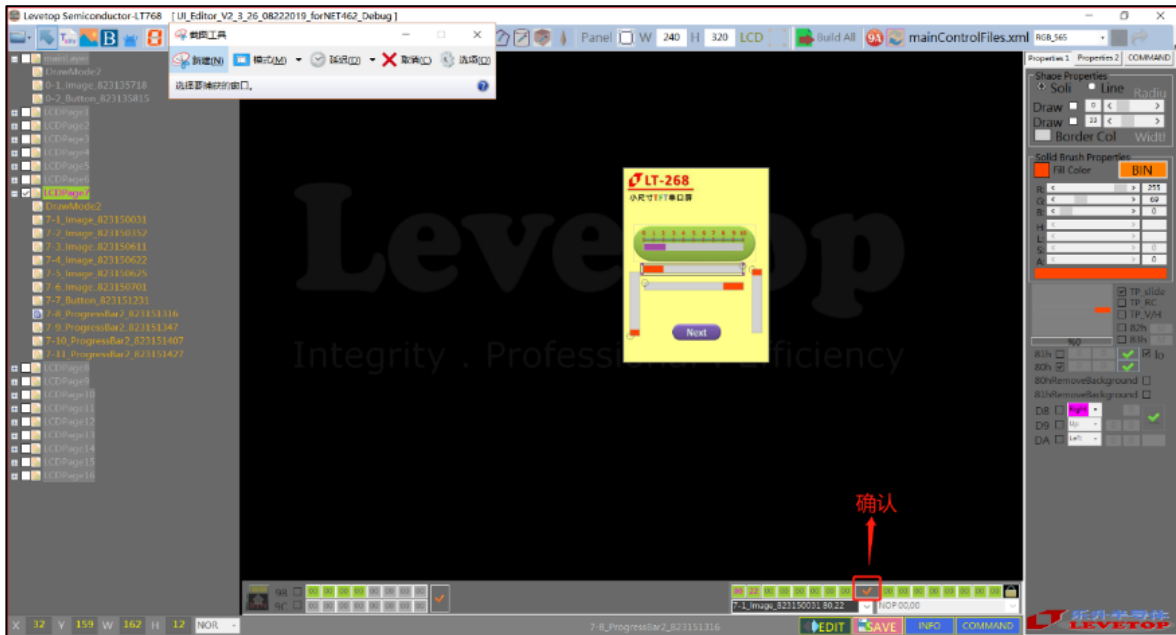


图 3-53：点选确定图标

8. 进度条指令[B0h]可设定显示是上、下、左、右的是显示方式，如下图所示参考：

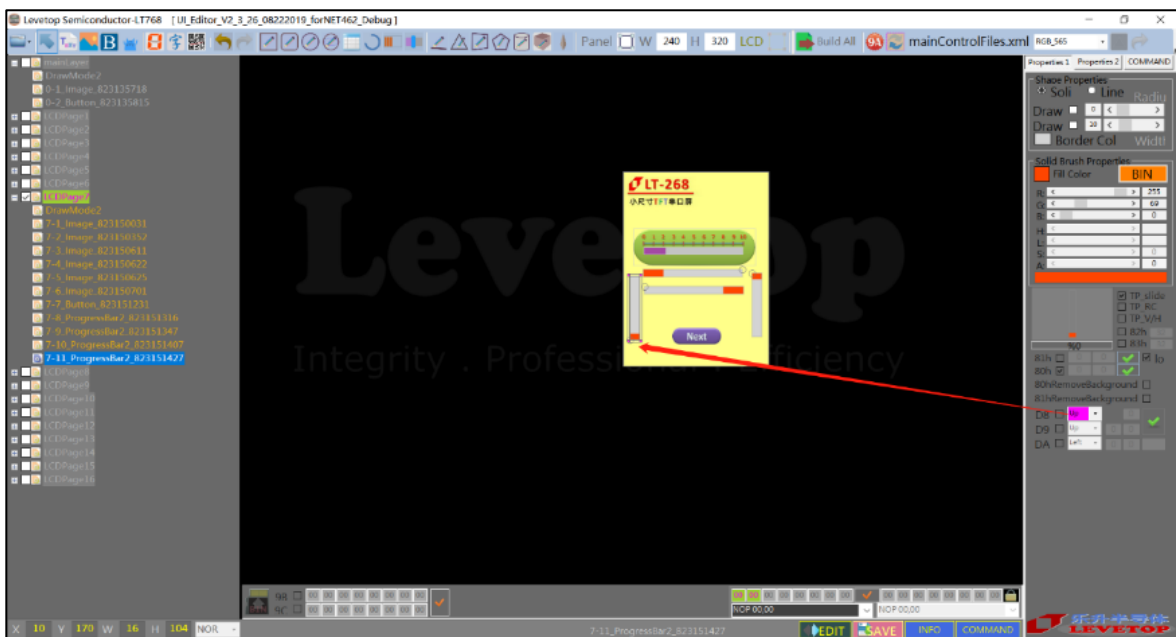


图 3-54：进度条指令范例

3.2.11 画直线(E0h)

画直线: 点击画直线按钮, 在屏幕框中点击一下, 即可产生一条直线。拖动直线上的两个端点, 可以改变直线的长度和方向。改变线条粗细设置框里的数值可以改变线条粗细。线条的颜色可以通过选择颜色按钮里的颜色来改变。

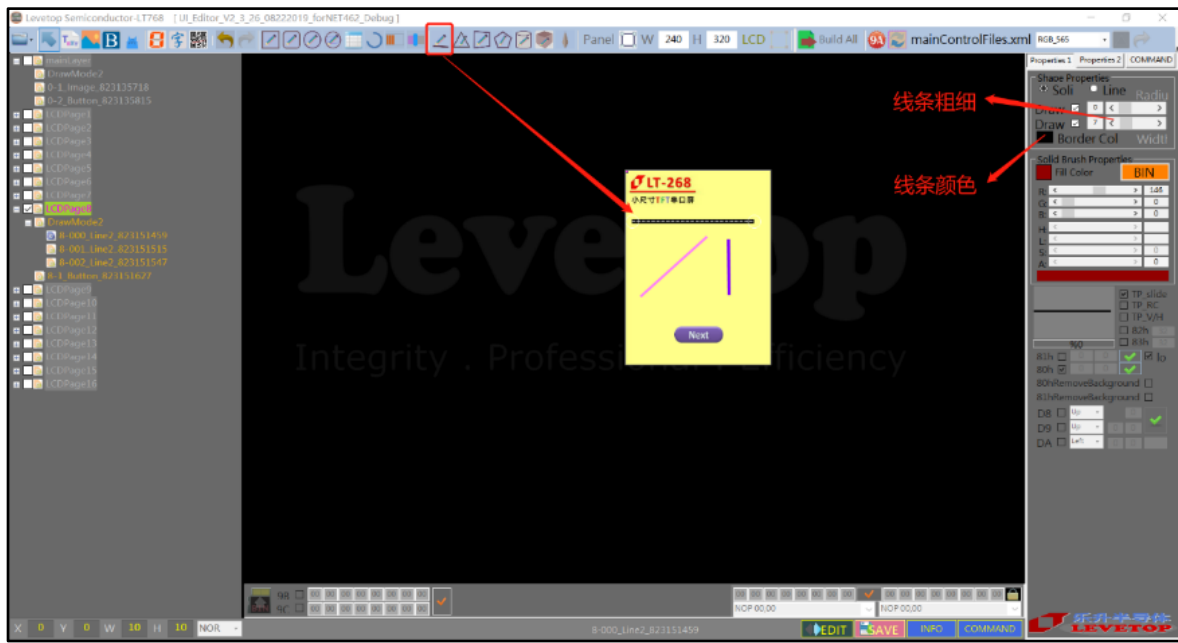


图 3-55: 画直线

3.2.12 基本几何绘图

目前支持的几何绘图功能有 7 种，类似的功能可以分为 3 类，如下表所示。

表 3-1：几何绘图功能

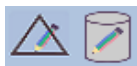
几何绘图功能	空心（指令）	实心（指令）	带框实心（指令）
圆环	-	✓ (DCh)	-
圆	✓ (E1h)	✓ (E2h)	✓ (E3h)
椭圆	✓ (E4h)	✓ (E5h)	✓ (E6h)
矩形	✓ (E7h)	✓ (E8h)	✓ (E9h)
圆角矩形	✓ (EAh)	✓ (EBh)	✓ (ECh)
三角	✓ (EDh)	✓ (EEh)	✓ (EFh)
圆柱体	-	-	✓ (F4h)



添加画圆环（任意角度）



分别是画矩形、画圆角矩形、画圆、画椭圆



分别是画三角形、画圆柱体

画任意角度圆环：点击添加圆环按钮，在屏幕框上点击一下，会出现带调整框的圆环。改变调整框的大小可以改变圆环的半径。圆环的粗细和颜色通过改变相应的按钮处的内容即可。在主控端通过串口与 TFT 串口屏连接后，发送指令和圆环的起始角度与转动角度就可以在 TFT 串口屏的屏幕上显示相应角度的圆环。圆环的转动方向是顺时针的，以 12 点钟方向为 0 角度点。

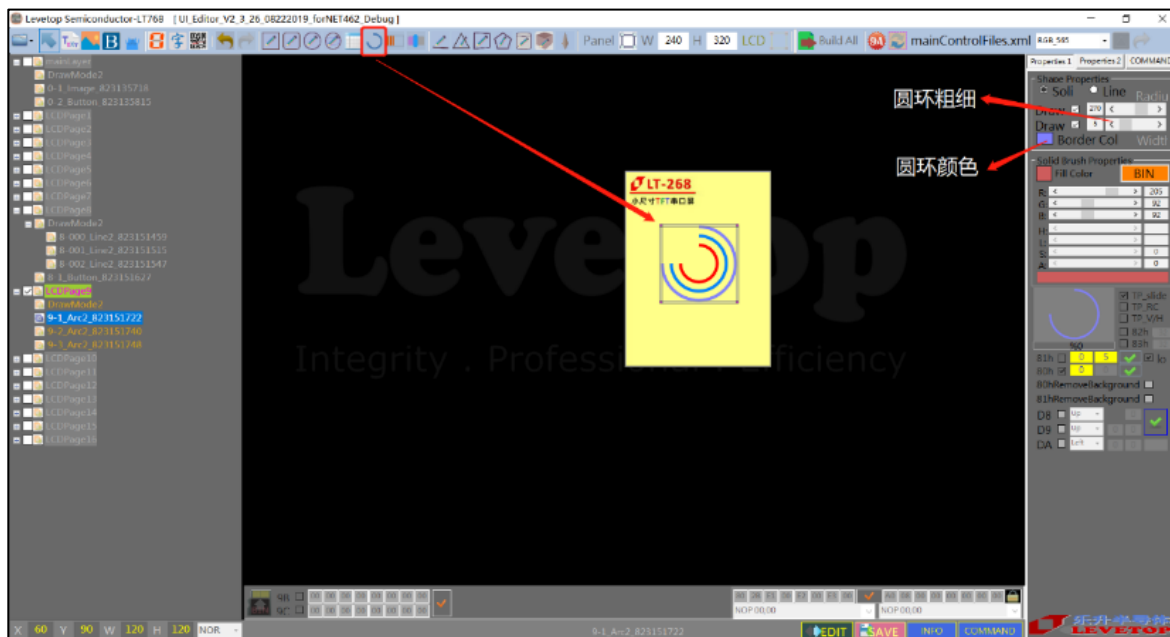


图 3-56：画圆环设置

画圆：点击画圆按钮，在屏幕框中点击并拉动操作框，即可出现圆形。调整外面的操作框可以改变圆的大小。其他的设置都是右边的 Properties1 设置栏中。

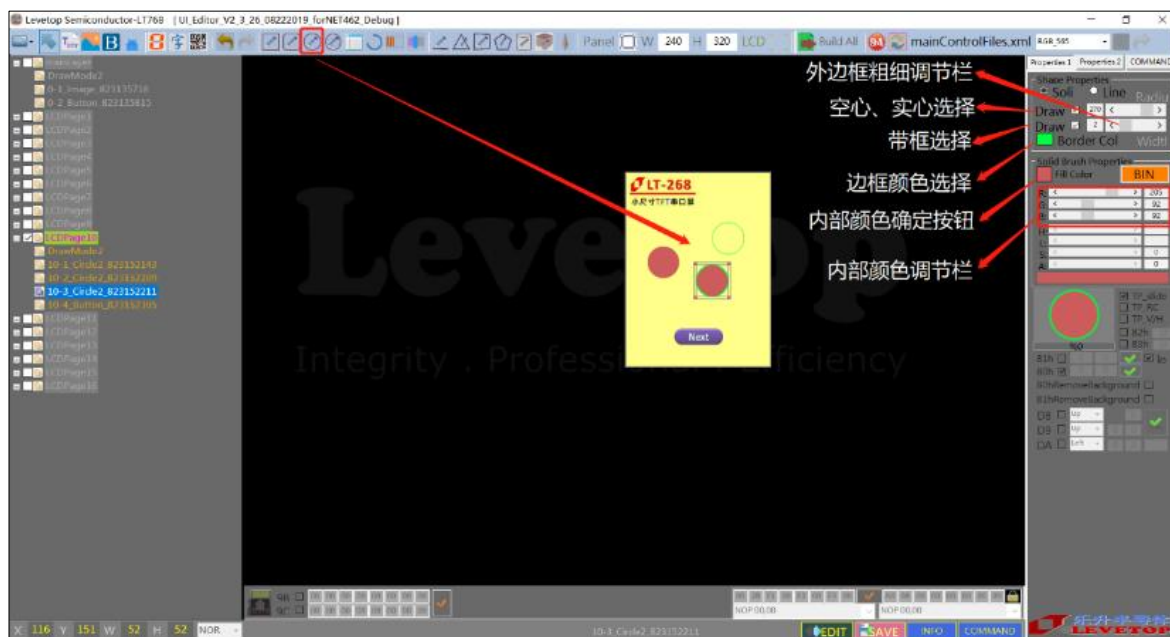


图 3-57：画圆设置

其中 Draw1 和 Draw2 的具体作用如下：

Draw1：空心、实心的选择框，Draw2 没有被勾选的前提下，勾选 Draw1 代表选择空心功能，不勾选 Draw1 代表选择实心功能。如果 Draw2 被勾选了，Draw1 将不受控。

Draw2：带框实心的选择框。勾选上则选择带框实心功能，不受 Draw1 控制。不勾选则选择空心、实心功能。只有在带框功能下才可以调节外边框的粗细。

要改变外边框的颜色，只需要点击外边框颜色选项，在其中选择一种心仪的颜色。要调整内部颜色，可以改变颜色调节框中 RGB 三色的数值，选一种喜欢的颜色，最后要按内部颜色确定按钮。

画椭圆：与画圆设置类似。

画矩形、圆角矩形：与画圆的设置类似。其中，设置圆角矩形时，圆角值的 2 倍不能超过圆角矩形的长或宽，否则无法正常显示。

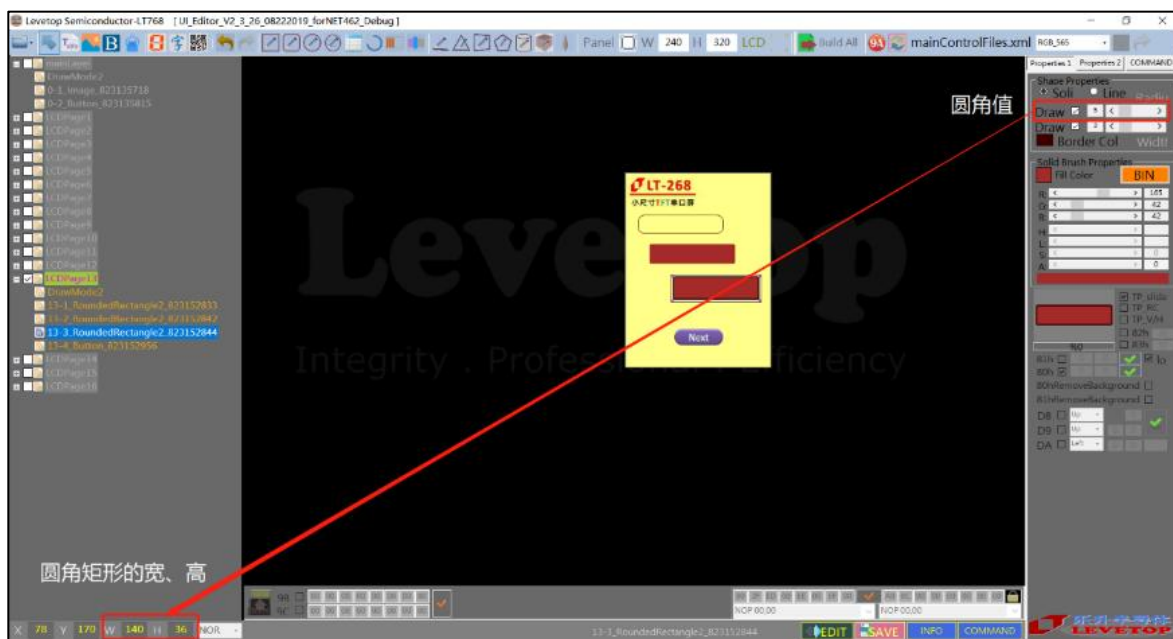


图 3-58：画圆角矩形设置

画三角形：与画圆柱体（见下面）是大体类似的，其中调整三角形的大小是通过拉动三角形的三个顶点来实现的。但三角形不支持外框粗细调节。

画圆柱体: 点击圆柱体添加按钮，在屏幕框内点击一下，就会出现一个圆柱体，改变圆柱体上的 3 个点可以改变圆柱体的形状。因为 LT268B 只支持画带外边框实心圆柱体，所以 Draw1 和 Draw2 都失效。边框颜色和内部颜色都和其他绘图功能的一样。若想移动圆柱体，可以在该图层选中圆柱体后，按住圆柱体上 3 个点组成的三角区域来拖动。

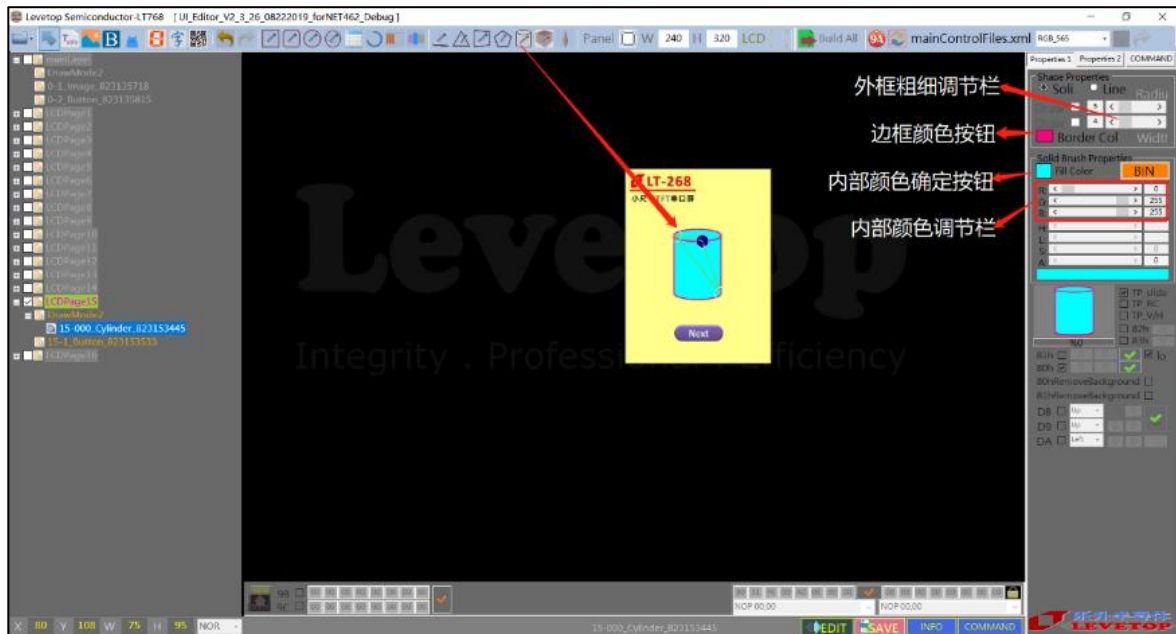


图 3-59: 画圆柱体设置

3.2.13 表格制作(F6h)

按表格添加按钮，在屏幕框内点击一下就会出现一个表格。表格的大小不能通过拉动边框来调整，必须使用右边 Properties2 栏中的 N-Col 和 N-Row 来调整。表格的每一个参数调整都可以在 Properties2 中进行设置。其中如果要更换项目栏的方向（横向、纵向），先选中表格改变 Properties2 中的 Mode 的勾选状态（勾选：横向、不勾选：纵向），再点一下表格就改变项目栏的方向。如果要改变项目栏和内容栏的底色，先选中表格调整对应颜色下面的 RGB 参数，得到自己想要的颜色后，点击一下 Fill color 前面的颜色选项即可。

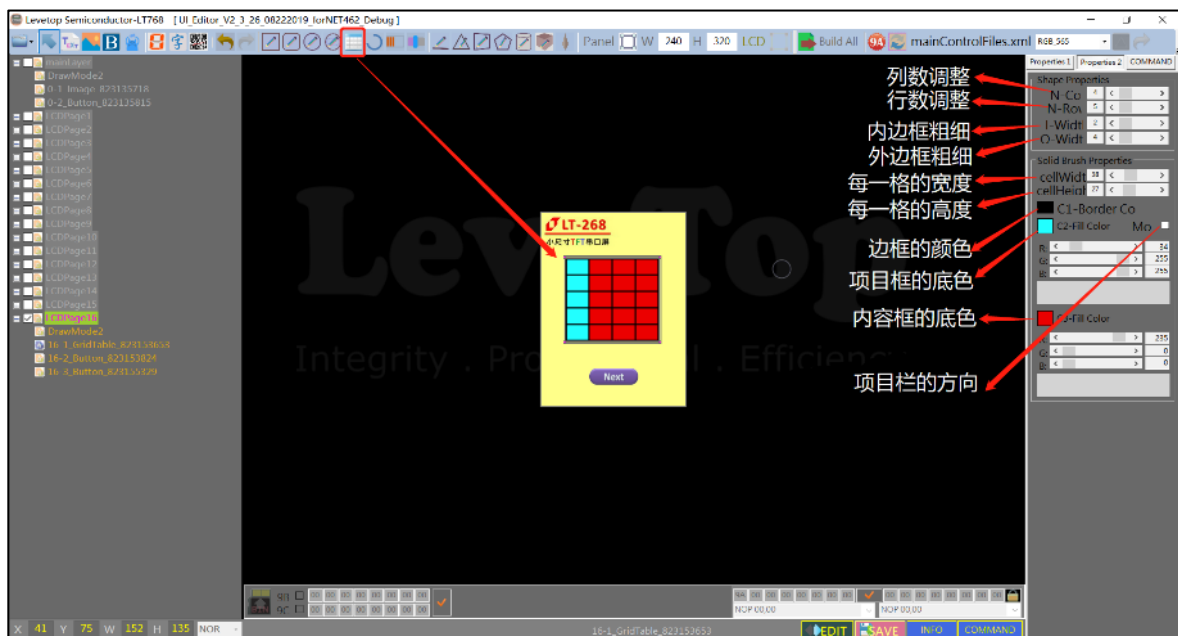


图 3-60：表格设置

3.3 其他功能

3.3.1 背光控制指令

可以在开机画面和按钮的指令中设定背光控制，只要在指令中选择 BA 指令。指令选项的最下面固定有 3 个 BA 指令，如下图所示：

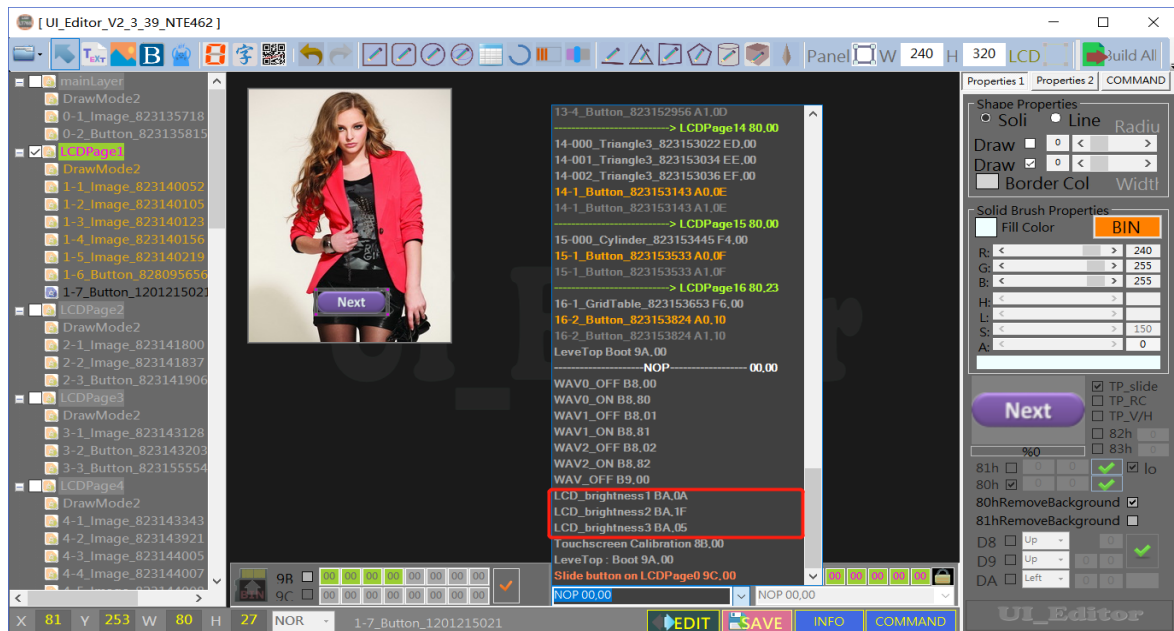


图 3-61：指令选项中 BA 指令

选择了 BA 指令中的一条，可以在 BA 指令处通过鼠标右键来呼出背光度选择框：

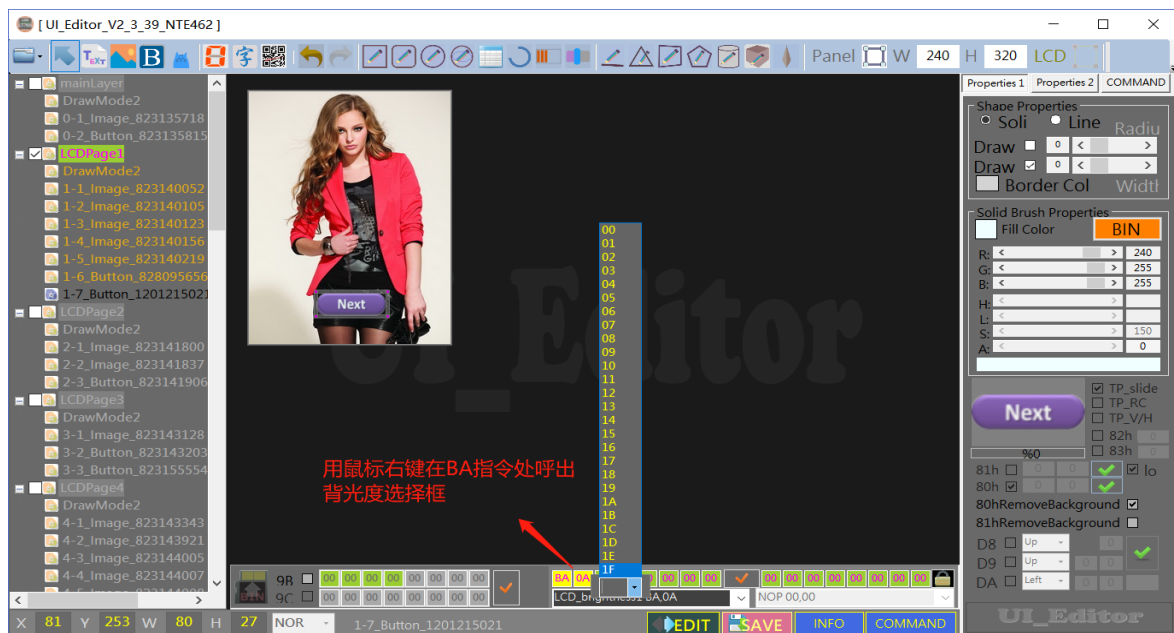


图 3-62：指令选项中 BA 指令

通过上位机直接发送 BA 指令改变背光。点击 Command，打开串口，连接到下位机。点击 BA 指令的 SEND 按钮。如果 BA 后面的数值是指背光的亮度级别。范围是 0x00-0x0F 共 16 个级别。可以通过慢双击 00 处，再输入亮度级别。按 SEND 键发送，屏幕亮度就会改变。

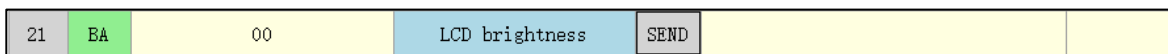


图 3-63: Command 发送背光控制指令

3.3.2 起始控制指令

当用 Command 发送指令 BCh 00 (00 代表 BCh 后面的数值) 给 TFT 串口屏，那么 TFT 串口屏的显示会被关闭 (Display Off)；发送指令 BCh 01 给 TFT 串口屏，那么 TFT 串口屏的显示会被开启 (Display On)。注意，显示屏 Off 后，TFT 串口屏将不接受任何绘图功能的指令，直到串口屏收到显示屏 On 指令。

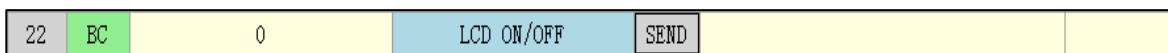


图 3-64: 起始控制指令

3.3.3 电阻屏控制指令

电阻式触控屏控制指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端系统或是主板的固定指令，这指令不需要在 UI_Editor 编程软件上设定。

表 3-2：电阻式触控屏控制指令

指令功能	指令码	序号	指令参数	指令说明
进行电阻屏校对	8Bh	--		进行电阻屏的 4 个角落校验点。

例如当 UART 串口传递控制指令 8Bh 给 TFT 串口屏，那么就会自动进入电阻屏的 4 个角落校验程序，使用者必须依照显示的角落点击完成电阻式触控屏校验程序。



图 3-65：电阻屏的 4 个角落校验

3.3.4 串口屏侦测指令

串口屏侦测指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端系统或是主板的固定指令，这指令也是不需要在 UI_Editor 编程软件上设定。

表 3-3：起始控制指令

指令功能	指令码	序 号	指令参数	指令说明
检查 TFT 串口屏	BEh	--		串口屏初始化完成或是处于 Ready 状态： 5Ah → Ready; 55h → Not Ready（忙状态）;
检查 TFT 串口屏版本	BFh	--		读取 LT268B 串口屏版本信息：MCU Code 版本(5Bytes) + 串口屏模块.(42Bytes)

当 UART 串口传递指令 BEh 给 TFT 串口屏，如果串口屏初始化完成或是处于 Ready 状态，串口屏会回应 5Ah 给主控端。如果串口屏是处于 Busy 忙状态，串口屏会回应 55h 给主控端。若是 TFT 串口屏与主控端的 Uart 通讯口未连接则不会响应任何信息。

串口指令 BFh 是用来读取 LT268B 的 TFT 串口屏版本，包括 5 个 Byte 的 LT268B 程序码版本，及 42 个 Bytes 的串口屏模块信息。当 UART 串口传递指令 BFh 给 TFT 串口屏，串口屏则依序送出这些信息给主控端 MCU。详细的版本信息可见第 4.2.18 节中的表 4-23。

3.3.5 字库文件的说明

字库文件存放于UI_Editor_V1.0\FONT路径下。其中fontList.txt文件的内容是所有字库文件的名称按排列顺序编写，如下图：

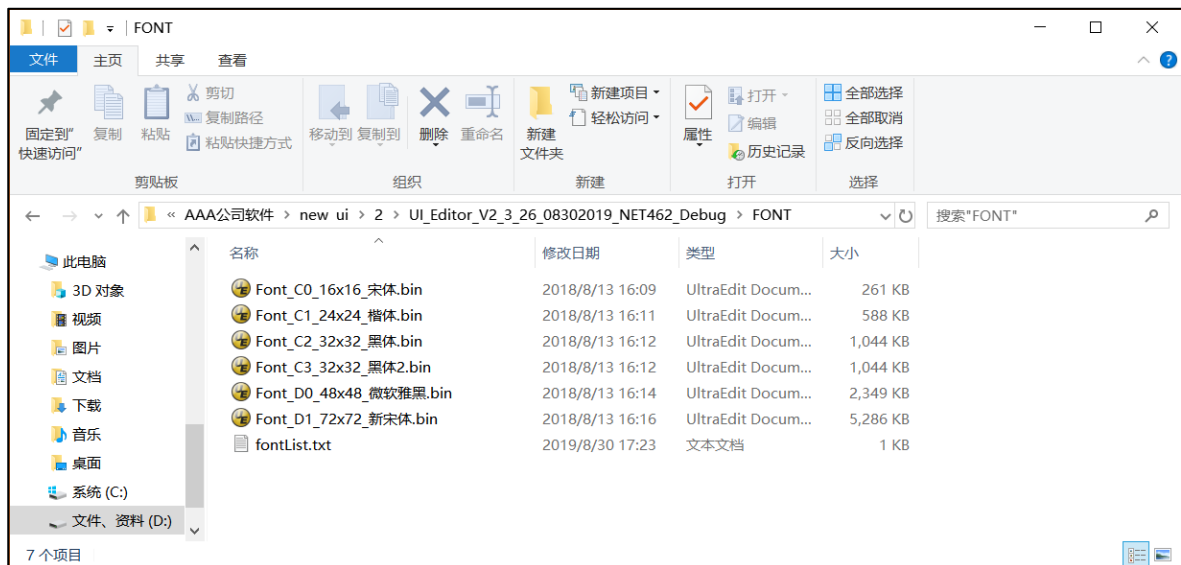


图 3-66: fontList.txt 文件

3.4 串口发送指令

设置 TFT 串口屏中屏幕的参数和主控端与 TFT 串口屏通信的参数。通过点击 UI_Editor 界面中的 INFO 按钮来打开参数设置。设置完后，点击 INI SAVE 按钮确定设置。

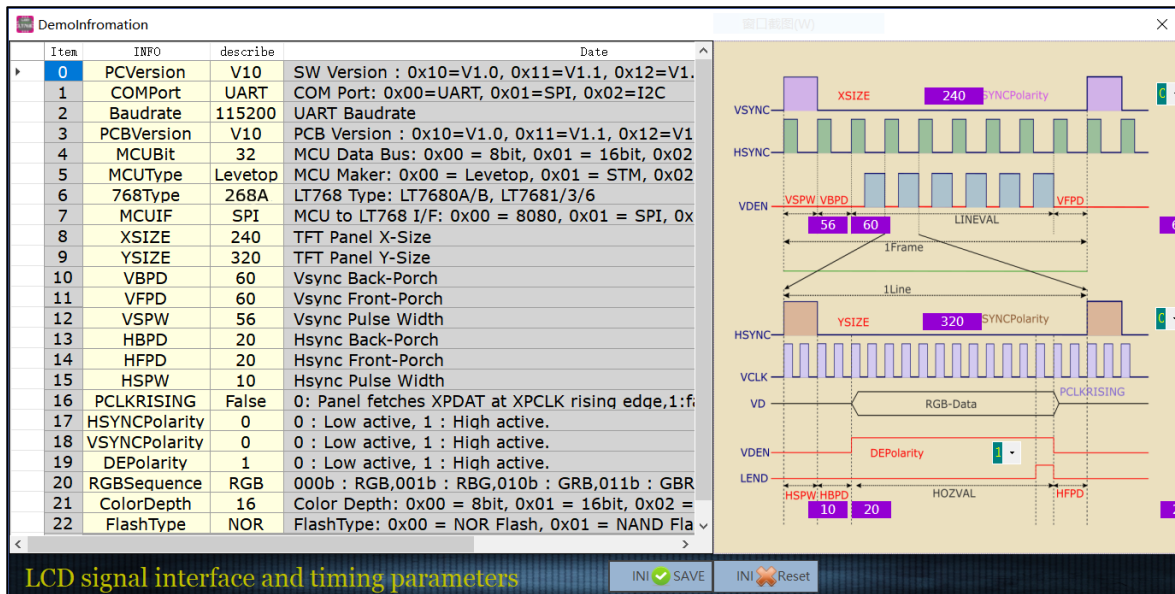


图 3-67: 配置参数

设置好 INFO 中的参数后，需要通过工程编译来完成设定。



图 3-68: 工程编译

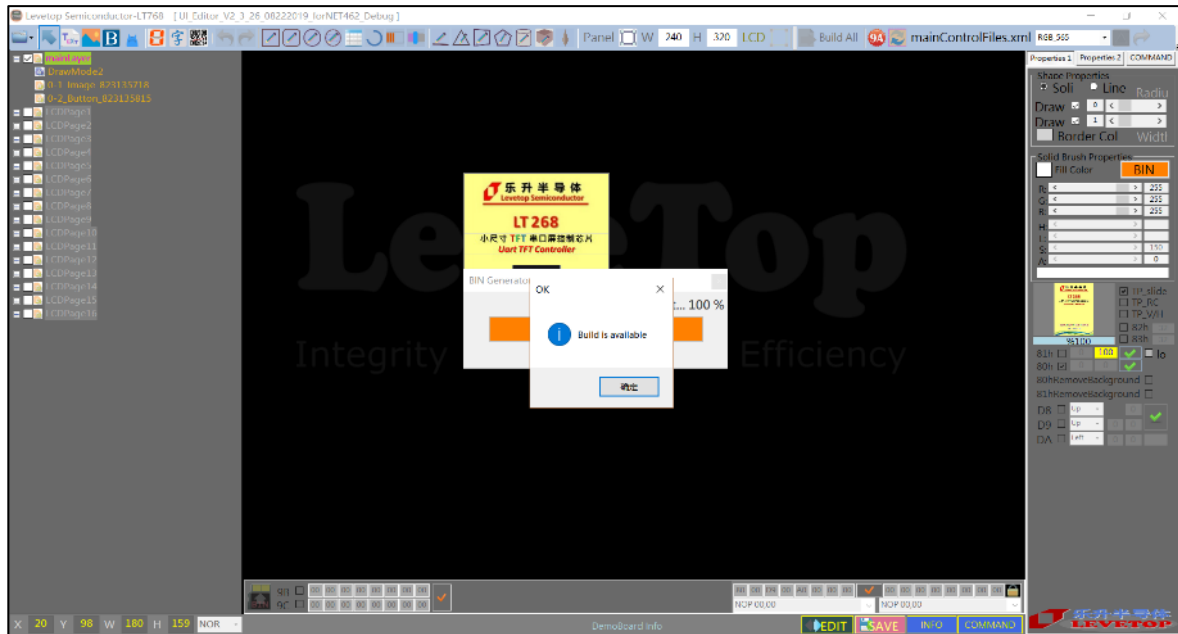


图 3-69：工程编译完成

编译完成后，然后按“BIN”按钮生成 BIN 文件，并被保存在 BINFILE 文件夹中，其中所有工程档案都同时被保存。

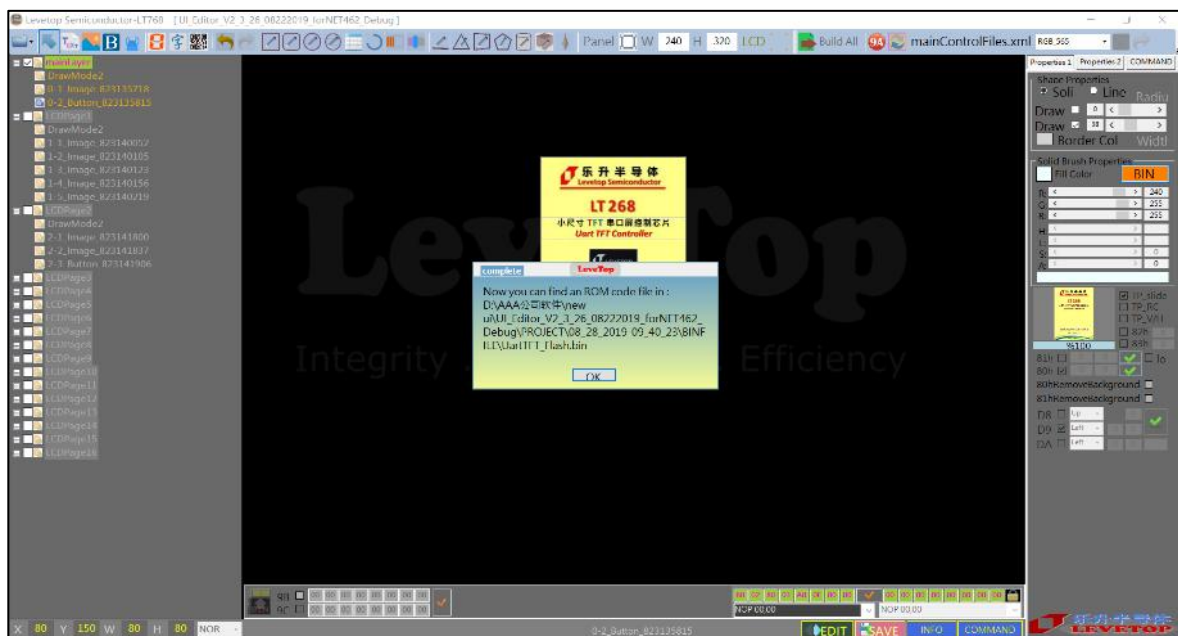


图 3-70：BIN 档产生

通过烧录工具将生成的 BIN 文件烧录到 TFT 串口屏中，连接 TFT 串口屏和主控端。点击 Command 按钮，弹出指令发送界面。设置串口波特率，选择串口号，打开串口，就可以发送指令。如果是要带数值的指令，Value 框中会有默认数值，可以改变它来实现其他的效果。按 SEND 按钮即可发送指令给 TFT 串口屏。

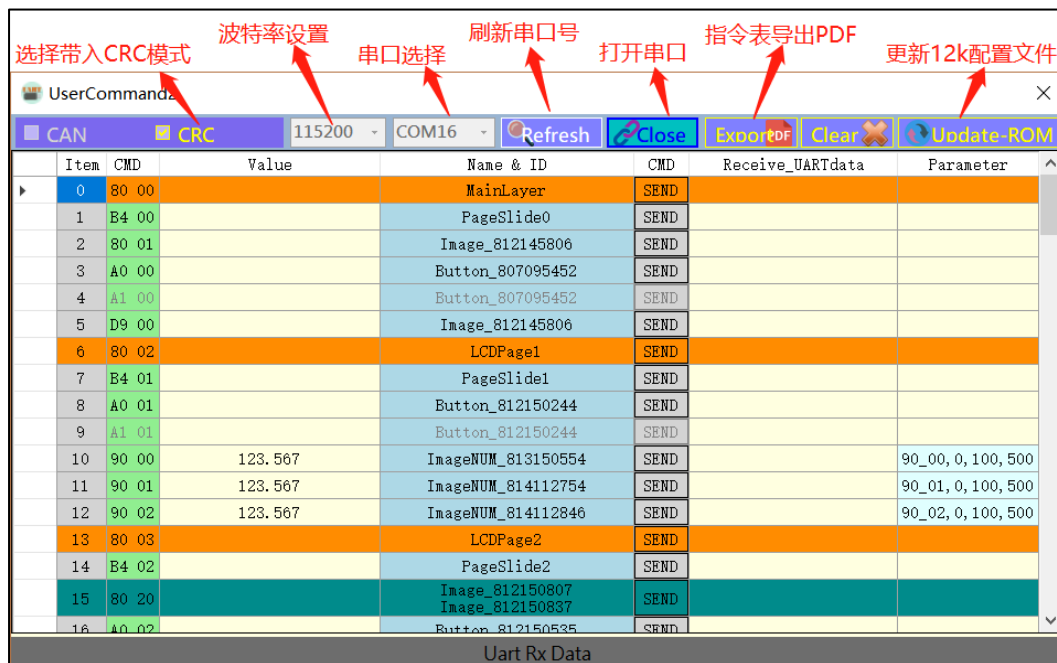


图 3-71: Command 发送指令给 TFT 串口屏

当开发者用 UI_Editor 将产品的 UI 接口规划及验证后，可以导出整个 UI 所运用到的指令表，主控端依据这些指令在其 MCU 程序上植入这些指令格式，在想要显示图片或是要求 TFT 串口屏做出动作时只要送出对映的指令及 CRC 码即可，同时主控端可以依据 TFT 串口屏的响应判断是否被正确执行。下图 3-72 为设计完成后导出的指令表。

LT268B : LCD Serial Communication Command Reference						
Author : LeveTop						
Run Date : 2019/9/6						
Item	CMD	Value	Name & ID	CMD	Receive_UARTdata	Parameter
0	80 00		MainLayer	SEND		
1	B4 00		PageSlide0	SEND		
2	80 01		0-1 Image 823135718	SEND		
3	D9 00		0-1 Image 823135718	SEND		
4	A0 00		0-2 Button 823135815	SEND		
5	A1 00		0-2 Button 823135815	SEND		
6	80 00		LCDPage1	SEND		
7	B4 01		PageSlide1	SEND		
8	80 02		1-1 Image 823140052	SEND		
9	80 03		1-2 Image 823140105	SEND		
10	80 04		1-3 Image 823140123	SEND		
11	80 05		1-4 Image 823140156	SEND		
12	80 06		1-5 Image 823140219	SEND		
13	A0 01		1-6 Button 828095656	SEND		
14	A1 01		1-6 Button 828095656	SEND		
15	80 00		LCDPage2	SEND		
16	B4 02		PageSlide2	SEND		
17	D9 01		2-1 Image 823141800	SEND		
18	D9 02		2-2 Image 823141837	SEND		
19	A0 02		2-3 Button 823141906	SEND		
20	A1 02		2-3 Button 823141906	SEND		
21	80 00		LCDPage3	SEND		
22	B4 03		PageSlide3	SEND		
23	80 07		3-1 Image 823143128	SEND		
24	D8 00		3-1 Image 823143128	SEND		
25	D9 03		3-1 Image 823143128	SEND		
26	A0 03		3-2 Button 823143203	SEND		
27	A1 03		3-2 Button 823143203	SEND		
28	A0 04		3-3 Button 823155554	SEND		
29	A1 04		3-3 Button 823155554	SEND		
30	80 08		LCDPage4	SEND		
31	B4 04		PageSlide4	SEND		

图 3-72：指令表导出 PDF 格式

由图 3-67 举例来说，主控端 MCU 程序送出 80h、00h、1Bh(CRC1)、98h(CRC2) 指令后，TFT 串口屏就会显示第一张 MainLayer 图片：



图 3-73：显示第一张 MainLyer

主控 MCU 程序送出 80h、31h、1Bh(CRC1)、98h(CRC2) 指令后，TFT 串口屏就会显示同一组号的两张图片：



图 3-74：同时显示两张图片

主控 MCU 程序送出 81h、00h、28h(CRC1)、A9h(CRC2) 指令后，TFT 串口屏就会循环显示同一组号的图片：



图 3-75：循环显示图片

下表是 LT268B 用 UI_Editor 设计后所能支持的指令总表，如果 UI_Editor 设计时没用到的功能将不会在导出的 PDF 内出现。

表 3-4: 指令总表

指令码	序 号	指 令 参 数	指 令 功 能
80h	nn		显示图片
81h	nn		循环显示重迭图片
84h	nn		取消循环显示重迭图片
88h	nn		显示 GIF 图片
89h	nn		取消显示 GIF 图片
8Bh			进行电阻屏校验
90h	nn	ddd.d	显示图片式的数字
91h	nn	ddd.d	显示图片式的数字
98h	nn	String	显示二维码
9Ah	nn		执行多组的指令，当 nn=00 时为执行开机指令
A0h	nn		显示控件图片
A1h	nn		取消显示控件图片及功能
A2h	nn		设置虚拟控件
A3h	nn		取消显虚拟控件及功能
B0h	nn	Value(2Bytes)	显示进度条指标图
B8h		REP(Bit7) + WAV	播放 Wav 檔
B9h			停止播放 Wav 檔
BAh		BL (00~0Fh)	调整背光亮度
BCh		00 or 01	显示 On/Off
BEh			检查 TFT 串口屏
BFh			版本侦测
C0h	nn	String	显示字库-1 文字
C1h	nn	String	显示字库-2 文字
C2h	nn	String	显示字库-3 文字
C3h	nn	String	显示字库-4 文字

表 3-4: 指令总表 (续)

指令码	序 号	指 令 参 数	指 令 功 能
D8h	nn		显示卷动出现图片
D9h	nn		显示循环卷动图片
DBh	nn		取消循环卷动图片
DCh	nn	S_Angle, A_Angle	显示环形指标图
E0h	nn		显示一条直线
E1h	nn		显示一空心圆形
E2h	nn		显示一实心圆形
E3h	nn		显示一带框实心圆形
E4h	nn		显示一空心椭圆形
E5h	nn		显示一实心椭圆形
E6h	nn		显示一带框实心椭圆形
E7h	nn		显示一空心矩形
E8h	nn		显示一实心矩形
E9h	nn		显示一带框实心矩形
EAh	nn		显示一空心圆角矩形
EBh	nn		显示一实心圆角矩形
ECh	nn		显示一带框实心圆角矩形
EDh	nn		显示一空心三角形
EEh	nn		显示一实心三角形
EFh	nn		显示一带框实心三角形
F4h	nn		显示一圆柱体
F6h	nn		显示一表格视窗

3.4.1 SPI Flash 的结构

工程编译之后按 BINGEN 按钮会在 Binfile 的文件夹中产生 UartTFT_Flash.bin 档和 UserInfo.bin 档案，这是要让开发者烧录到 SPI Flash 的档案。下图为 SPI Flash 的结构，可以看出 UartTFT_Flash.bin 档实际包括了 UserInfo.bin 档案的内容，而 UserInfo.bin 档是存放 Command 参数，其他区域则是示图片、动画文件、文字库等数据，如果只是修改 Command 参数，在 MakeBin 时虽然会同时产生 UartTFT_Flash.bin 档及 UserInfo.bin 档案，但实际上只需要更新 UserInfo.bin 档即可，UserInfo.bin 档案的大小固定为 128Kbytes，因此直接在 UI_Editor 环境下用 USB 转 RS232 连接线快速更新即可，不需要使用专用的 SPI Flash 烧录器。此外开发者可以透过 PC 的 USB 及使用 LT268B_ISP_Vxx.exe 程序将 Bin 档烧录到 SPI Flash 内，详细请参考第 6 章说明。

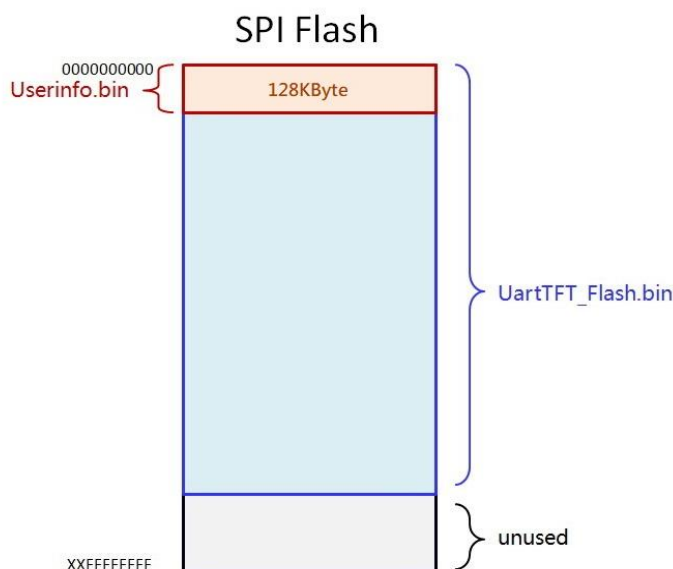


图 3-76: SPI Flash 的结构

3.4.2 Userinfo.bin (128K) 更新



图 3-77: UserInfo.bin 更新按钮

如果只是更新部分指令的个别参数，例如某张图片的显示位置、GIF 动画的播放速度、按键控件所执行的指令等不需要重新烧 Flash 的改变。可以在改变了这些参数后重新编译工程，产生 BIN 文件。之后打开 Command 使用 128K 更新去更改 SPI Flash 内的 UserInfo.bin，就可以更改 TFT 串口屏中的参数。

3.5 UI_Editor 范例下载

关于 UI_Editor 的使用，用户可以至 [乐升半导体](http://www.levetop.cn) 网站下载 LT268B 的 UI_Editor 范例（如 [LT268B_UI_Editor_Demo_240x320.rar](#)）来操作：

1. 自本公司网站（www.levetop.cn）下载 UI_Editor 压缩文件（UI_Editor_Vx_x_x.rar），然后解压缩 “UI_Editor_Vx_x_x.rar”。
2. 下载 UI_Editor 范例（如 LT268B_UI_Editor_Demo_240x320.rar）解压缩及打开压缩包。

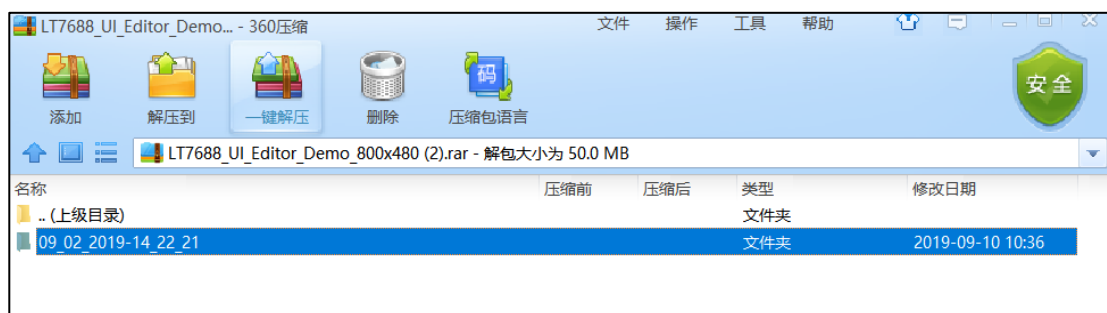


图 3-78：解压缩及打开压缩包

3. 将演示范例目录如 “09_02_2019-14_22_21” 文件夹解压到 UI_Editor 的 PROJECT 文件夹内。



图 3-79：范例目录拷贝到 UI_Editor 的 PROJECT 文件夹内

注意：

- ① 不能修改压缩包文件夹名称及内容。
- ② 压缩包内的文件夹必须解压到 UI_Editor 的 PROJECT 文件夹内。

4. 打开 UI_Editor，打开菜单选项后点击 Load：



图 3-80：打开 UI_Editor 点击 Load

5. 打开文件夹内的 COMMANDFILE，选择工程文件后点击打开。

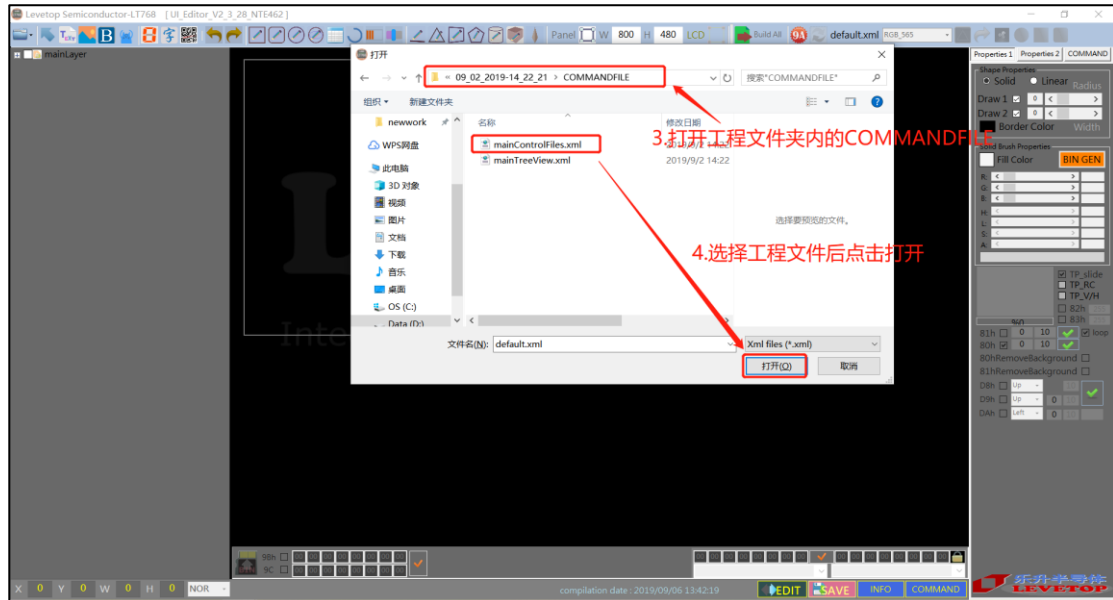


图 3-81：打开 UI_Editor 演示范例

4. 图文整合编译器(UartTFT_Tool.exe)

使用 LT268B TFT 串口屏进行显示功能开发的另一种方式是用 [图文整合编译器 \(UartTFT_Tool.exe\)](#)，本章节是介绍图文整合编译器所定义的程序格式及使用方式。图文整合编译器可以让使用者以文字编辑方式来撰写指令，编译器会将使用者的指令文件进行编译，然后将编译后的信息与图片字库等整合成一 Bin 文件，再透过电脑USB接口及 [LT268B_ISP_Vxx.exe](#) 程序将 Bin 文件烧录到 SPI Flash 内，完成 TFT 串口屏的显示开发，其示意简图如下图所示。

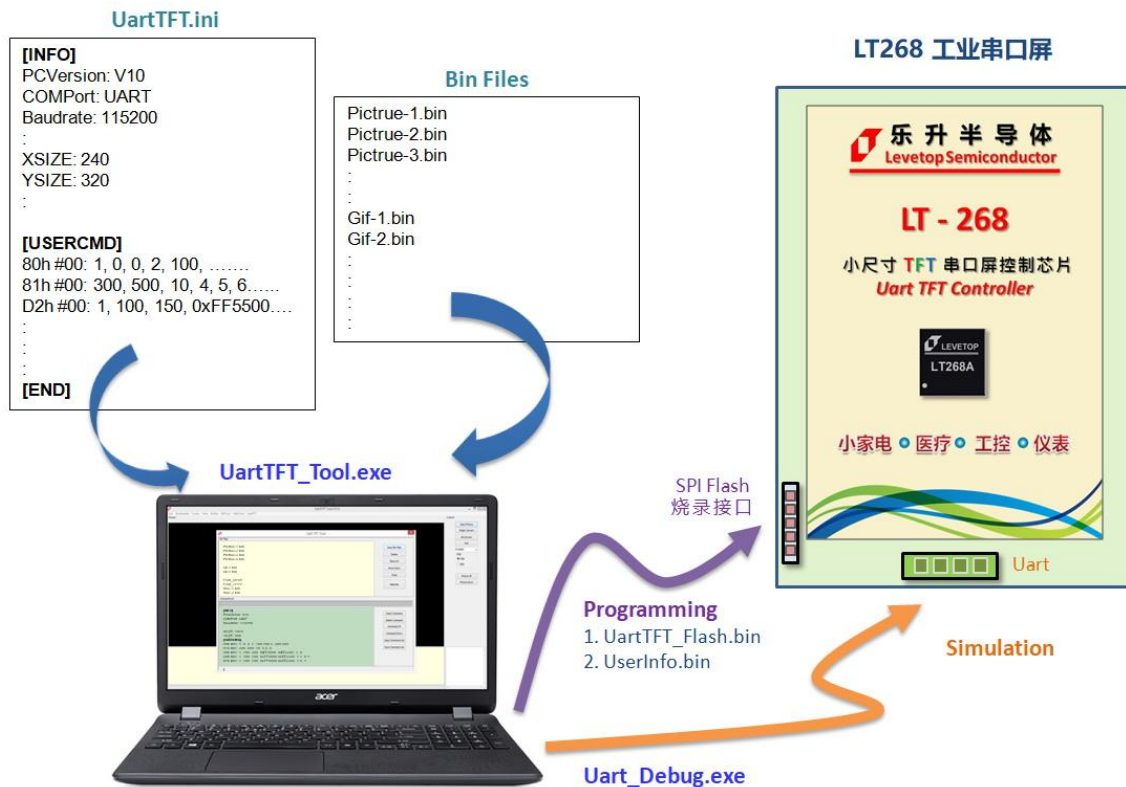


图 4-1：使用图文整合编译器的开发示意图

使用者先将 TFT 串口屏会用到的图片、字库，或是 Gif 档等准备好，然后开始编辑指令文件 ([UartTFT.ini](#))，编辑后点击图文整合编译器的编译器功能，产生 [UartTFT_Flash.bin](#) 和 [UserInfo.bin](#) 文件，其中 [UartTFT_Flash.bin](#) 是存放图片、字库，或是 Gif 档等数据数据，[UserInfo.bin](#) 是存放指令数据，再用 SPI Flash 烧写器将这 2 个 Bin 文件烧写到 SPI Flash 内，主控端的系统或是主板只需要传递指令码及动作序号给 TFT 串口屏，串口屏上的 LT268B 就会解析指令码，然后读取存在 SPI Flash 内的指令动作流程去实现图片或文字的显示。使用者可以用 USB 转 RS232 转接器，然后以 [UartTFT_Tool.exe](#) 连结到 [乐升半导体](#) 的 [UartDebug.exe](#) 通讯软件，可自 PC 端对 TFT 串口屏下指令做前期的验证，以确认设计是否有误或是需要修改。有关 UartDebug.exe 通讯软件的使用说明可以参考本应用手册第 5 章。

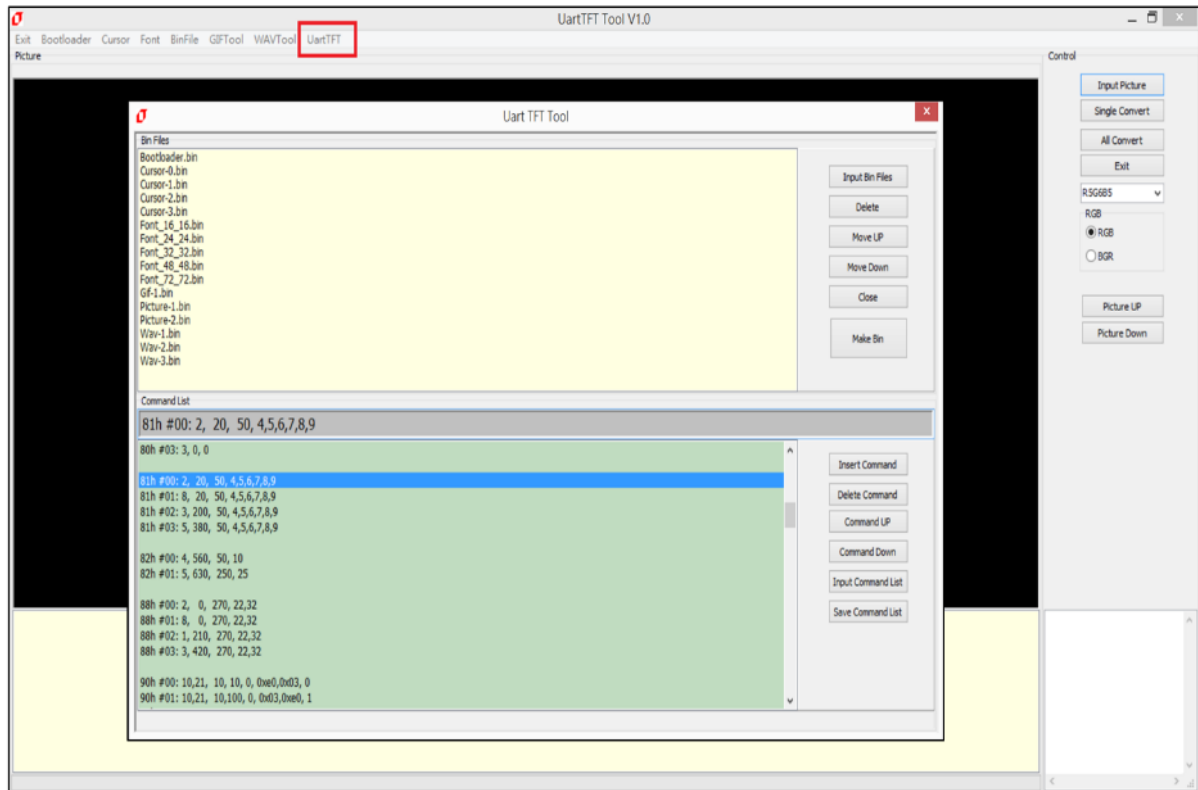


图 4-2: 图文整合编译器 (UartTFT_Tool.exe)

有关 LT268B 使用 UartTFT_Tool 的环境做开发，使用者可以自本公司网址(www.levetop.cn)下载 LT268B 的 UartTFT_Tool 演示案例 “LT268B_UartTFT_Tool_Demo_240x320.rar” 来进行操作。

4.1 编辑指令设定文件

使用图文整合编译器之前除了将会用到的图片、字库，或是 Gif 档等准备好之外，还必须用文本编辑器来撰写 **指令设定文件** (UartTFT.ini)，指令设定文件包括 2 个部分，一个是 TFT 串口屏的信息设定 [INFO]，另一个是 TFT 串口屏的指令参数设定 [USERCMD]，将分别如下说明。

4.1.1 信息设定 [INFO]

```
[INFO]
PCVersion: V10           // SW Version : 0x10=V1.0, 0x11=V1.1, 0x12=V1.2 .....
COMPort: UART           // COM Port: 0x00=UART, 0x01=SPI, 0x02=I2C
Baudrate: 115200         // UART Baudrate
PCBVersion: V10          // PCB Version : 0x10=V1.0, 0x11=V1.1, 0x12=V1.2 .....
MCUBit: 32               // MCU Data Bus: 0x00 = 8bit, 0x01 = 16bit, 0x02 = 32bit
MCUType: Levetop          // MCU Maker: 0x00 = Levetop, 0x01 = STM, 0x02 = STC
768Type: 268B            // 0x00: 7680A, 0x01: 7680B, 0x02: 7681, 0x03: 7683,
                          // 0x04: 7686, 0x05: 7688, 0x06: 268A, 0x07: 268B
768IF: SPI               // MCU to LT768 I/F: 0x00 = 8080-8, 0x01 = 8080-16,
                          // 0x02 = SPI, 0x03 = I2C

XSIZE: 240               // TFT Panel X-Size
YSIZE: 320               // TFT Panel Y-Size
VBPD: 23                 // Vsync Back-Porch
VFPD: 22                 // Vsync Front-Porch
VSPW: 3                  // Vsync Pulse Width
HBPD: 46                 // Hsync Back-Porch
HFPD: 210                // Hsync Front-Porch
HSPW: 20                 // Hsync Pulse Width
PCLKRISING: 1            // 0: Panel fetches XPDAT at XPClk rising edge, 1: falling edge
HSYNCPolarity: 0         // 0: Low active, 1: High active.
VSYNCPolarity: 0         // 0: Low active, 1: High active.
DEPolarity: 1            // 0: Low active, 1: High active.
RGBSequence: RGB         // 000b: RGB, 001b: RBG, 010b: GRB, 011b: GBR,
                          // 100b: BRG, 101b: BGR
ColorDepth: 16           // Color Depth: 0x00 = 8bit, 0x01 = 16bit, 0x02 = 24bit
FlashType: NOR            // FlashType: 0x00 = NOR Flash, 0x01 = NAND Flash
FlashSize: 16            // Flash Size, Unit: 1MByte
```

以上的信息设定大部分是 LT268B TFT 串口屏的设计厂商会提供，串口屏的设计厂商会依据 TFT 屏的特性，及硬件的设计方式提供上面的信息，这些信息提供给 TFT 串口屏，LT268B 内部程序得以进行适当的应对，让使用者的指令可以正确实现。因此使用者只要把它们加在指令文件的开头即可，例如使用 TFT 串口屏是 240*320 的 TFT 屏，则 XSIZE、YSIZE 必须设置如下：

```
XSIZE: 240
YSIZE: 320
```

TFT 串口屏 TFT 屏的分辨率在出厂时就是固定的，使用者只要把设计厂商针对该款 TFT 串口屏所提供的 [INFO] 信息全部加在指令文件的前端，这样就可以完成 [INFO] 的信息设定了。

4.1.2 指令参数设定 [USERCMD]

指令文件的另一个部分就是指令参数设定 [USERCMD]，这个部分就是使用者依据下面第 4.2 节所说的串口指令，将串口指令用文字表达来呈现，例如：

80h #00: 0, 0, 10, 20 // 在 (10, 20) 显示 P0

就是设计将编号为 0 的图片显示在 (10, 20) 的位置，一旦主控端 MCU 送来 Start(1Byte) + 80h + 00 + CRC(2Bytes) + End(4Bytes)，那么图片 0 会马上被显示在 (10, 20) 的位置，也就是图片 0 的左上角会在 (10, 20) 的位置上。

80h #01: 1, 0, 50, 30, 2, 1, 50, 150 // 在 (50, 30) 显示 P1；在 (50, 150) 显示 P2

上面例题就是设计同时将编号为 1 的图片显示在(50, 30)的位置，编号为 2 的图片显示在(50, 150) 的位置，也就是一旦主控端 MCU 送来 Start(1Byte) + 80h + 01 + CRC(2Bytes) + End(4Bytes)，那么图片 1 会显示在 (50, 30) 的位置，图片 2 会以 PNG 格式显示在 (50, 150) 的位置。再举一个例题：

88h #00: 1,1, 20, 180, GIF0 //在 (20, 180) 位置显示 GIF 0 动画

就是设计将编号为 0 的 Gif 动画以速度为 1 的方式显示在(20, 180)的位置，一旦主控端 MCU 送来 Start(1Byte) + 88h + 00 + CRC(2Bytes) + End(4Bytes)，那么 Gif 0 动画被显示在(20, 180) 的位置。

C0h #00: 1, 0, 10, 200, 0xFF5500, 0xEE3355, 1, 1, 0, 1
C0h #01: 1, 0, 100, 200, 0x335500, 0x883355, 1, 1, 0, 1

上面例题就是设计 2 个显示字符串的指令，一旦主控端送来 Start(1Byte) + C0h + 00 + “电子有限公司-LT268B” + CRC(2Bytes) + End(4Bytes)，TFT 串口屏就会在 (0, 10) 的位置以字库 1 的字型显示前景色 0xFF5500、背景色 0xEE3355、不放大、不透明的模式显示出“电子有限公司-LT268B”。主控端送来 Start(1Byte) + C0h + 01 + “串口屏” + CRC(2Bytes) + End(4Bytes)，TFT 串口屏就会在 (100, 100) 的位置以字库 1 的字型显示前景色 0x335500、背景色 0x883355、不放大、不透明的模式显示出“串口屏”，指令内的 200 代表文字框宽度。

下面是一个指令设定 [[USERCMD]] 的范例：

[USERCMD]

80h #00: 1, 0, 10, 20, 3, 100, 200

// 在 (0,0) 显示 P1; (10,20) 显示 P2, (100, 200) 显示 P3

81h #00: 100, 50, 10, 4, 5, 6

// 在 (100,50) 显示 P4, P5, P6, Time interval: 1sec

90h #00: PT0, 20, 30, 0xFF5500, 0xEE3355, 2, 0

// 靠左在 (20,30) 显示 "91h Number" with PictureNumber-1, FC=0xFF5500, BC=0xEE3355, pitch=2,

C0h #00: 1, 10, 20, 100, 0xFF5500, 0xEE3355, 1, 1, 0, 1

// 在 (10,20) 显示 "C0h String" with Font-1, IntF-1, FC=0xFF5500, BC=0xEE3355, HS=1, VS=1, 不透明, 对齐, 100 代表文字框宽度。

C1h #00: 1, 10, 80, 100, 0xFF5500, 0xEE3355, 1, 1, 0, 1

// 在 (10,80) 显示 "C1h String" with Font-2, IntF-1, FC=0xFF5500, BC=0xEE3355, HS=1, VS=1, 不透明, 对齐, 100 代表文字框宽度。


在指令文件的最前面加上 [INFO] 表示后面的描述是信息设定；而 [USERCMD] 表示后面的描述是指令设定；指令文件的最后必须加上 **[END]**，代表整个指令文件的结尾，这样编译器才会知道每一段文件代表的意义。

4.2 指令参数的设定

4.2.1 固定图片设定及显示指令

显示图片是 TFT 串口屏的主要功能，LT268B 的 TFT 串口屏有表 4-1、表 4-2 所列的几种图片设定指令。通常使用者会依据屏得分辨率及产品所需要的显示内容先设计好这些图片，这些图片在使用者设计完后可以用图文整合编译器导入，每张图片都会赋予图片编号，同时产生 Bin 文档，再烧录到 SPI Flash 内。因此 TFT 串口屏收到指令后会依据指令参数在 SPI Flash 内找到图片信息，然后透过 LT268B 硬件加速功能将图片显示在 TFT 屏上。

表 4-1：固定图片设定指令

指令功能	指令码	序号	指令参数	指令说明
设定显示多张图片	80h	#nn:	Paa(2), PNGaa(1), Xaa(2), Yaa(2), Pbb, PNGbb, Xbb, Ybb, Pcc, PNGcc, Xcc, Ycc	<p>将图片 P 显示在各别的 (X, Y) 位置。PNG=1 代表背景为透明 (Png 图档)，PNG=0 则否，其中 nn 是所有指令的序号，一律由 0 开始；aa、bb、cc 代表图片编号。每个序号最多支持 10 张图片。</p>  <p>注意：() 内的数字代表占用的字节数。</p>
设定显示多张图片	8Ah	#nn:	Paa(2), PNGaa(1), Xaa(2), Yaa(2), Pbb, PNGbb, Xbb, Ybb, Pcc, PNGcc, Xcc, Ycc	此为 80h 的扩展指令，用法与 80h 相同。

例如在 UartTFT.ini 输入下面的 2 个设定图片指令：

```
80h #00: 1, 0, 50, 20, 2, 0, 50, 120    // 在 (50, 20) 显示 P1; (50, 120) 显示 P2
80h #01: 3, 0, 180, 20                  // 在 (180, 20) 显示 P3
```

在编译完成设定指令后，主控端的系统或是主板透过 UART 串口传递显示指令 Start(1Byte) + 80h + 00 + CRC(2Bytes) + End(4Bytes) 给 TFT 串口屏，那么 TFT 上就会在 (50, 20) 显示第一张图片，在 (50, 120) 显示第二张图片；当主控端透过 UART 串口传递显示指令 Start(1Byte) + 80h + 01 + CRC(2Bytes) + End(4Bytes) 给 TFT 串口屏，那么 TFT 上就会在 (180, 20) 显示第三

张图片。如下图所示（以 320*240 的 TFT 显示屏为例）。

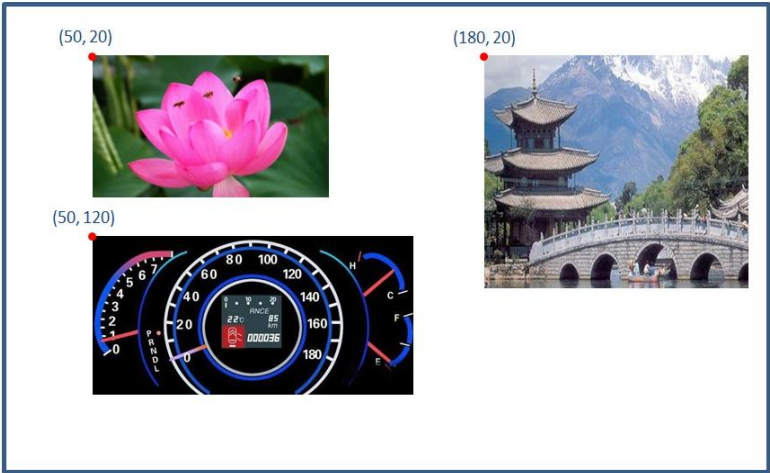


图 4-3：80h 指令的显示图片范例

除了 80h、8Ah 指令外，主控端主板也可以直接使用 8Fh 的固定命令传送给 TFT 串口屏直接将指定的图号显示出来，这指令不需要在 UartTFT_Tool 上位机软件或是 UartTFT.ini 内设定：

表 5-1B：固定图片显示指令

指令功能	指令码	序号	指令参数	指令说明
直接显示单张图片	8Fh	nn	X(2), Y(2), PNG(1), Pnn(2)	将编号为 nn 的图片 P 显示在 (X, Y) 位置。PNG=1 代表背景为透明(Png 图档)，PNG=0 则否，

例如主控端的系统或是主板透过 UART 串口传递显示命令 Start(1Byte) + 8Fh + 00 + 0064h(2Bytes) + 0032h(2Bytes) + 00h(1Bytes) + 0001h(2Bytes) + (CRC(2Bytes) + End(4Bytes) 给 TFT 串口屏，那么 TFT 上就会在 (100, 50) 显示第 1 张 bmp 格式的图片。

传递显示命令 Start(1Byte) + 8Fh + 00 + 0064h(2Bytes) + 00F0h(2Bytes) + 01h(1Bytes) + 0002h(2Bytes) + (CRC(2Bytes) + End(4Bytes) 给 TFT 串口屏，那么 TFT 上就会在 (100, 240) 显示第 2 张具有 Png 格式的图片。

主控端 MCU 透过 UART 串口传递的指令都要在前端加 Start(1Byte) 及后端加上 CRC(2Bytes) 及 End(4Bytes) 给 TFT 串口屏，请参考 表 2-2：主控端与 TFT 串口屏协议表，而为避免冗长说明，后续所提到的范例将省略 Start(1Byte)、CRC(2Bytes) 及 End(4Bytes)。

4.2.2 动态图片设定及显示指令

表 4-2：动态图片设定指令

指令功能	指令码	序号	指令参数	指令说明
设定循环显示 重迭图片	81h	#nn:	Delay(1), X(2), Y(2), PNG(1), Paa(2), Pbb(2), Pcc(2)	将各图片循环显示在 (X, Y) 的位置；切 换时间为 Delay*10ms，Delay 的设置范 围为 1~99。PNG=1 代表背景为透明(Png 图档)，PNG=0 则否，常用在类似循环动 画效果的显示。每个序号 最多支持 10 张 图片。 
取消循环显示 重迭图片	84h	nn		将 81h 设定循环显示重迭图片取消
设定显示 GIF 动画图片	88h	#nn:	Loop(1), Delay(1), X(2), Y(2), GIFaa(1)	将 GIF 动画图片循环显示在 (X, Y) 的位 置；GIF 动画是由许多张图片组成，每张 图片切换时间为 Delay*10ms。其中 aa 代表 GIF 檔的编号。Loop 代表循环模式， Loop=0 代表循环播放一次（循环结束， 动画静止），Loop=1 代表一直循环播放。 常用在类似开机动画的效果显示。 
取消显示 GIF 动画图片	89h	nn		将 88h 设定显示 GIF 动画图片取消

如果想在 TFT 屏上显示类似循环动画效果，可以使用 81h 的设定指令，如下图中 P4~P9 是相同大小的图片，用 81h 指令可以产生循环风扇的显示效果。

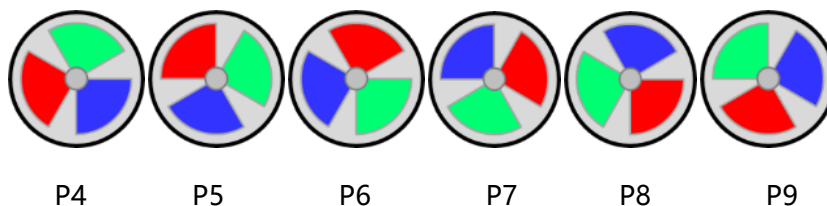


图 4-4：循环风扇的图片

例如在 UartTFT.ini 输入下面的设定图片指令：

```
81h #00: 5, 50, 50, 1, 4, 5, 6, 7, 8, 9 // 在 (50, 50) 显示 P4~P9, @50ms
```

在编译完成设定指令后，主控端系统透过 UART 串口传递显示指令 81h 及 00 给 TFT 串口屏，那么 TFT 上就会在 (50, 50) 的位置每隔 50ms 显示第 4 张到第 9 张图片（背景为透明）的循环图片。

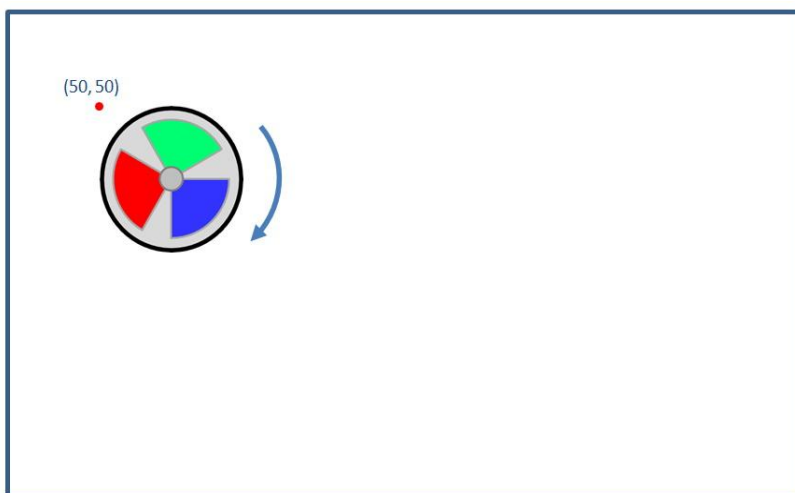


图 4-5：81h 指令的显示循环动画范例

主控端的系统或是主板透过 UART 串口传递显示命令 84h 及 00 给 TFT 串口屏，那么原 81h、00 显示的循环动画图片将被取消。

上表 4-2 中指令 88h 所显示的 GIF 动画图片功能，也是在使用者设计完后用图文 UI 编辑器或是图文整合软件导入，它会自动产生连续编号的许多张图片，与其它图片或字库整合成一 Bin 文档后，再烧录到 SPI Flash 内。在编译完成后，主控端系统透过 UART 串口传递显示指令 88h 给 TFT 串口屏，那么 TFT 串口屏收到指令后会依据指令参数在 SPI Flash 内找到 GIF 档的连续图片信息，然后透过 LT268B 分时显示在 TFT 屏上，此功能常用在类似开机动画的效果显示。

88h #00: 1, 3, 50, 100, Gif-1

// 在 (50, 100) 显示 GIF-1, @30ms

如上例，在编译完成设定指令后，主控端系统透过 UART 串口传递显示指令 88h 及 #00 给 TFT 串口屏，那么 TFT 上就会在 (50, 100) 的位置依序播放 GIF-1 檔的所有动画图片，一直循环。

若想只循环播放一次 GIF 动画，可以发送下面指令：

88h #00: 0, 3, 50, 100, Gif-1

// 在 (50, 100) 显示 GIF-1, @30ms

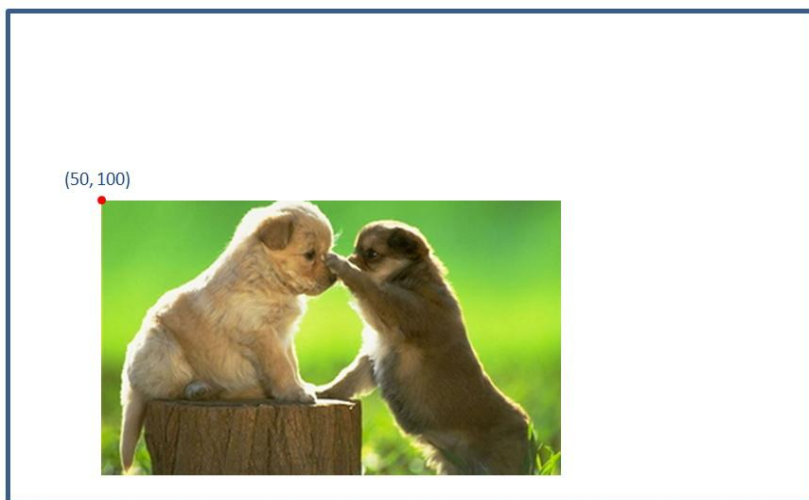


图 4-6：88h 指令的显示 GIF 图片范例

主控端的系统或是主板透过 UART 串口传递显示命令 89h 及 00 给 TFT 串口屏，那么原 88h、00 显示的 GIF 动画将被取消。

4.2.3 控件图片设定及显示指令

表 4-3：控件图片设定指令

指令功能	指令码	序号	指令参数	指令说明
设定显示控件功能的图片	A0h	#nn:	P(2), PNG(1), X(2), Y(2), CM1(1), NU1(1), CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8	将图片 P 显示在 (X, Y) 位置, PNG=1 代表背景为透明 (Png 图档), PNG=0 则否。当触控按在图片 P 上后, TFT 串口屏会自动执行 CM 命令, NU 为 CM 命令后的序号。此指令可以在按下触控后同时执行 8 组命令。
取消显示控件功能的图片	A1h	nn		将 A0h 设定显示控件功能的图片取消

例如下面范例 1 在 UartTFT.ini 输入下面的控件功能图片指令：

```
80h #01: 3, 0, 180, 20           // 在 (180, 20) 显示 P3
88h #00: 3, 50, 100, Gif-1      // 在 (50, 100) 显示 GIF-1, @30ms
A0h #00: 7, 1, 50, 50, 0x80, 0x01, 0x88, 0x00
// 在 (50, 50) 显示 P7, 触控按在图片 P7 上后会自动执行 80h 及 01 显示指令, 及 88h、00
播放 Gif 动画指令。
```

在编译完成设定指令后, 主控端的系统或是主板透过 UART 串口传递显示指令 A0h 及 00 给 TFT 串口屏, 那么 TFT 上就会在 (50, 50) 显示第 7 张图片 (背景为透明), 而当触控按在 7 张图片上后, TFT 串口屏会自动执行显示指令 80h 及 01, 那么 TFT 上就会在 (180, 20) 显示第三张图片 (参考第 4.2.1 节范例), 同时在 (50, 100) 显示 GIF-1 动画, 如下图所示。

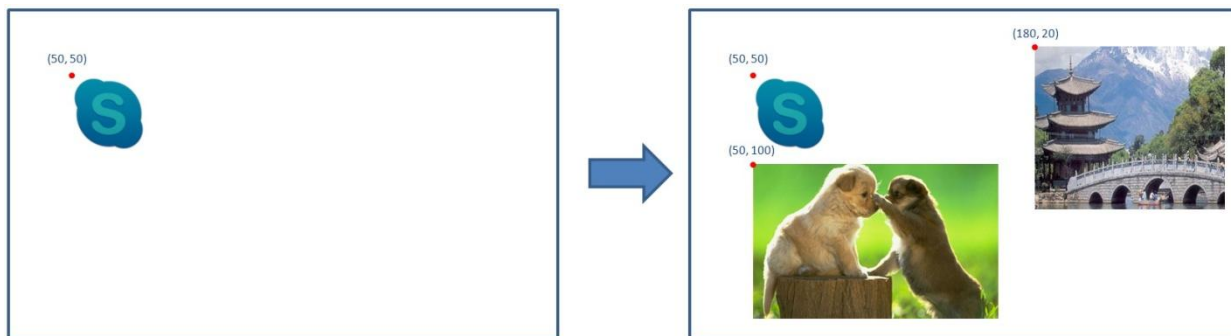


图 4-7A：A0h 指令的控件图片显示范例 1

主控端的系统或是主板透过 UART 串口传递显示指令 A1h 及 00 给 TFT 串口屏, 那么原 A0h、00 显示的控件图片及触控功能将被取消。

A0 控件指令可以支持在按下触控屏后同时执行 8 组指令，每组指令包括 2 个 Bytes，一个是指令码 CM，另一个是指令码的序号 NU，没有用到的部分不须补 0x00, 0x00。

当 TFT 串口屏在控件图片触控位置按下时会响应 10 个 Byte 反馈信息，透过 Uart 串口传递给主控端，包括**起始码**、**指令码**、**序号**、**状态反馈码**、**CRC 码**、**结束码**，此时的**序号**代表控件图片 ID 号、按下时**状态反馈码**为 **0x31**，如格式下表所示：

表 4-4A：按下控件图片时反馈信息

起始码	指令码	序号	状态反馈码	CRC 码	结束码
0xAA (1 Byte)	A0h (1 Byte)	控件图片 ID 号 (1 Byte)	0x31	(2 Bytes)	0xE4、0x1B、 0x11、0xEE (4 Bytes)

当控件图片触控位置放开后也会响应 10 个 Byte 反馈信息，其差别为**状态反馈码**是 **0x30**，有关串口屏的指令反馈信息请参考第 2.2 节。

表 4-4B：控件图片松开后反馈信息

起始码	指令码	序号	状态反馈码	CRC 码	结束码
0xAA (1 Byte)	A0h (1 Byte)	控件图片 ID 号 (1 Byte)	0x30	(2 Bytes)	0xE4、0x1B、 0x11、0xEE (4 Bytes)

除了 A0h 控件图片功能外，LT268B 串口屏还提供一种不带图片的虚拟控件指令 - A2h，可以让使用者在 TFT 触控屏上设置一个区域，当触控按在此区域上后，串口屏会自动执行 8 组命令

表 4-5A：虚拟控件设定指令

指令功能	指令码	序号	指令参数	指令说明
设定虚拟控件区域及功能	A2h	#nn:	X(2), Y(2), Width(2), Height(2), CM1(1), NU1(1), CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8	将 (X, Y) 位置，宽度为 Width，高度为 Height 的区域设置为触控区。当触控按在此区域上后，串口屏会自动执行 CM 命令，NU 为 CM 命令后的序号。此指令可以在按下触控后同时执行 8 组命令。
取消虚拟控件区域及功能	A3h	nn		将 A2h 设定虚拟控件功能取消

例如下面范例 1 在 UartTFT.ini 输入下面的控件功能图片指令：

```
80h #01: 3, 0, 500, 50           // 在 (500, 50) 显示 P3
88h #00: 1, 3, 100, 200, Gif-1    // 在 (100, 200) 显示 GIF-1, @30ms
A2h #00: 100, 100, 200, 75, 0x80, 0x01, 0x88, 0x00
// 在 (100, 100) 宽为 200，高为 75 的区域，触控按在此区域上后会自动执行 80h 及 01 显示命令，及 88h、00 播放 Gif 动画指令。
```

在编译完成设定指令后，主控端的系统或是主板透过 UART 串口传递显示命令 A2h 及 00 给串口屏，那么当触控按在设置的虚拟区域上后，串口屏会自动执行显示命令 80h 及 01，那么 TFT 上就会在 (500, 50) 显示第三张图片（参考第 4.2.1 节范例），同时在 (100, 200) 显示 GIF-1 动画，如下图所示。

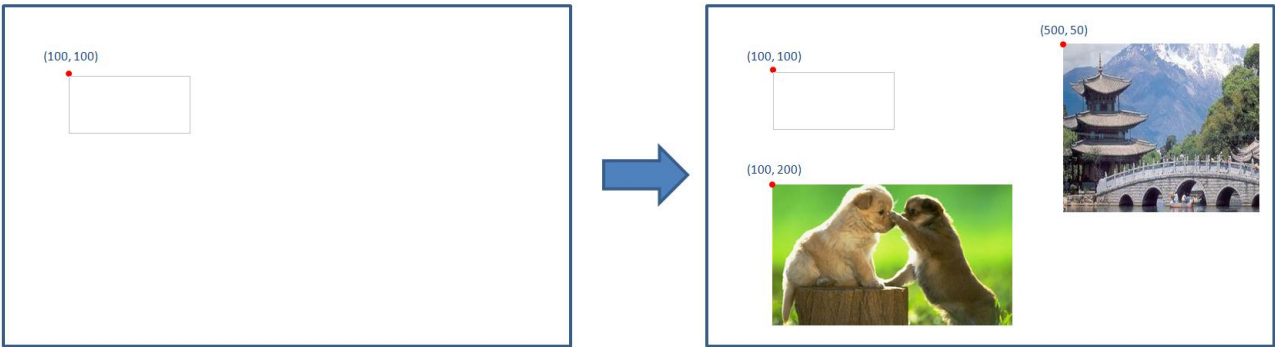


图 4-7B：A2h 指令的虚拟控件范例

主控端的系统或是主板透过 UART 串口传递显示命令 **A3h 及 00** 给 TFT 串口屏，那么原 A2h、00 的触控功能将被取消。

A2 控件指令可以支持在按下触控屏后同时执行 8 组指令，每组指令包括 2 个 Bytes，一个是命令码 CM，另一个是命令码的序号 NU，没有用到的部分不须补 0x00, 0x00。

当虚拟触控位置按下或放开后时也会响应 10 个 Byte 反馈信息，透过 Uart 串口传递给主控端，包括起始码、指令码、序号、状态反馈码、CRC 码、结束码，此时的序号代表虚拟控件 ID 号、按下时状态反馈码为 **0x31**，如格式下表所示：

表 4-5B：按下虚拟控件时反馈信息

起始码	指令码	序号	状态反馈码	CRC 码	结束码
0xAA (1 Byte)	A2h (1 Byte)	虚拟控件 ID 号 (1 Byte)	0x31	(2 Bytes)	0xE4、0x1B、 0x11、0xEE (4 Bytes)

当触控位置放开后也会响应 10 个 Byte 反馈信息，其差别为**状态反馈码**是 **0x30**，有关串口屏的指令反馈信息请参考第 2.2 节。

表 4-5C：虚拟控件松开后反馈信息

起始码	指令码	序号	状态反馈码	CRC 码	结束码
0xAA (1 Byte)	A2h (1 Byte)	虚拟控件 ID 号 (1 Byte)	0x30	(2 Bytes)	0xE4、0x1B、 0x11、0xEE (4 Bytes)

4.2.4 图片式的数字显示指令

显示数字也是 TFT 串口屏应用的常用功能，其来源有可能是使用者以图片方式自己建立的数字，或是由 PC 导入的字库来建立的数字。图片式的数字显示指令是指使用者在 TFT 屏上想要显示数字的“字形”是自己用单色图片所建立的，例如在应用上想要显示一个数字，但是字库里又没有适当大小的数字库，那么使用者可以建一套 0~9 的数字图片，然后用这些指令来调用这些数字（图片）在想要显示的地方，图片式的数字显示指令有 2 种，如果是使用 90h 指令，则数字的前景与背景可以由指令参数来指定。如果是使用 91h 指令，则数字的颜色与图片一致。

表 4-6：图片式的数字设定指令

指令功能	指令码	序号	指令参数	指令说明
设定显示图片式（客制化）的数字 - 1	90h	#nn:	PT(2), X(2), Y(2), Dir(1), Color-F(1), Color-B(1), EN-B(1)	将图片式的数字显示在 (X, Y) 位置，前景颜色为 Color-F、背景颜色为 Color-B、显示方向为 Dir（0：靠左；1：靠右），其中 PT 代表图片式的数字编号。EN-B=0 代表背景为透明，EN-B=1 代表去背景（透明）。
设定显示图片式（客制化）的数字 - 2	91h	#nn:	CPT(2), X(2), Y(2), Dir(1), EN-B(1)	将图片式的数字显示在 (X, Y) 位置，显示方向为 Dir（0：靠左；1：靠右），其中 CPT 代表图片式的数字编号。EN-B=0 代表背景为透明，EN-B=1 代表去背景（透明）。

使用显示图片式（客制化）的数字，其在 TFT 屏上显示的大小取决于使用者在图片设计时的分辨率，同时在设计时数字必须使用单色，使用者设计完这些图片文件后可以用上位机软件或是图文整合软件导入产生 Bin 文档，再烧录到 SPI Flash 内。而在 LT268B 的 BTE 功能下，当使用这些指令时，可以让使用者建立的图片式数字用不同颜色显示前景、背景，或是背景透明。如下是 2 组不同大小的数字图片组，每个字都是代表一个图片文件：

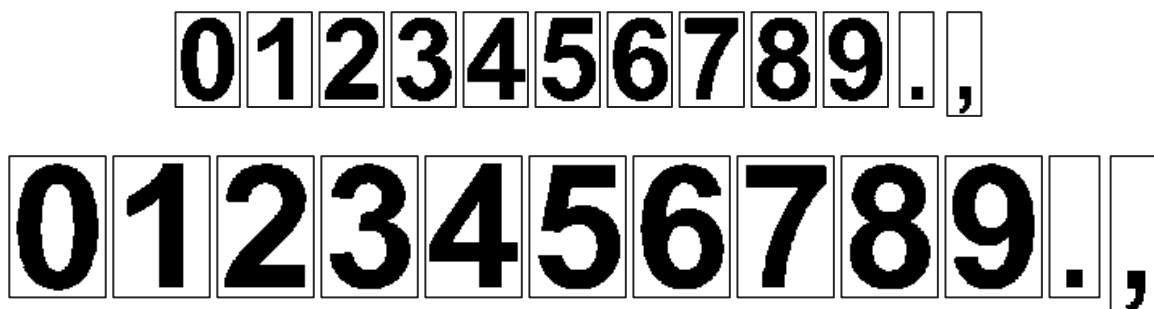


图 4-8A：客制化的图片数字

例如在文字编译程序 `UartTFT.ini` 输入下面的显示图片式文字指令：

```
90h #00: PT0, 40, 020, 0, 0xE0, 0x03, 0
90h #01: PT0, 40, 120, 0, 0x03, 0xE0, 1
```

在编译完成设定指令后，主控端的系统或是主板透过 UART 串口传递显示指令 `90h`、`00` 及 `"867.3"` 给 TFT 串口屏，那么 TFT 上就会在 (40, 20) 起的位置依据第 1 个图片式的数字组去显示前景色为 `0xE3`(红色)、背景色为 `0x03`(蓝色)的数字 867.3。当 UART 串口传递指令 `90h`、`01` 及 `"4,567"` 给 TFT 串口屏，那么 TFT 上就会在 (40, 120) 起的位置依据第 1 个图片式的数字组去显示前景色为 `0x03`(蓝色)的数字 4,567，因为 `EN-B=1` 代表去背景(透明)，所以设定的背景色 `0xE3`(红色)不会秀出。

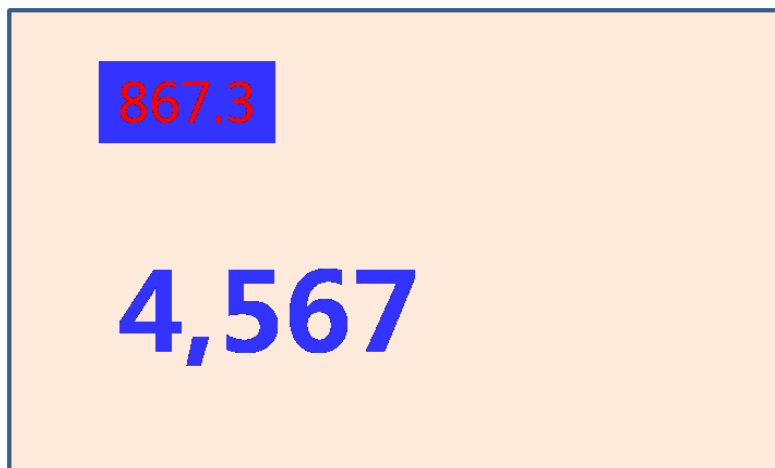


图 4-8B：显示图片式数字范例

`91h` 指令用法与 `90h` 类似，只是不需要设置前景色及背景色，显示出来的数字颜色与图片完全一致，例如建立一组 `91h` 指令的客制化的图片数字：



图 4-9A：91h 指令的客制化的图片数字

在文字编译程序 `UartTFT.ini` 输入下面的显示图片式文字指令：

```
91h #00: CPT0, 100, 50, 0, 0
```

在编译完成设定指令后，主控端的系统或是主板透过 UART 串口传递显示命令 91h、00 及 "867.3" 给 TFT 串口屏，那么 TFT 上就会在(100, 50)起的位置依据第 1 个图片式的数字组(CPT0)去显示的数字 867.3：

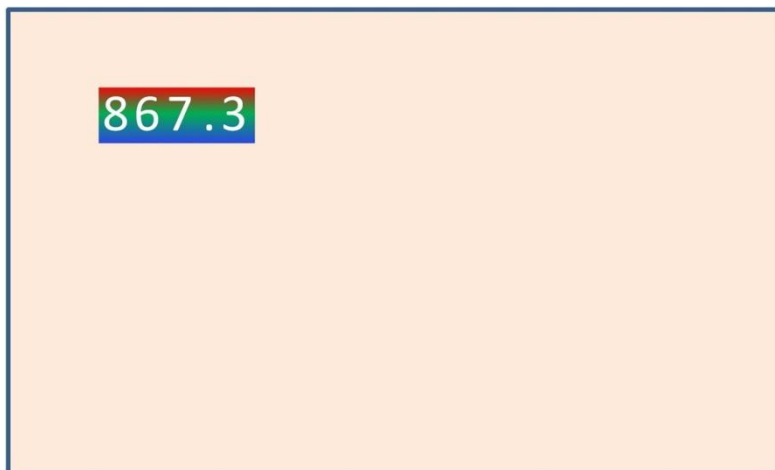


图 4-9B: 91h 指令显示图片式数字范例

4.2.5 使用字库的文字设定及显示指令

使用字库的文字显示指令是指使用者在 TFT 屏上想要显示的文字或数字来源是由 PC 端导入的字库，例如在应用上想要显示一串文字，可以藉由 乐升半导体 提供的图文整合编译器软件将想要的字库产生 Bin 文件，例如用图文整合编译器软件产生一套 16*16 的 Font_16_16.bin 文件、一套 24*24 的 Font_24_24.bin、一套 32*32 的 Font_32_32.bin 文件，然后都烧录到 SPI Flash 内，之后用这些指令来调用这些文字在想要显示的地方。用 C0~C3 的指令所调用字库的分辨率限定为 16*16、24*24、32*32，如果外部 SPI Flash 容量许可，可以允许使用到 4 组中文字库（字库-1 ~ 字库-4）。

表 4-7：使用字库的文字设定指令

指令功能	指令码	序号	指令参数	指令说明
设定显示字库-1 文字	C0h	#nn:	F01(1), X(2), Y(2), W(2), Color-F(3), Color-B(3), Size-H(1), Size-V(1), Transparency(1), Alignment(1)	将字库-1 文字显示在 (X, Y) 位置，前景颜色为 Color-F、背景颜色为 Color-B、水平垂直放大分别为为 Size-H 和 Size-V。Transparency = 0：不透明; 1: 透明。Alignment = 0：不对齐; 1: 对齐。W 代表文字框宽度。
设定显示字库-2 文字	C1h	#nn:	F02(1), X(2), Y(2), W(2), Color-F(3), Color-B(3), Size-H(1), Size-V(1), Transparency(1), Alignment(1)	将字库-2 文字显示在 (X, Y) 位置，前景颜色为 Color-F、背景颜色为 Color-B、垂直水平放大分别为为 Size-H 和 Size-V。Transparency = 0：不透明; 1: 透明。Alignment = 0：不对齐; 1: 对齐。W 代表文字框宽度。
设定显示字库-3 文字	C2h	#nn:	F03(1), X(2), Y(2), W(2), Color-F(3), Color-B(3), Size-H(1), Size-V(1), Transparency(1), Alignment(1)	将字库-3 文字显示在 (X, Y) 位置，前景颜色为 Color-F、背景颜色为 Color-B、垂直水平放大分别为为 Size-H 和 Size-V。Transparency = 0：不透明; 1: 透明。Alignment = 0：不对齐; 1: 对齐。W 代表文字框宽度。
设定显示字库-4 文字	C3h	#nn:	F04(1), X(2), Y(2), W(2), Color-F(3), Color-B(3), Size-H(1), Size-V(1), Transparency(1), Alignment(1)	将字库-4 文字显示在 (X, Y) 位置，前景颜色为 Color-F、背景颜色为 Color-B、垂直水平放大分别为为 Size-H 和 Size-V。Transparency = 0：不透明; 1: 透明。Alignment = 0：不对齐; 1: 对齐。W 代表文字框宽度。

如上指令所示，在 LT268B 的 BTE 功能下，可以让文字用不同颜色显示前景、背景，或是背景透明。例如在 UartTFT.ini 输入下面的显示文字指令：

```
C0h #00: 1, 40, 40, 200, 0xFF0000, 0x0000FF, 1, 1, 0, 0
```

```
C1h #00: 1, 40, 120, 200, 0xFF0000, 0x00FF00, 1, 1, 0, 0
```

当 UART 串口传递显示指令 C0h、00 及 “深圳市乐升半导体有限公司-LT268” 给 TFT 串口屏，那么 TFT 上就会在 (40, 40) 起的位置依据第 1 个字库，以不放大、不透明、不对齐去显示前景色为 0xFF0000、背景色为 0x0000FF 的 “深圳市乐升半导体有限公司-LT268” 字符串，如果第 1 个字库是 24*24 大小，那显示出来就是 24*24 大小的字。当 UART 串口传递显示指令 C1h、00 及 “深圳市乐升半导体” 给 TFT 串口屏，那么 TFT 上就会在 (40, 120) 起的位置依据第 2 个字库，以放大 4 倍、不透明、不对齐去显示前景色为 0xFF0000、背景色为 0x00FF00 的 “深圳市乐升半导体” 字符串，显示出来就是 32*32 大小的字。

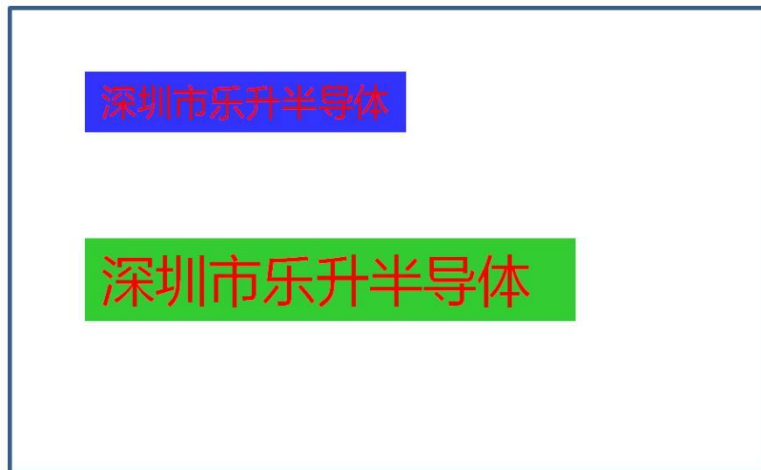


图 4-10：使用 24*24、32*32 字库的文字显示范例

4.2.6 画点指令

LT268B 内部包含功能强大的几何绘图硬件加速器，因此执行几何绘图时非常有效率，只需要配合输入简单的图形参数，就可以画出几何图形，以下为画制几何图形的串口指令：

表 4-8：几何绘图设定指令 - 画点

指令功能	指令码	序号	指令参数	指令说明
设定画点	DFh	#nn:	Type(1), R(1), Color(3)	画颜色为 Color 的圆点 (Type=0) 或是方形点(Type=1) ，画圆点时 R 代表半径，画方形点 R 代表宽度。

例如在 UartTFT.ini 输入下面的画点指令：

```
DFh #00: 0, 3, 0xFF0000          // 显示半径为 3 的红色圆点
DFh #01: 1, 5, 0x0000FF          // 显示长宽为 5 的蓝色方形点
```

在编译完成后，主控端的系统或是主板透过 UART 串口传递显示指令 DFh、00、0010、0010 给 TFT 串口屏，那么 TFT 上就会在 (10, 10) 显示半径为 3 的红色圆点。主控端的系统或是主板透过 UART 串口传递显示指令 DFh、01、0020、0020 给 TFT 串口屏，那么 TFT 上就会在 (20, 20) 显示长宽为 5 的蓝色方形点。

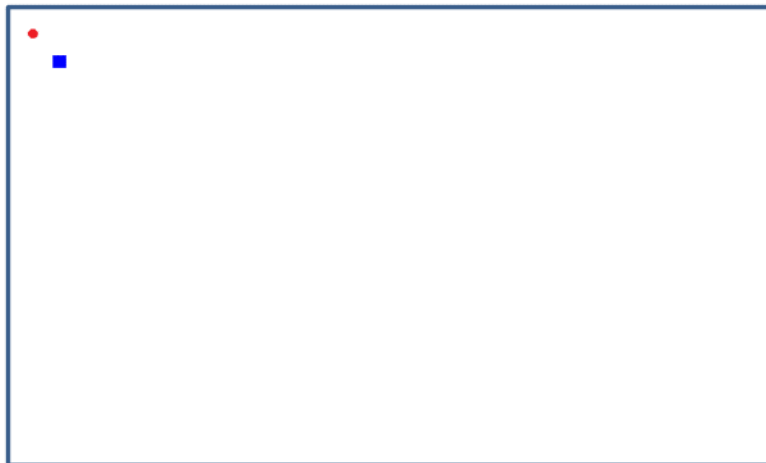


图 4-11：画点指令的显示范例

4.2.7 画圆形/椭圆形指令

表 4-9：几何绘图设定指令 - 圆形

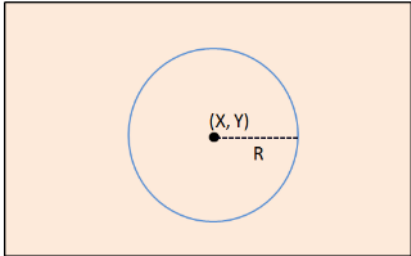
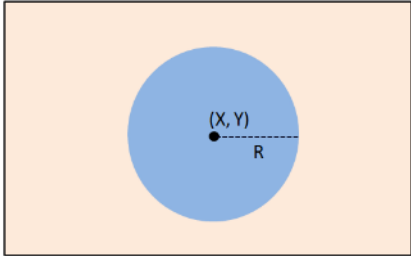
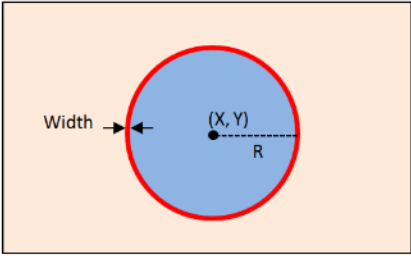
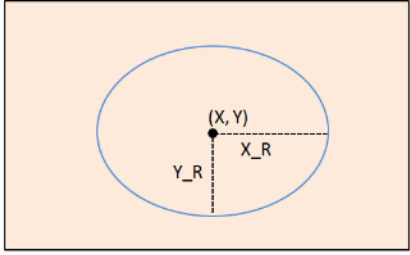
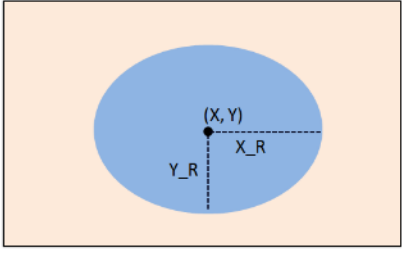
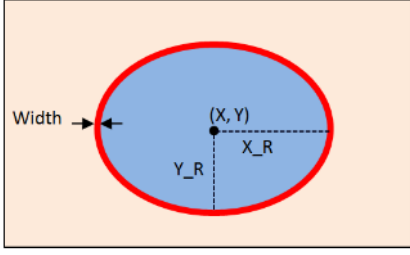
指令功能	指令码	序号	指令参数	指令说明
设定画一空心圆形	E1h	#nn:	X(2), Y(2), R(2), Color(3)	<p>以 (X, Y) 为圆心画一半径为 R、颜色为 Color 的空心圆形。</p> 
设定画一实心圆形	E2h	#nn:	X(2), Y(2), R(2), Color(3)	<p>以 (X, Y) 为圆心画一半径为 R、颜色为 Color 的实心圆形。</p> 
设定画一带框实心圆形	E3h	#nn:	X(2), Y(2), R(2), Color(3), Color-F(3), Width(1)	<p>以 (X, Y) 为圆心画一半径为 R、框颜色为 Color、宽为 Width、实心颜色为 Color-F 的带框实心圆形。</p> 

表 4-10：几何绘图设定指令 - 椭圆形

指令功能	指令码	序号	指令参数	指令说明
设定画一空心椭圆形	E4h	#nn:	X(2), Y(2), X-R(2), Y-R(2), Color(3)	以 (X, Y) 为圆心画一 X 半径为 X-R、Y 半径为 Y-R、颜色为 Color 的空心椭圆形。 
设定画一实心椭圆形	E5h	#nn:	X(2), Y(2), X-R(2), Y-R(2), Color(3)	以 (X, Y) 为圆心画一 X 半径为 X-R、Y 半径为 Y-R、颜色为 Color 的实心椭圆形。 
设定画一带框实心椭圆形	E6h	#nn:	X(2), Y(2), X-R(2), Y-R(2), Color(3), Color-F(3), Width(1)	以 (X, Y) 为圆心画一 X 半径为 X-R、Y 半径为 Y-R、框颜色为 Color、宽为 Width、实心颜色为 Color-F 的带框实心椭圆形。 

如上指令 E1h~E6h 所示，例如在 UartTFT.ini 输入下面的几何绘图设定指令：

E1h #00: 40, 40, 30, 0x00F800
E2h #00: 120, 40, 30, 0x00F800
E3h #00: 200, 40, 30, 0x00F800, 0x00001F, 5
E4h #00: 40, 160, 30, 18, 0x00F800
E5h #00: 120, 160, 30, 18, 0x00F800

E6h #00: 200, 160, 30, 18, 0x00F800,0x00001F, 5

当 UART 串口传递显示指令 E1h 00、E2h 00、E3h 00 给 TFT 串口屏，那么 TFT 上就会在 (40, 40) (120, 40) (200, 40) 起的位置显示出指定颜色的空心圆形、实心圆形及带框实心圆形。当 UART 串口传递显示指令 E4h 00、E5h 00、E6h 00 给 TFT 串口屏，那么 TFT 上就会在 (40, 160) (120, 160) (200, 160) 起的位置显示出指定颜色的空心椭圆形、实心椭圆形及带框实心椭圆形，如下图所示：

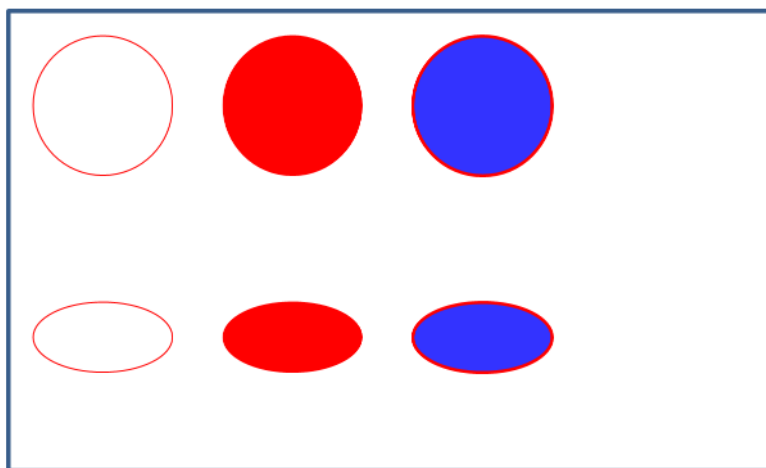
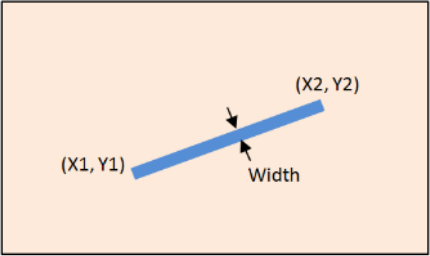


图 4-12A：圆形与椭圆形指令的显示范例

4.2.8 画直线指令

表 4-11：几何绘图设定指令 - 直线

指令功能	指令码	序号	指令参数	指令说明
设定画一条直线	E0h	#nn:	X1(2), Y1(2), X2(2), Y2(2), Color(3), Width(1)	由 (X1, Y1) 到 (X2, Y2) 画一条宽为 Width、颜色为 Color 的直线。 

例如在 [UartTFT.ini](#) 输入下面的画直线指令：

E0h #00: 40, 40, 160, 200, 0xFF0000, 3 // 显示宽为 5 的红色直线

在编译完成后，主控端的系统或是主板透过 UART 串口传递显示命令 E0h、00 给 TFT 串口屏，那么 TFT 上就会在 (40, 40) 到 (160, 200) 显示宽为 3 的红色直线。

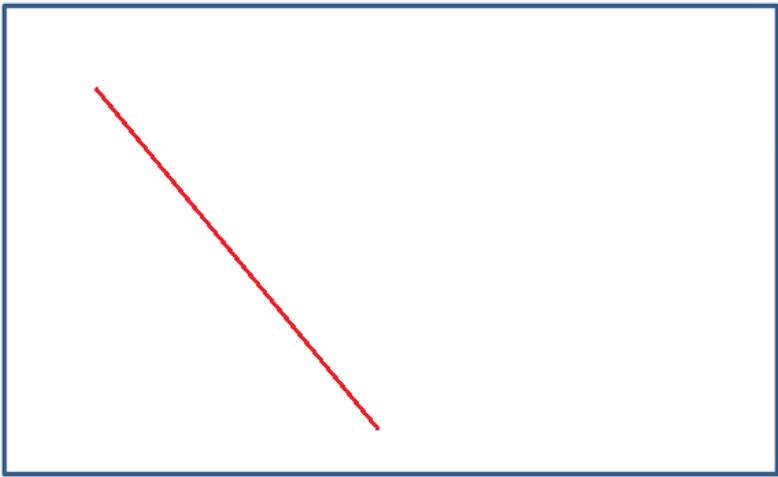


图 4-12B：画直线指令的显示范例

4.2.9 画矩形/圆角矩形指令

表 4-12：几何绘图设定指令 - 矩形

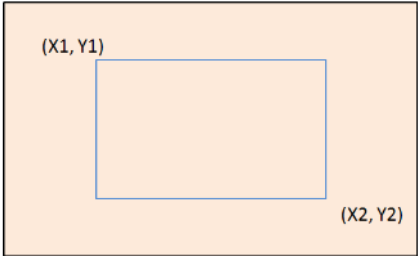
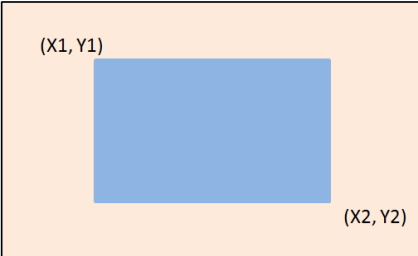
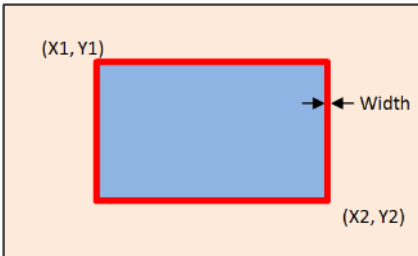
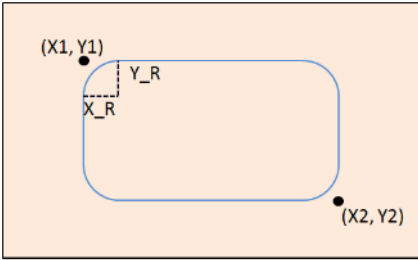
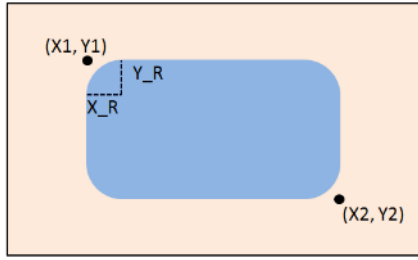
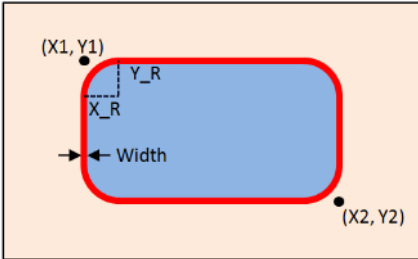
指令功能	指令码	序号	指令参数	指令说明
设定画一空心矩形	E7h	#nn:	X1(2), Y1(2), X2(2), Y2(2), Color(3)	以 (X1, Y1) (X2, Y2) 为对角、画一颜色为 Color 的空心矩形。 
设定画一实心矩形	E8h	#nn:	X1(2), Y1(2), X2(2), Y2(2), Color(3)	以 (X1, Y1) (X2, Y2) 为对角、画一颜色为 Color 的实心矩形。 
设定画一带框实心矩形	E9h	#nn:	X1(2), Y1(2), X2(2), Y2(2), Color(3), Color-F(2), Width(1)	以 (X1, Y1) (X2, Y2) 为对角、画一框颜色为 Color、宽为 Width、实心颜色为 Color-F 的带框实心矩形。 

表 4-13：几何绘图设定指令 - 圆角矩形

指令功能	指令码	序号	指令参数	指令说明
设定画一空心圆角矩形	EAh	#nn:	X1(2), Y1(2), X2(2), Y2(2), X-R(2), Y-R(2), Color(3)	以 (X1, Y1) (X2, Y2) 为对角、画一圆角 X 半径为 X-R、Y 半径为 Y-R、颜色为 Color 的空心圆角矩形。 
设定画一实心圆角矩形	EBh	#nn:	X1(2), Y1(2), X2(2), Y2(2), X-R(2), Y-R(2), Color(3)	以 (X1, Y1) (X2, Y2) 为对角、画一圆角 X 半径为 X-R、Y 半径为 Y-R、颜色为 Color 的实心圆角矩形。 
设定画一带框实心圆角矩形	ECh	#nn:	X1(2), Y1(2), X2(2), Y2(2), X-R(2), Y-R(2), Color(3), Color-F(3), Width(1)	以 (X1, Y1) (X2, Y2) 为对角、画一圆角 X 半径为 X-R、Y 半径为 Y-R、框颜色为 Color、宽为 Width、实心颜色为 Color-F 的带框实心圆角矩形。 

如上 E7h~ECh 指令所示，例如在 UartTFT.ini 输入下面的几何绘图设定指令：

E7h #00: 42, 25, 80, 105, 0x00F800

E8h #00: 120, 25, 158, 105, 0x00F800

E9h #00: 200, 25, 238, 105, 0x00F800, 0x00001F, 5

EAh #00: 42, 145, 80, 225, 15, 10, 0x00F800

EBh #00: 120, 145, 158, 225, 15, 10, 0x00F800

ECh #00: 200, 145, 238, 225, 15, 10, 0x00F800, 0x00001F, 5

当 UART 串口传递显示指令 E7h 00、E8h 00、E9h 00 给 TFT 串口屏，那么 TFT 屏就会在 (42, 25) 和 (80, 105) 2 个对角、(120, 25) 和 (158, 105) 2 个对角、(200, 25) 和 (238, 105) 2 个对角，分别产生出指定颜色的空心矩形、实心矩形及带框实心矩形。当 UART 串口传递显示指令 EAh 00、EBh 00、ECh 00 给 TFT 串口屏，那么 TFT 上就会在 (42, 145) 和 (80, 225) 2 个对角、(120, 145) 和 (158, 225) 2 个对角、(200, 145) 和 (238, 225) 2 个对角，分别产生出圆角半径为 15、10 和指定颜色的空心圆角矩形、实心圆角矩形及带框实心圆角矩形。如下图所示：

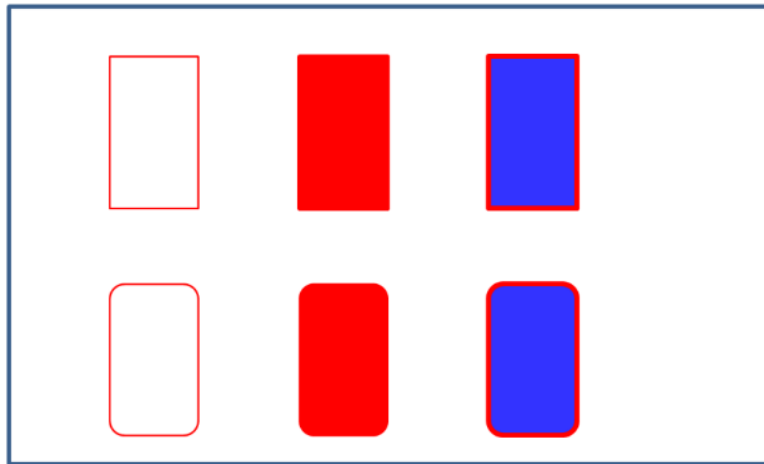
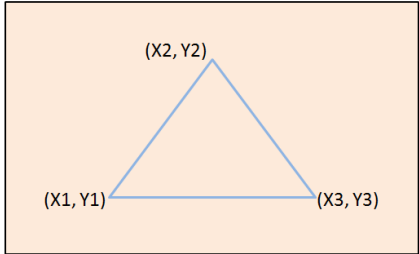
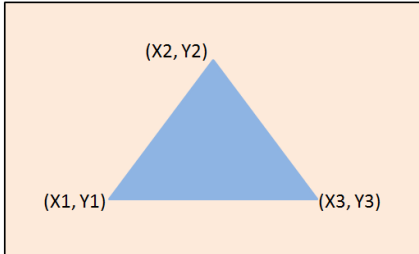
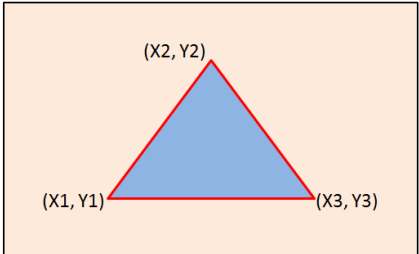


图 4-13：矩形与圆角矩形的显示范例

4.2.10 画三角形指令

表 4-14：几何绘图设定指令 - 三角形

指令功能	指令码	序号	指令参数	指令说明
设定画一空心三角形	EDh	#nn:	X1(2), Y1(2), X2(2), Y2(2), X3(2), Y3(2), Color(3)	以 (X1, Y1) (X2, Y2) (X3, Y3) 为对角、画一颜色为 Color 的空心三角形。 
设定画一实心三角形	EEh	#nn:	X1(2), Y1(2), X2(2), Y2(2), X3(2), Y3(2), Color(3)	以 (X1, Y1) (X2, Y2) (X3, Y3) 为对角、画一颜色为 Color 的实心三角形。 
设定画一带框实心三角形	EFh	#nn:	X1(2), Y1(2), X2(2), Y2(2), X-R(2), Y-R(2), Color(2), Color-F(3)	以 (X1, Y1) (X2, Y2) (X3, Y3) 为对角、画一框颜色为 Color、实心颜色为 Color-F 的带框实心三角形。 

如上指令 EDh~EFh 所示，例如在 UartTFT.ini 输入下面的几何绘图设定指令：

EDh #00: 80, 25, 40, 105, 120, 105, 0x00F800

EEh #00: 160, 25, 120, 105, 200, 105, 0x00F800

EFh #00: 240, 25, 200, 105, 280, 105, 0x00F800, 0x00001F

当 UART 串口传递显示指令 EDh 00、EEh 00、EFh 00 给 TFT 串口屏，那么 TFT 上就会在个别指定的三个坐标位置显示出设定颜色的空心三角形、实心三角形及带框实心三角形。如下图所示：

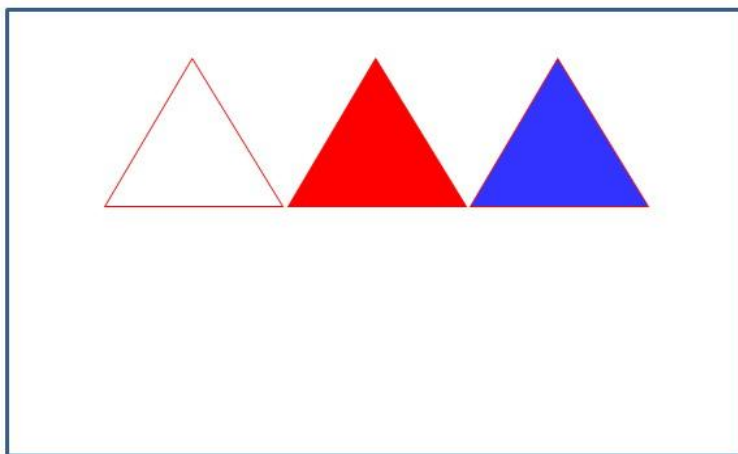
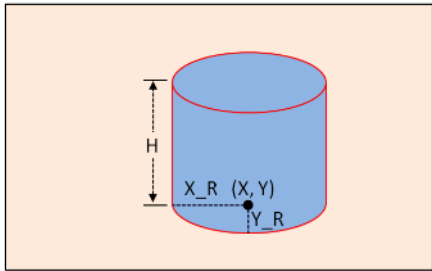
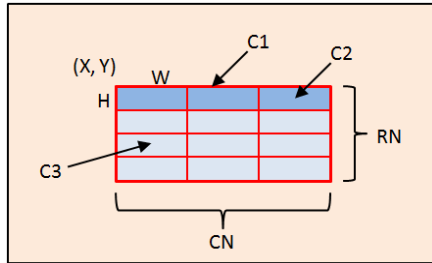
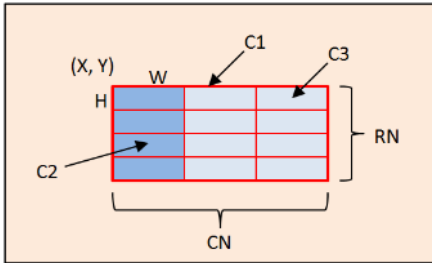


图 4-14：三角形指令的显示范例

4.2.11 画圆柱状体、画表格指令

表 4-15: 几何绘图设定指令 -圆柱体

指令功能	指令码	序号	指令参数	指令说明
设定画圆柱体	F4h	#nn:	X(2), Y(2), X-R(2), Y-R(2), Height(2), Color(3), Color-F(3), Width(1)	<p>以 (X, Y) 为底圆心、画一底圆心 X 半径为 X-R、Y 半径为 Y-R、高度为 Height、框颜色为 Color、实心颜色为 Color-F 的圆柱体，框的宽度为 Width。</p> 
设定画表格视窗	F6h	#nn:	X(2), Y(2), Width(2), Height(2), RN(1), CN(1), C1(3), C2(3), C3(3), I-Width(1), O-Width(1), Mode(1)	<p>以 (X, Y) 为起始、画一宽度为 Width、高度为 Height、行数为 CN、列数为 RN、线框颜色 C1、项目栏背景色 C2、内部窗口背景色 C3、内框宽度为 I-Width、外框宽度为 O-Width 的纵向 (Mode=0) 或是横向 (Mode=1) 表格视窗。</p>  

如上指令 F4h、F6h 所示，例如在 UartTFT.ini 输入下面的几何绘图设定指令：

F4h #00: 256, 148, 35, 20, 110, 0x00F800, 0x00001F, 3

F6h #00: 20, 25, 35, 15, 5, 6, 0x00F800, 0x0007E, 0x00001F, 2, 4, 0

当 UART 串口传递显示指令 F4h 00 给 TFT 串口屏，那么 TFT 上就会在指定的坐标位置显示出设定颜色的圆柱体。当 UART 串口传递显示指令 F6h 00 给 TFT 串口屏，那么 TFT 上就会在 (20, 25) 起的坐标位置显示出设定颜色的表格视窗。下图为圆柱体与表格视窗指令的显示范例：

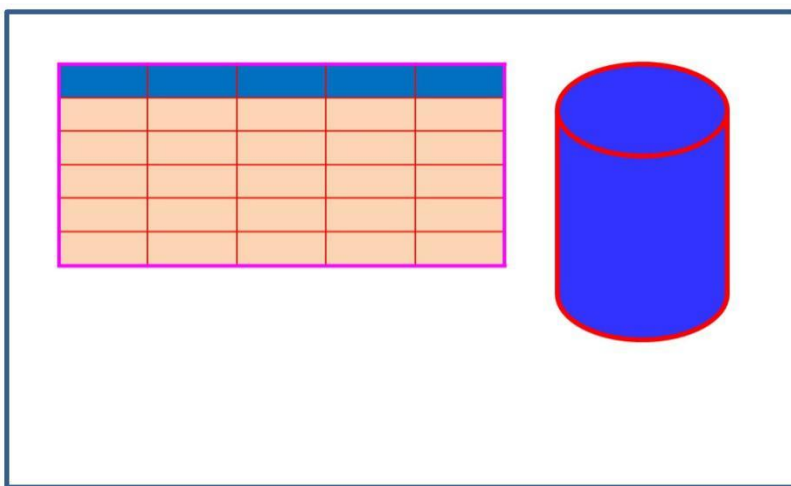


图 4-15：圆柱体、方柱体与表格视窗指令的显示范例

4.2.12 开机设定指令

开机设定指令用来设定串口屏开机后自动执行的指令，它不需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会在开机时自动执行。

表 4-16A：开机设定指令

指令功能	指令码	序号	指令参数	指令说明
设定开机后执行的指令	9Ah	00	CM1(2), NU1(2), CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8	设定在开机后执行的指令。此指令可以在串口屏开机后同时执行最多 8 组指令。

当在指令文件 (UartTFT.ini) 设置 9Ah、#00 指令后，UART 串口屏上的 LT268B 就会在开机时自动执行 9Ah、#00 指令所列的指令，例如 UartTFT.ini 输入下面的设定指令：

9Ah #00: 0x80, 0x02, 0x81, 0x01, 0x88, 0x00

那么 TFT 串口屏开机时等同于执行主控端透过 UART 串口传递显示指令 80h 及 02、循环显示重迭图片显示指令 81h 及 01、及播放 GIF 指令 0x88, 0x00 指令。此指令目的是让主控端尚未连结或是传达信息给 TFT 串口屏前，能让串口屏上先显示开机画面。

4.2.13 合并指令

合并指令是用来设定执行多组的命令。

表 4-16B：合并设定指令

指令功能	指令码	序号	指令参数	指令说明
设定执行多组的命令	9Ah	#nn:	CM1(2), NU1(2), CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8	当 nn 不为 00 时，此指令为一命令组合，可以在同时执行最多 8 组命令。

另外 9Ah、#nn 指令也可以用来设定执行多组的命令，例如 UartTFT.ini 输入下面的设定指令：

9Ah #01: 0x80, 0x03, 0x81, 0x02, 0x88, 0x01

当主控端透过 UART 串口传递 9Ah 及 01 后，串口屏就会执行显示命令 80h 及 02、循环显示重迭图片显示命令 81h 及 02、及播放 GIF 指令 0x88, 0x01 指令。此指令目的用于需要执行重复的命令，可以简化文字编译程序。

4.2.14 图片卷动设定及显示指令

图片卷动控制指令主要是提供图片能在 TFT 屏某个显示区内进行上、下、左、右不同方向的循环卷动，达到画面更生动的效果。

表 4-17：图片卷动设定指令

指令功能	指令码	序号	指令参数	指令说明
设定卷动出现图片	D8h	#nn:	X(2), Y(2), Dir(1), Speed(1), Paa(2)	<p>卷动出现单张图片到达在 (X, Y) 位置；aa 代表图片编号。Dir 代表是上下左右出现方向 → 0: 向上；1: 向下；2: 向左；3: 向右。Speed 代表图片卷动速度（单位：10ms，也就是每卷动出现 4 个像素的时间），Speed 的设置范围为 1~99。</p> 
设定循环卷动图片	D9h	#nn:	X(2), Y(2), Dir(1), Speed(1), Paa(2), Pbb, Pcc	<p>在 (X, Y) 位置循环卷动多张图片；aa/bb/cc 代表图片编号。Dir 代表是上下左右卷动方向 → 0: 向上卷动；1: 向下卷动；2: 向左卷动；3: 向右卷动。Speed 代表图片卷动速度（单位：10ms，也就是卷动时移位 1 个像素的时间），Speed 的设置范围为 1~99。</p> 
取消显示循环卷动图片	DBh	nn		将 D9h 显示循环卷动的图片取消

例如在 UartTFT.ini 输入下面的卷动出现单张图片设定指令：

D8h #00: 60, 200, 0, 2, 6 // 在 (60, 200) 以 20ms 向上卷动 4 个像素的速度显示 P6

在编译完成后，主控端的系统或是主板透过 UART 串口传递显示指令 D8h、00 给 TFT 串口屏，那么 TFT 上就会在 (60, 200) 向上卷动显示出 P6 图片。注意，要卷动出现的图片不能超过 TFT 显示区域。

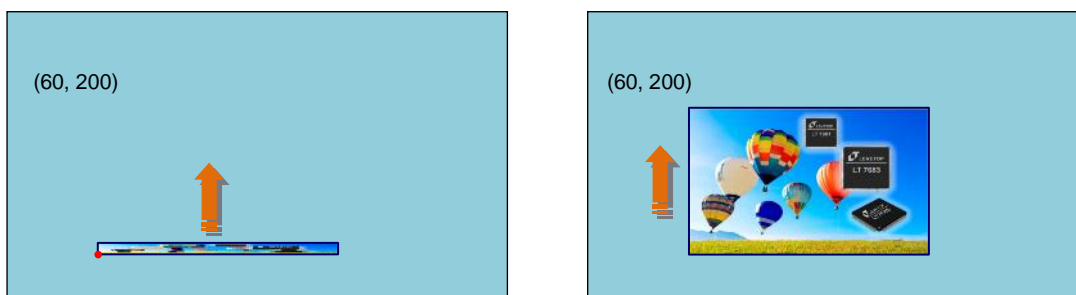


图 4-16：显示卷动出现范例

例如在 UartTFT.ini 输入下面的循环卷动图片设定指令：

D9h #00: 80, 50, 0, 2, 5, 6 // 在 (80, 50) 以 20ms 向上卷动 1 个像素的速度循环
// 显示 P5, P6

D9h #01: 150, 60, 2, 5, 7, 8, 9 // 在 (150, 60) 以 50ms 向左卷动 1 个像素的速度循环
// 显示 P7, P8, P9

在编译完成后，主控端的系统或是主板透过 UART 串口传递显示指令 D9h、00 给 TFT 串口屏，那么 TFT 上就会在 (80, 50) 向上卷动循环显示 P5、P6 图片。如果传递显示指令 D9h、01 给 TFT 串口屏，那么 TFT 上就会在 (150, 60) 向左卷动循环显示 P7、P8、P9 图片。注意，卷动的图片必须要大小一致，同时不能超过 TFT 显示区域。

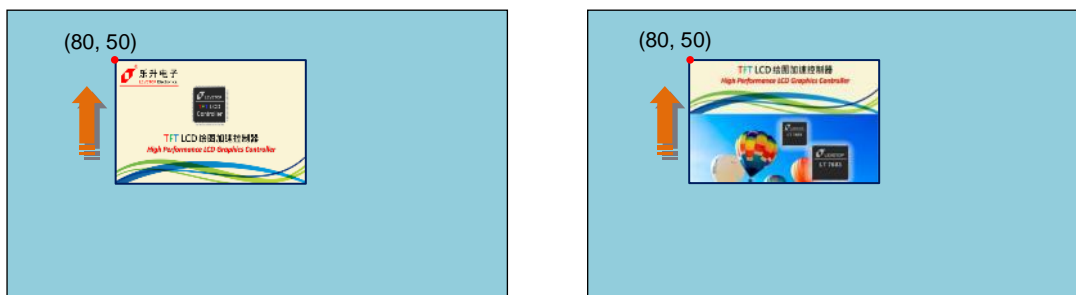


图 4-17：显示图片卷动范例

主控端的系统或是主板透过 UART 串口传递显示命令 DBh 及 00 给 TFT 串口屏，那么原 D9h、00 显示的图片卷动将被取消。

注意：弹出图片 (D8h)、循环卷动 (D9h) 与前面提到的弹出图片循环显示 (81h)、GIF 动画 (88h) 这些指令在 TFT 屏上的显示区域要避免重迭，以免造成画面交互显示及错乱！

4.2.15 环形绘图设定及显示指令

环形绘图指令主要是提供在 TFT 屏显示一环状指标图。

表 4-18：环形指标设定及显示指令

指令功能	指令码	序号	指令参数	指令说明
设定环形指标图	DCh	#nn:	X(2), Y(2), R(2), Width(2), Color(3), Speed(1)	以 (X, Y) 为中心点位置, 显示半径为 R; 宽度为 Width; 颜色为 Color; Speed 代表是指标显示速度 (单位: 1ms); 角度为 Angle 的环形指标图。

例如在 UartTFT.ini 输入下面的环状指针图设定指令：

```
DCh #00: 160, 100, 45, 8, 0xFF0000, 5    // 在 (160, 100) 为中心点位置, 显示
                                           // 半径为 45; 显示宽度为 8; 以 5ms
                                           // 显示红色的环形图。
```

在编译完成后, 主控端的系统或是主板透过 UART 串口传递显示指令 DCh、00、0000、010E 给 TFT 串口屏, 那么 TFT 上就会在 (160, 100) 为中心点, 半径为 45 位置以 5ms 由 0° 位置显示出 270° 红色的环形指标图。

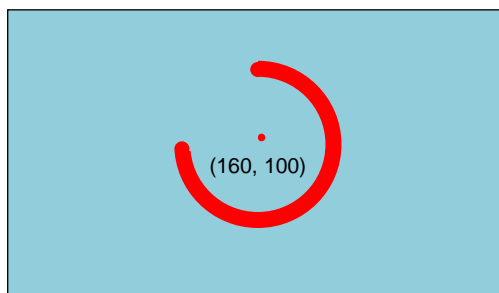
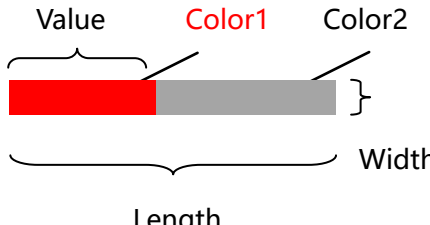


图 4-18：显示环形指标图范例

4.2.16 进度条设定及显示指令

进度条指令主要是提供在 TFT 屏显示一长方形条状指标图。

表 4-19：进度条指标设定及显示指令

指令功能	指令码	序号	指令参数	指令说明
设定进度条指标图	B0h	#nn:	X(2), Y(2), Dir(1), Width(1), Color1(3), Mark(1), COM(2), Length(2), Color2(3)	<p>在(X, Y)位置画颜色为 Color1 的进度条; Dir 代表是上下左右出现方向 → 0: 向上; 1: 向下; 2: 向左; 3: 向右。Width 代表是进度条宽。Mark 去选择用发生 80 指令(COM)做背景图还是只依据 Length 及 Color2 做底图, 当 Mark=1 时用发送 80 指令方式做背景图, Mark=0 则依据 Length 及 Color2 只做矩形进度条底图, 其中 Length 是底图的长度, Color2 为底图颜色。</p> 

例如在 UartTFT.ini 输入下面的进度条图设定指令：

B0h #00: 55, 120, 3, 20, 0xFF0000, 0, 0x80, 0x05, 200, 0x00FF00

// 在 (55, 120) 位置, 往右显示宽为 20 Pixel 的红色 (0xFF0000) 进度条;

// 底图为总长 200 的绿色 (0x00FF00) 矩形

// Mark = 0, 所以 0x80, 0x05 的背景图指令会被忽略

在编译完成后, 主控端的系统或是主板透过 UART 串口传递显示指令 B0h、00、00、40 给 TFT 串口屏, 那么 TFT 上就会在 (55, 120) 位置, 显示长度为 64 Pixel (0040h = 64) 的红色进度条指标图, 而进度条的底图为长 200 的绿色矩形, 当 UART 串口再传递显示指令 B0h、00、00、C0 给 TFT 串口屏, 那么红色进度条指标变长为 192 Pixel (00C0h = 192), 如下图所示。

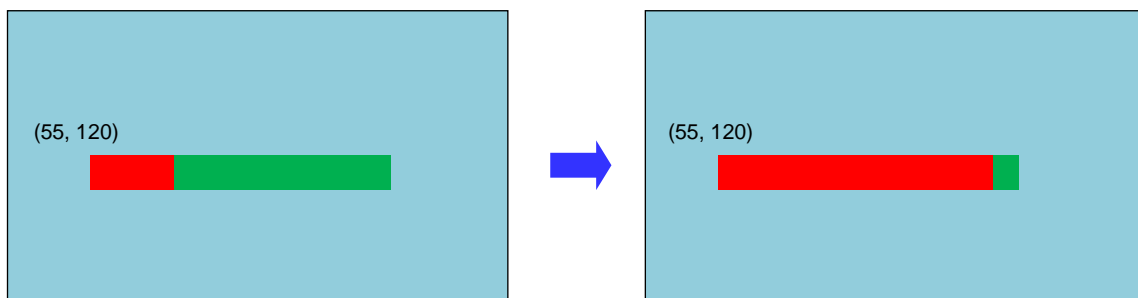


图 4-19：显示进度条指标图范例 1

如果把 Mark 设成 1，例如在 `UartTFT.ini` 输入下面的进度条图设定指令：

```
B0h #00: 85, 140, 3, 35, 0xFF0000, 1, 0x80, 0x05, 220, 0x000000
```

// 在 (85, 140) 位置，往右显示宽为 35 Pixel 的红色 (0xFF0000) 进度条；

// 进度条总长为 220

// Mark = 1，所以进度条切换数值时会自动引用 0x80, 0x05 指令所产生的背景图

// 0x80, 0x05 为显示图片指令，必须在指令文件内设置好，如下：

```
80h #05: 6, 0, 80, 80
```

// 在 (80, 80) 位置显示 P6 彩色背景图



在编译完成后，主控端的系统或是主板透过 UART 串口传递显示指令 `B0h, 00, 00, 96` 给 TFT 串口屏，那么 TFT 上就会在 (85, 140) 位置，显示长度为 150 Pixel (0096h = 150) 的红色进度条指标图，而进度条的底图为 `0x80, 0x05` 指令所产生的，当 UART 串口再传递显示指令 `B0h, 00, 00, 3C` 给 TFT 串口屏，那么红色进度条指标变长为 60 Pixel (003Ch = 60)，背景图不会变化，如下图所示。

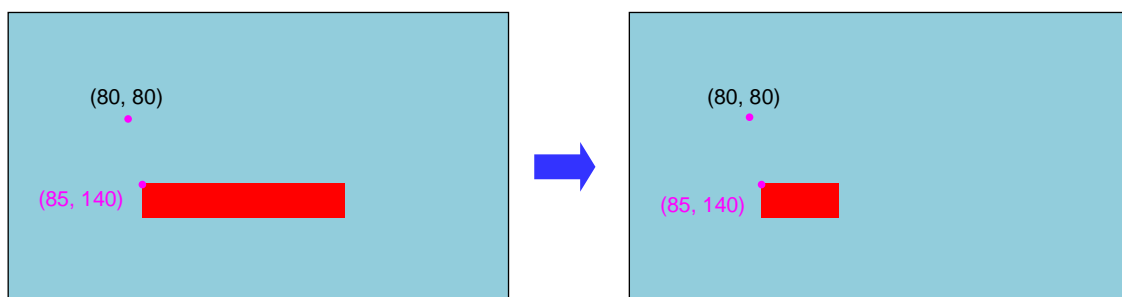


图 4-20：显示进度条指标图范例 2

4.2.17 电阻屏控制指令

电阻式触控屏控制指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端系统或是主板的固定指令，这指令不需要在 UartTFT_Tool 上位机软件或是编程软件设定。

表 4-20：电阻式触控屏控制指令

指令功能	指令码	序号	指令参数	指令说明
进行电阻屏校对	8Bh	--		进行电阻屏的 4 个角落校验点。

触控屏控制指令不需要写在 UartTFT.ini 内，例如当 UART 串口传递控制指令 8Bh 给 TFT 串口屏，那么就会自动进入电阻屏的 4 个角落校验程序，使用者必须依照显示的角落点击完成电阻式触控屏校验程序。

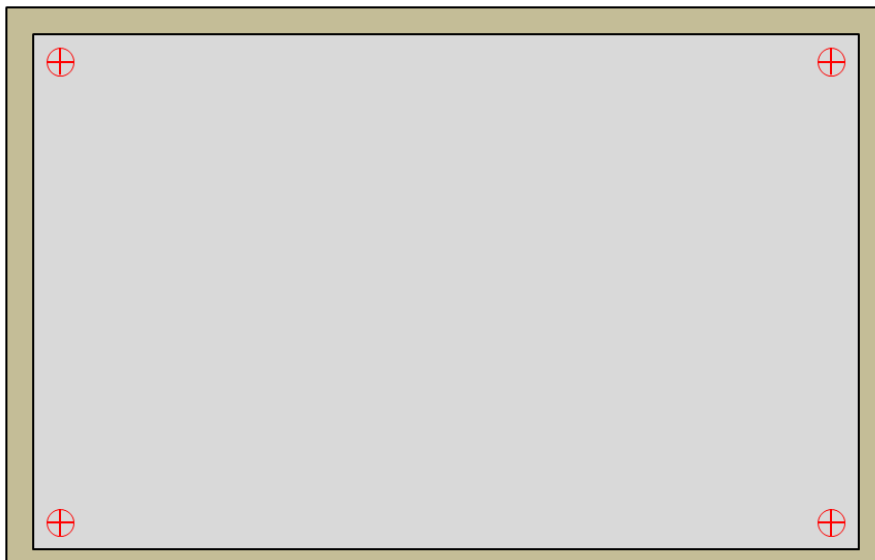


图 4-21：电阻屏的 4 个角落校验

4.2.18 背光控制指令

使用 LT268B 的 PWM 输出，分别可以用在控制背光亮度，背光控制指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端系统或是主板的固定指令，这指令不需要在 UartTFT_Tool 上位机软件或是 UartTFT.ini 内设定。

表 4-21：背光控制指令

指令功能	指令码	序号	指令参数	指令说明
调整背光亮度	BAh	--	BL	透过 PWM1 控制背光亮度。其中 BL 代表背光亮度 (00~0Fh)，是 1 个 Byte 的 16 进制单位；数值越大代表背光越亮。
显示屏 On/Off	BCh	--	0/1	0：设定 Display Off； 1：设定 Display On；

调整背光亮度指令不需要写在 UartTFT.ini 内，例如当 UART 串口传递指令 BAh、05 给 TFT 串口屏，那么 TFT 串口屏的背光就变暗，当 UART 串口传递指令 BAh、0E 给 TFT 串口屏，那么 TFT 串口屏的背光就变亮。

当 UART 串口传递指令 BCh、00 给 TFT 串口屏，那么 TFT 串口屏的显示会被关闭 (Display Off)；传递指令 BCh、01 给 TFT 串口屏，那么 TFT 串口屏的显示会被开启 (Display On)。注意，显示屏 Off 后，TFT 串口屏将不接受任何绘图功能的指令，直到串口屏收到显示屏 On 指令。

4.2.19 串口屏侦测指令

串口屏侦测指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端系统或是主板的固定指令，这指令不需要在 UartTFT_Tool 上位机软件或是 UartTFT.ini 内设定。

表 4-22：起始控制指令

指令功能	指令码	序号	指令参数	指令说明
检查 TFT 串口屏	BEh	--		串口屏初始化完成或是处于 Ready 状态： 5Ah → Ready; 55h → Not Ready（忙状态）;
检查 TFT 串口屏版本	BFh	--		读取 LT268B 串口屏版本信息：MCU Code 版本(5Bytes) + 串口屏模块。 (42Bytes)

当 UART 串口传递指令 BEh 给 TFT 串口屏，如果串口屏初始化完成或是处于 Ready 状态，串口屏会回应 5Ah 给主控端。如果串口屏是处于 Busy 忙状态，串口屏会回应 55h 给主控端。若是 TFT 串口屏与主控端的 Uart 通讯口未连接则不会响应任何信息。

串口指令 BFh 是用来读取 LT268B 的 TFT 串口屏版本，包括 5 个 Byte 的 LT268B 程序码版本，及 42 个 Bytes 的串口屏模块信息，如下表 4-23。当 UART 串口传递指令 BFh 给 TFT 串口屏，串口屏则依序送出这些信息给主控端 MCU。

表 4-23：串口屏版本信息

Items	No.	定义	说明	举例	
MCU Code 版本内容	1	年		07	07E3=2019
	2			E3	
	3	月		07	7 月
	4	日		05	5 日
	5	自定义号 0~255		CC	
串口屏保存信息	6	上位机版本号	0x10 代表 V1.0, 0x11 代表 V1.1, 0x12 代表 V1.2	10	V1.0
	7	通信接口	0x00-Uart, 0x01-SPI, 0x02-IIC	00	Uart
	8	通信接口速率	高位在前, 低位在后	01	01C200=115200
	9			C2	
	10			00	
	11	PCB 版本号	0x10 代表 V1.0, 0x11 代表 V1.1, 0x12 代表 V1.2	10	V1.0
	12	MCU	0x00 代表 8 位单片机, 0x01 代表 16 位, 0x02 代表 32 位	02	32bit
	13	MCU 厂家	0x00: Levetop, 0x01: ST, 0x02 = STC	01	Levetop
	14	Part Number	0x00: 7680A, 0x01: 7680B, 0x02: 7681, 0x03: 7683, 0x04: 7686, 0x05: 7688, 0x06: 268A, 0x07: 268B,	07	LT268B
	15	LT268B 与 TFT 屏通信接口	MCU to LTx68 I/F: 0x00 = 8080-8, 0x01 = 8080-16, 0x02 = SPI, 0x03 = I2C	02	SPI
	16	水平像数		01	01E0=480
	17			E0	
	18	垂直像数		01	0110=272
	19			10	
	20	VBPD		00	0014=20
	21			14	
	22	VFPD		00	000C=12
	23			0C	
	24	VSPW		00	0003=3
	25			03	
	26	HBPD		00	008C=140
	27			8C	
	28	HFPD		00	00A0=160
	29			A0	
	30	HFPW		00	0014=20
	31			14	
	32	PCLKRISING		01	1
	33	HSYNCPolarity		00	0
	34	VSYNCPolarity		00	0
	35	DEPolarity		01	1
	36	RGBSequence	000b : RGB,001b : RBG,010b : GRB,011b : GBR,100b : BRG,101b : BGR	00	RGB
	37	ColorDepth	0x00: 8bits, 0x01: 16bits, 0x02: 24bits	01	16Bits
	38	FlashType	0x00: NorFlash, 0x01: NandFlash	00	NOR Flash
	39	FlashSize	单位: 1MBytes	00	0080=128Mbyte
	40			80	
	41	UI Editor/UartTFT_Tool 编译工程版本	0x06: 06 (ex: 06_04_2019-09_24_53)	07	7 月
	42		0x04: 04	05	5 日
	43		0x14: 20	14	2019
	44		0x13: 19	13	
	45		0x09: 09	0A	时
	46		0x18: 24	11	分
	47		0x35: 53	25	秒

4.2.20 二维码 (QR-Code) 设定及显示指令

LT268B 的 TFT 串口屏支持显示二维码功能，它可以将主控端系统或是主板的文字串藉由 MCU 软件产生二维码，并由 LT268B 的图形显示功能秀出二维码图片。

表 4-28：二维码设定及显示指令

指令功能	指令码	序号	指令参数	指令说明
设定显示二维码	98h	#nn:	X(2), Y(2), Size(1)	将主控端传来的文字变成二维码显示在 (X, Y) 位置，Size 为垂直水平放大倍数 (1~32)。
发送二维码数据	98h	nn	String	依据指令码 98h 所设定的放大倍数，将文字 String 转换成二维码图片显示在 (X, Y) 位置。

例如在文字编译程序输入下面的显示二维码设定指令：

98h #00: 100, 200, 4

98h #01: 400, 50, 8

当 UART 串口传递显示命令 98h, 0, “www.levetop.cn” 给 TFT 串口屏，那么 TFT 串口屏就会就会在 (100, 200) 的位置显示一个对应到 “www.levetop.cn” 的二维码图片，用手机二维码扫描功能就会指到 www.levetop.cn 的网址。当 UART 串口传递显示命令 98h, 01, “www.levetop.cn” 给 TFT 串口屏，那么 TFT 串口屏就会就会在 (400, 50) 的位置显示一个对应到 “www.levetop.cn” 的二维码图片。



图 4-22：显示二维码图片范例

4.2.21 声音控制指令

LT268B 的 PWM 输出可以用在推动蜂鸣器或是加上一功率放大器或是三极管来推动喇叭播放出简易音乐，声音控制指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端系统或是主板的固定命令，这些固定命令与上位机软件或是编程软件无关。

表 4-31：声音控制指令

指令功能	指令码	序号	指令参数	指令说明
播放 Wav 檔	B8h	--	REP(Bit7) + WAV	透过 PWM2 播放 Wav 音乐文件。其中 WAV 代表 Wav 档案的编号 (0x00~0x7F)。 Bit7 用来控制是否循环播放, 当 Bit7 为 0: 播放一次, Bit7 为 1: 循环播放。
停止播放 Wav 檔	B9h	--		停止播放 Wav 音乐文件。

播放 Wav 指令不需要写在指令文件内，例如当 UART 串口传递命令 B8h、00 给 TFT 串口屏，那么 TFT 串口屏的喇叭或是蜂鸣器就会播放第一个 wav 檔的音乐。当 UART 串口传递命令 B8h、81 给 TFT 串口屏，那么 TFT 串口屏就会不断循环播放第二个 wav 檔的音乐。当 UART 串口传递命令 B9h 给 TFT 串口屏，那么 TFT 串口屏就会停止播放 Wav 音乐文件的音乐。

如果要用其他的音乐格式如 MP3、Midi 等，可以在电脑上先透过其它音乐转换软件进行转换。至于 Wav 檔的 Bin 文件产生方式可以使用图文整合编译器 (UartTFT_Tool.exe)，请参考第 4 章的 4.4.5 节。

4.2.22 时钟控制指令

时钟控制指令只需要透过 UART 或是 SPI 传递到 LT268B 的 TFT 串口屏就会执行，它是主控端的固定命令，这些固定命令与上位机软件或是编程软件无关。

表 3-33A: RTC 时钟控制指令

指令功能	指令码	序号	指令参数	指令说明
接收时钟数据	8Ch	--	Y(1), M(1), D(1), H(1), M(1), S(1), W(1)	主控端设置目前 TFT 串口屏上的时钟: 年/月/日/时/分/秒/周 数据, 共 7 Bytes。
发送时钟数据	8Dh	--		回应主控端目前 TFT 串口屏上的时钟信息: 年/月/日/时/分/秒/周 数据。

时钟控制指令不需要写在指令文件内，例如当 UART 串口传递命令 8Dh 给 TFT 串口屏，那么 TFT 串口屏就会将串口屏上的时钟数据 - 年/月/日/时/分/秒/周 上传给主控端。

4.2.23 指令总表

下表 ☐ 设定指令是使用 UartTFT_Tool 时在 UartTFT.ini 文件内所描述的指令设定格式；而 ☐ 显示/控制指令 则是主控端发送给 TFT 串口屏的指令设定格式。

表 4-24：指令总表

☐：设定指令 ☐：显示/控制指令

指令码	序号	指令参数	回应参数	指令功能
80h	#nn:	Paa, PNGaa, Xaa, Yaa, Pbb, PNGbb, Xbb, Ybb, Pcc, PNGcc, Xcc, Ycc		设定显示单张或多张图片
80h	nn			显示图片
81h	#nn:	Delay, X, Y, PNG, Paa, Pbb, Pcc		设定循环显示重迭图片
81h	nn			循环显示重迭图片
84h	nn			取消循环显示重迭图片
88h	#nn:	Delay, X, Y, GIF		设定显示 GIF 图片
88h	nn			显示 GIF 图片
89h	nn			取消显示 GIF 图片
8Ah	#nn:	Paa, PNGaa, Xaa, Yaa, Pbb, PNGbb, Xbb, Ybb, Pcc, PNGcc, Xcc, Ycc		设定显示单张或多张图片
8Ah	nn			显示单张或多张图片
8Bh				进行电阻屏校验
8Ch		Y, M, D, H, M, S, W		设定时钟 年/月/日/时/分/秒/周
8Dh			Y, M, D, H, M, S, W	回应时钟 年/月/日/时/分/秒/周
8Fh	nn	X, Y, PNG, Pnn		显示单一图片
90h	#nn:	PT, X, Y, Dir, Color-F, Color-B, EN-B		设定显示图片式（客制化）的数字
90h	nn	ddd.d		显示图片式的数字
98h	#nn:	X, Y, Size		设定显示二维码
98h	nn	String		显示二维码
9Ah	#nn:	CM1, NU1, CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8		设定执行多组的指令(最多 8 组), 当 nn = 00 时为开机后执行的指令
9Ah	nn			执行多组的指令, 当 nn = 00 时为执行开机指令

表 4-24: 指令总表 (续)

指令码	序号	指令参数	回应参数	指令功能
A0h	#nn:	P, PNG, X, Y, CM1, NU1, CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8	ID, Status	设定显示控件图片, 及在按下触控后执行的指令 (8 组)
A0h	nn			显示控件图片
A1h	nn			取消显示控件图片及功能
A2h	#nn:	X, Y, Width, Height, CM1, NU1, CM2, NU2, CM3, NU3, CM4, NU4, CM5, NU5, CM6, NU6, CM7, NU7, CM8, NU8	ID, Status	设定虚拟控件区域, 及在按下触控后执行的命令 (8 组)
A2h	nn			设定虚拟控件
A3h	nn			取消虚拟控件及功能
B0h	#nn:	X, Y, Dir, Width, Color1, Mark, 80h Command, Length, Color2		设定进度条指标图
B0h	nn	Value (2 Bytes)		显示进度条指标图
B8h		REP(Bit7) + WAV		播放 Wav 档
B9h				停止播放 Wav 档
BAh		BL (00~0Fh)		调整背光亮度
BCh		0/1		显示 On/Off
BEh			5Ah / 55h	检查下 TFT 串口屏
BFh			版本信息: (47 Bytes)	TFT 串口屏版本侦测
C0h	#nn:	F01, X, Y, W, Color-F, Color-B, Size-H, Size-V, Transparency, Alignment		设定显示字库-1 文字
C0h	nn	String		显示字库-1 文字
C1h	#nn:	F02, X, Y, W, Color-F, Color-B, Size-H, Size-V, Transparency, Alignment		设定显示字库-2 文字
C1h	nn	String		显示字库-2 文字
C2h	#nn:	F03, X, Y, W, Color-F, Color-B, Size-H, Size-V, Transparency, Alignment		设定显示字库-3 文字
C2h	nn	String		显示字库-3 文字
C3h	#nn:	F04, X, Y, W, Color-F, Color-B, Size-H, Size-V, Transparency, Alignment		设定显示字库-4 文字
C3h	nn	String		显示字库-4 文字
D8h	#nn	X, Y, DIR, Speed, Paa		设定卷动出现图片
D8h	nn			显示卷动出现图片

表 4-24: 指令总表 (续)

令码	序号	指令参数	回应参数	指令功能
D9h	#nn	X, Y, DIR, Speed, Paa, Pbb, Pcc		设定循环滚动图片
D9h	nn			显示循环滚动图片
DBh	nn			取消循环滚动图片
DCh	#nn	X, Y, R, Width, Color, Speed		设定环形指标图
DCh	nn	S_Angle, A_Angle		显示环形指标图
DFh	#nn	Type, R, Color		设定画点
DFh	nn	X, Y		显示画点
E0h	#nn:	X1, Y1, X2, Y2, Color, Width		设定画条直线
E0h	nn			显示一条直线
E1h	#nn:	X, Y, R, Color		设定画空心圆形
E1h	nn			显示一空心圆形
E2h	#nn:	X, Y, R, Color		设定画实心圆形
E2h	nn			显示一实心圆形
E3h	#nn:	X, Y, R, Color, Color-F, Width		设定画带框实心圆形
E3h	nn			显示一带框实心圆形
E4h	#nn:	X, Y, X-R, Y-R, Color		设定画空心椭圆形
E4h	nn			显示一空心椭圆形
E5h	#nn:	X, Y, X-R, Y-R, Color		设定画实心椭圆形
E5h	nn			显示一实心椭圆形
E6h	#nn:	X, Y, X-R, Y-R, Color, Color-F, Width		设定画带框实心椭圆形
E6h	nn			显示一带框实心椭圆形
E7h	#nn:	X1, Y1, X2, Y2, Color		设定画空心矩形
E7h	nn			显示一空心矩形
E8h	#nn:	X1, Y1, X2, Y2, Color		设定画实心矩形
E8h	nn			显示一实心矩形
E9h	#nn:	X1, Y1, X2, Y2, Color Color-F, Width		设定画带框实心矩形
E9h	nn			显示一带框实心矩形
EAh	#nn:	X1, Y1, X2, Y2, X-R, Y-R, Color		设定画空心圆角矩形
EAh	nn			显示一空心圆角矩形

表 4-24: 指令总表 (续)

令 码	序 号	指令参数	回 应 参 数	指令功能
EBh	#nn:	X1, Y1, X2, Y2, X-R, Y-R, Color		设定画实心圆角矩形
EBh	nn			显示一实心圆角矩形
ECh	#nn:	X1, Y1, X2, Y2, X-R, Y-R, Color, Color-F, Width		设定画带框实心圆角矩形
ECh	nn			显示一带框实心圆角矩形
EDh	#nn:	X1, Y1, X2, Y2, X3, Y3, Color		设定画空心三角形
EDh	nn			显示一空心三角形
EEh	#nn:	X1, Y1, X2, Y2, X3, Y3, Color		设定画实心三角形
EEh	nn			显示一实心三角形
EFh	#nn:	X1, Y1, X2, Y2, X-R, Y-R, Color, Color-F		设定画带框实心三角形
EFh	nn			显示一带框实心三角形
F4h	#nn:	X, Y, X-R, Y-R, Height, Color, Color-F, Width		设定画圆柱体
F4h	nn			显示一圆柱体
F6h	#nn:	X, Y, Width, Height, CN, RN, C1, C2, C3, I-Width, O-Width, Mode		设定画表格视窗
F6h	nn			显示一表格视窗

4.3 使用 UartTFT_Tool 的设计流程

下图为用图文整合编译器 (UartTFT_Tool.exe) 开发的详细流程图, 使用者也可以至 [乐升半导体](#) 网站下载 UartTFT_Tool 的范例来操作, 将更快速的了解开发模式。同时建议使用者先依据所需功能及 TFT 屏幕大小准备好素材, 因为这些显示图片、动画文件、文字库、声音文件等是存放在 SPI Flash 内, 资料量都不小, 而 SPI Flash 的烧录所需时间较长, 因此尽量避免开发中反复对 SPI Flash 进行 UartTFT_Flash.bin 檔的烧写, 以免延误开发效率, 建议多使用 UI_Emulator 串口屏仿真器做前期验证。[UI_Emulator 串口屏仿真器软件](#)可以自 [乐升半导体 官网](#) “串口屏上位机软件” 下载专区下载。

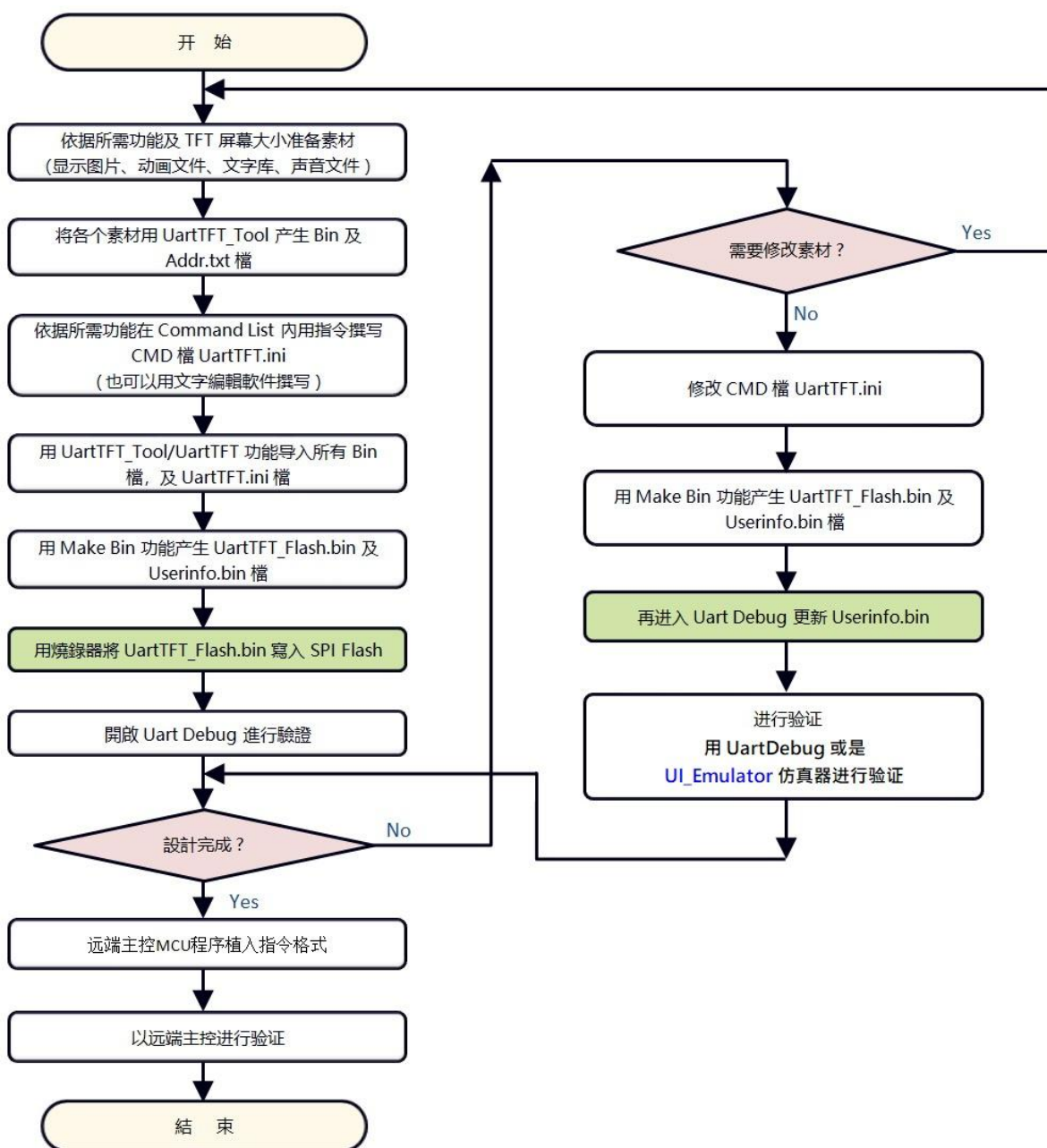


图 4-23: 使用 UartTFT_Tool 的设计流程

下图为 SPI Flash 的结构, UartTFT_Flash.bin 档实际包括了 UserInfo.bin 档的内容, 而 UserInfo.bin 档是存放 Command 参数, 其他区域则是示图片、动画文件、文字库、声音文件等数据, 如果只是修改 Command 参数, 在 Make Bin 时虽然会同时产生 UartTFT_Flash.bin 档及 UserInfo.bin 档案, 但实际上只需要更新 UserInfo.bin 档即可, UserInfo.bin 档案的大小固定为 128KBytes, 因此直接在 “UartDebug” 环境下用 USB 转 RS232 连接线快速更新即可, 不需要使用 [LT268B_ISP_Vxx.exe](#) 程序或是专用的 SPI Flash 烧录器。有关 UartDebug 的用法请参考第 5 章的说明。

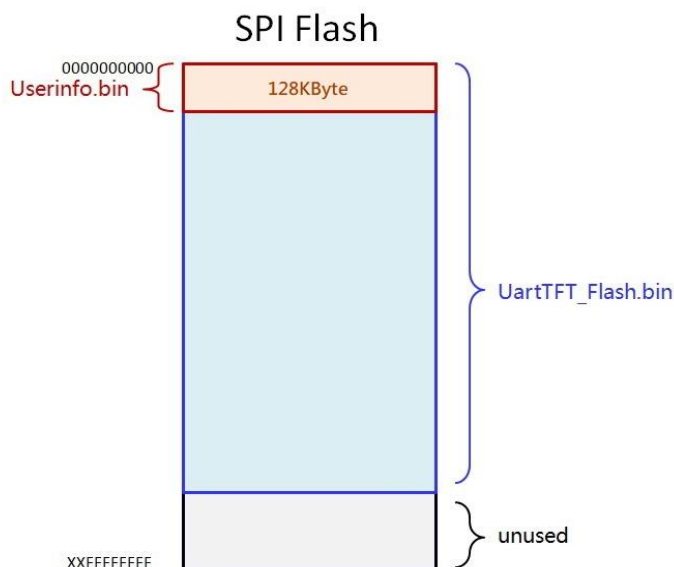


图 4-24A: SPI Flash 的结构



图 4-24B: SPI Flash 烧录器

4.4 图文整合编译器操作说明

本章节将介绍如何使用图文整合编译器 (**UartTFT_Tool.exe**) 来进行 TFT 串口屏的显示功能开发, **UartTFT_Tool.exe** 是依据 乐升半导体 的 Bin 文件整合软件 (LT_Image_Tool.exe) 加上 TFT 串口屏的指令编译器所组成的一个专用整合程序, 针对 LT268B TFT 控制器外接的 SPI Flash, 可以制作图片 Bin 文件、字库 Bin 文件、GIF 档的 Bin 文件、Wav 档的 Bin 文件, 以及制作 LT268B 的开机启动程序、图形光标, 然后将这些 Bin 文件整合起来, 产生可以烧录到 SPI Flash 的 Bin 文件, 同时可以对使用者的串口指令进行编译产生指令 Bin 文件, 当主控端的系统或是主板传递指令码及动作序号给 TFT 串口屏, 串口屏上的 LT268B 就会解析指令码, 然后读取存在 SPI Flash 内的指令动作流程去实现图片或文字的显示。

图文整合编译器总共有 6 个功能, 分别为:

- 一. TFT 串口指令编译 - **UartTFT**
- 二. 连结通讯软件 - **UartDebug.exe**
- 三. 制作「图片 Bin 文件」- **InputPicture**
- 四. 制作「GIF 档 Bin 文件」- **GIFTool**
- 五. 制作「字库 Bin 文件」- **Font**
- 六. Bin 文件整合 - **BinFile**



图 4-25: 图文整合编译器主画面

使用者可以自本公司网址 (www.levetop.cn) 下载 LT268B 的 UartTFT_Tool 演示案例 **LT268B_UartTFT_Tool_Demo_240x320.rar** 来进行操作。

4.4.1 TFT 串口指令编译

当指令文件（UartTFT.ini）用文字编辑软件完成后就可以执行 UartTFT_Tool 内的指令编译器程序，首先将本公司专用的 USB 加密器插入电脑 USB 接口，执行 UartTFT_Tool.exe，然后点击“UartTFT”进入 TFT 串口指令编译器，使用者可以参考以下的说明：

1. 点击【UartTools 菜单>UartTFT】即可打开指令编译器界面：

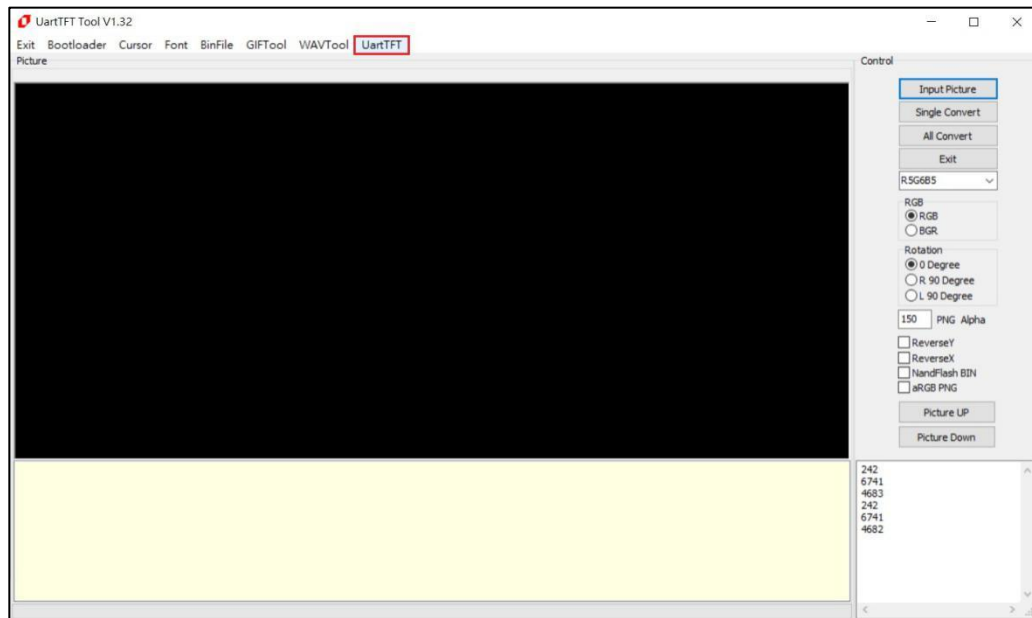


图 4-26：打开指令编译器画面

2. 点击【Input Bin Files】抓取 Bin 文件：

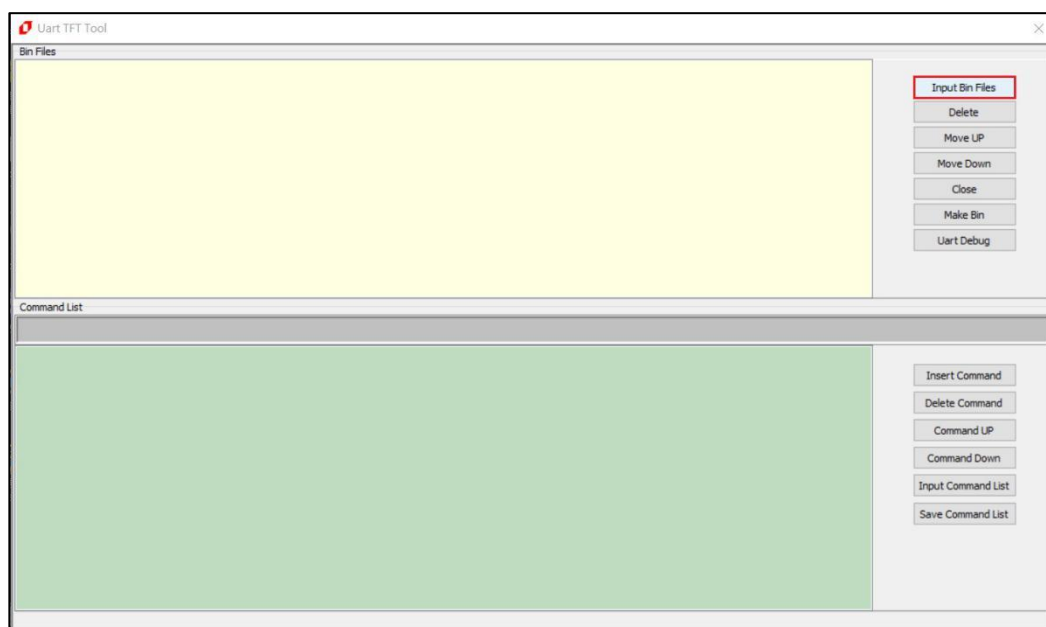


图 4-27：读取 Bin 文件

- 3、点选图形、字库等 Bin 文件：使用者应先将 Bin 文件存在同一个目录底下，点选任一 bin 档案，UartTools 会自动抓取所有 Bin 文件，使用者可以与本公司联系下载 LT268B 的 UartTFT_Tool Demo 文件，进行实际操作。

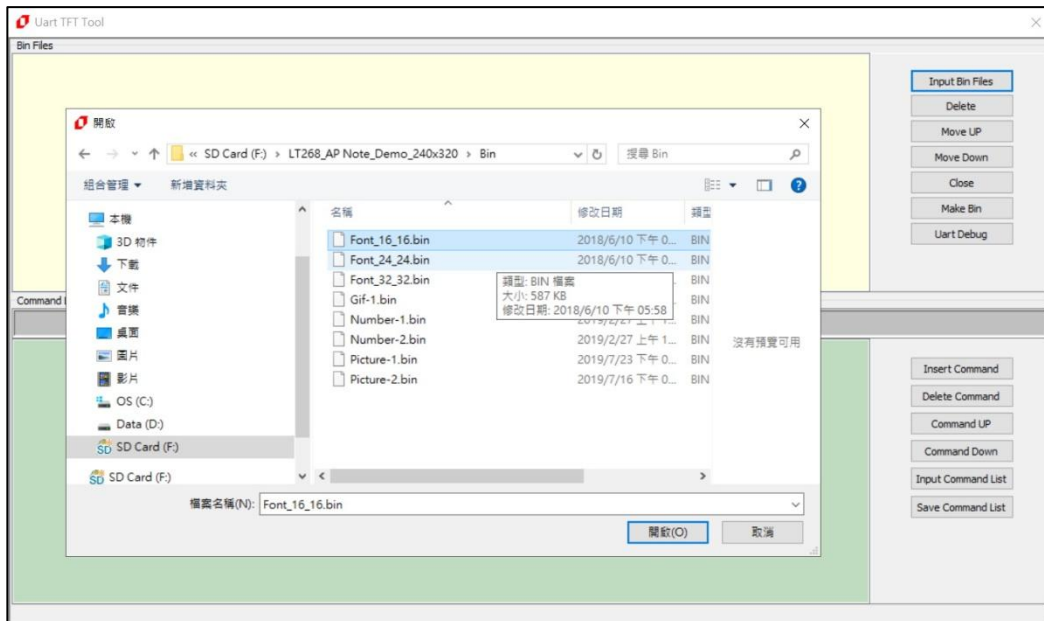


图 4-28：抓取所有 Bin 文件

- 4、UartTools 会自动要求点选输入（.ini）的指令设定文件，LT268B 的 UartTFT_Tool Demo 文件内已有 UartTFT.ini 范例（如下），用户可以使用此范例操作。

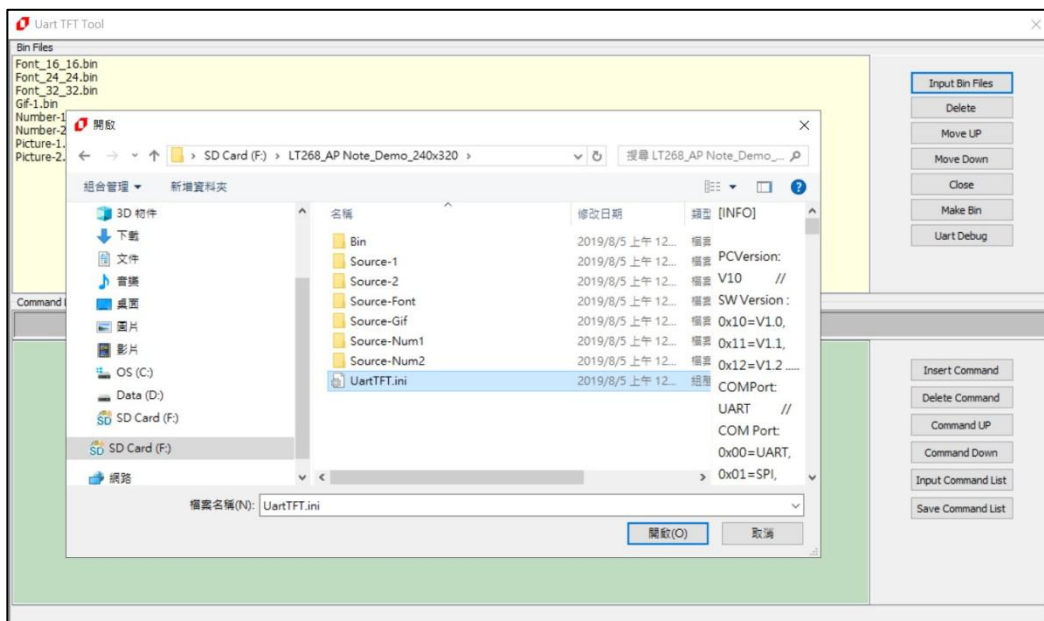


图 4-29：输入 UartTFT.ini 的指令设定文件

LT268B 的 UartTFT_Tool Demo 文件内的 UartTFT.ini 范例如下:

[INFO]

```
PCVersion: V10          // SW Version : 0x10=V1.0, 0x11=V1.1, 0x12=V1.2 .....
COMPort: UART          // COM Port: 0x00=UART, 0x01=SPI, 0x02=I2C
Baudrate: 115200        // UART Baudrate
PCBVersion: V10         // PCB Version : 0x10=V1.0, 0x11=V1.1, 0x12=V1.2 .....
MCUBit: 32              // MCU Data Bus: 0x00 = 8bit, 0x01 = 16bit, 0x02 = 32bit
MCUType: Levetop        // MCU Maker: 0x00 = Levetop, 0x01 = STM, 0x02 = STC
MCUIF: SPI              // MCU to LT268B I/F: 0x00 = 8080, 0x01 = SPI, 0x02 = I2C
768Type: 268B          // 0x00: 7680A, 0x01: 7680B, 0x02: 7681, 0x03: 7683,
                        // 0x04: 7686, 0x05: 7688, 0x06: 268A, 0x07: 268B
768IF: SPI              // MCU to LT768 I/F: 0x00 = 8080-8, 0x01 = 8080-16,
                        // 0x02 = SPI, 0x03 = I2C
XSIZE: 240              // TFT Panel X-Size
YSIZE: 320              // TFT Panel Y-Size
VBPD: 23                // Vsync Back-Porch
VFPD: 22                // Vsync Front-Porch
VSPW: 3                 // Vsync Pulse Width
HBPD: 46                // Hsync Back-Porch
HFPD: 210               // Hsync Front-Porch
HSPW: 20                // Hsync Pulse Width
PCLKRISING: 1           // 0: Panel fetches XPDAT at XPCLK rising edge, 1: falling edge
HSYNCPolarity: 0        // 0: Low active, 1: High active.
VSYNCPolarity: 0        // 0: Low active, 1: High active.
DEPolarity: 1           // 0: Low active, 1: High active.
RGBSequence: RGB        // 000b: RGB, 001b: RBG, 010b: GRB, 011b: GBR, 100b: BRG, 101b: BGR
ColorDepth: 16          // Color Depth: 0x00 = 8bit, 0x01 = 16bit, 0x02 = 24bit
FlashType: NOR           // FlashType: 0x00 = NOR Flash, 0x01 = NAND Flash
FlashSize: 128          // Flash Size, Unit: 1MByt
```

[USERCMD]

```
9Ah #00: 0x80, 0x00, 0xD9, 0x00, 0xA0, 0x00    //Boot

A0h #00: 6, 0, 79, 150, 0x80, 0x01, 0xA0, 0x01
A0h #01: 7, 1, 82, 270, 0x80, 0x02, 0x80, 0x04, 0x80, 0x08, 0xA0, 0x02    //Home
A0h #02: 7, 1, 82, 270, 0x80, 0x02, 0x88, 0x00, 0xA0, 0x03
A0h #03: 7, 1, 82, 270, 0x80, 0x02, 0xD9, 0x01, 0xA0, 0x04
A0h #04: 7, 1, 82, 270, 0x80, 0x02, 0x80, 0x09, 0x81, 0x00, 0x81, 0x01, 0x81, 0x02, 0xA0, 0x05
A0h #05: 7, 1, 82, 270, 0x80, 0x02, 0x80, 0x0A, 0xA0, 0x06
A0h #06: 7, 1, 82, 270, 0x80, 0x03, 0x81, 0x03, 0xA0, 0x07
A0h #07: 5, 1, 82, 270, 0x9A, 0x00

80h #00: 0, 0, 0, 0      // Show Picture 1
80h #01: 1, 0, 0, 0      // Show - Fruit
80h #02: 2, 0, 0, 0      // Show Picture 2
80h #03: 3, 0, 0, 0      // Show Picture 3
80h #04: 4, 1, 23, 105    // Cmp
80h #05: 5, 0, 82, 270    // Home
80h #06: 6, 0, 90, 156    // LT268B PKG
80h #07: 7, 0, 82, 270    // Next
80h #08: 8, 1, 145, 120   // Winner
80h #09: 12, 1, 31, 82    // Bar-1
80h #0A: 13, 0, 15, 85    // Font
80h #0B: 45, 0, 0, 0      // Show Picture 4
80h #0C: 46, 0, 0, 0      // Show Picture 5
```

```

80h #0D: 47, 0, 0, 0      // Show Picture 6
80h #0E: 48, 0, 0, 0      // Show Picture 7
80h #0F: 49, 0, 0, 0      // Show Picture 8

D8h #00: 58, 82, 3, 1, 10  // Pop-up a Picture
D9h #00: 0, 101, 2, 3, 9
D9h #01: 58, 82, 3, 2, 10, 11 // Show Rotate Pictures

81h #00: 4, 45, 119, 0, 14, 15, 16, 17, 18      // Show Bar1-Bar5
81h #01: 4, 135, 172, 0, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34 // Show 0~9
81h #02: 8, 56, 172, 0, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44 // Show 0~9
81h #03: 6, 50, 98, 0, 19, 20, 21, 22, 23, 24      // Show Fun1-6

88h #00: 1,1, 20, 80, GIF0      // Show GIF

90h #00: PT0, 10, 100, 0, 0x03, 0xe0, 1      // Show Graphics Text

C0h #00: 0, 10, 80, 200, 0xff0000, 0x00001f, 1,1,0,1      // Show Text 16*16
C0h #01: 0, 10, 100, 200, 0x0000FF, 0x00001f, 1,1,1,1
C1h #00: 1, 10, 120, 200, 0x00001f, 0x00f800, 1,1,0,1      // Show Text 24*24
C1h #01: 1, 10, 150, 200, 0xff0000, 0x00001f, 1,1,1,1
C2h #00: 2, 10, 180, 200, 0xff0000, 0x00001f, 1,1,0,1      // Show Text 32*32
C2h #01: 2, 10, 220, 200, 0x00FF00, 0x00001f, 1,1,1,1

```

[END]

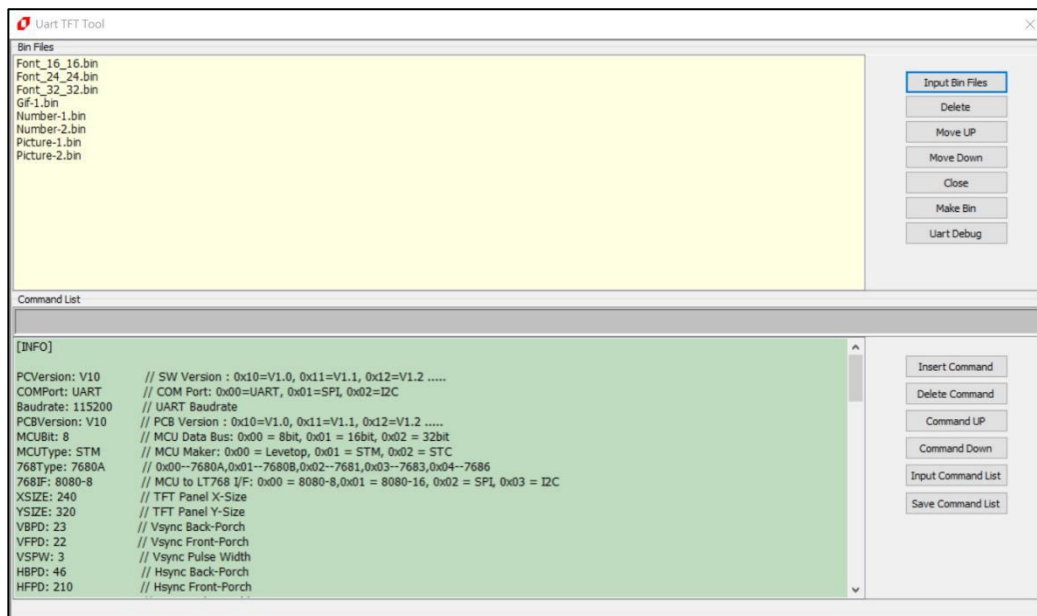


图 4-30：输入 UartTFT.ini 后的画面

- 5、点击【Make Bin】进行编译，编译后会产生 `UserInfo.bin` 及 `UartTFT_Flash.bin` 两个档案，让使用者存入指定的目录：

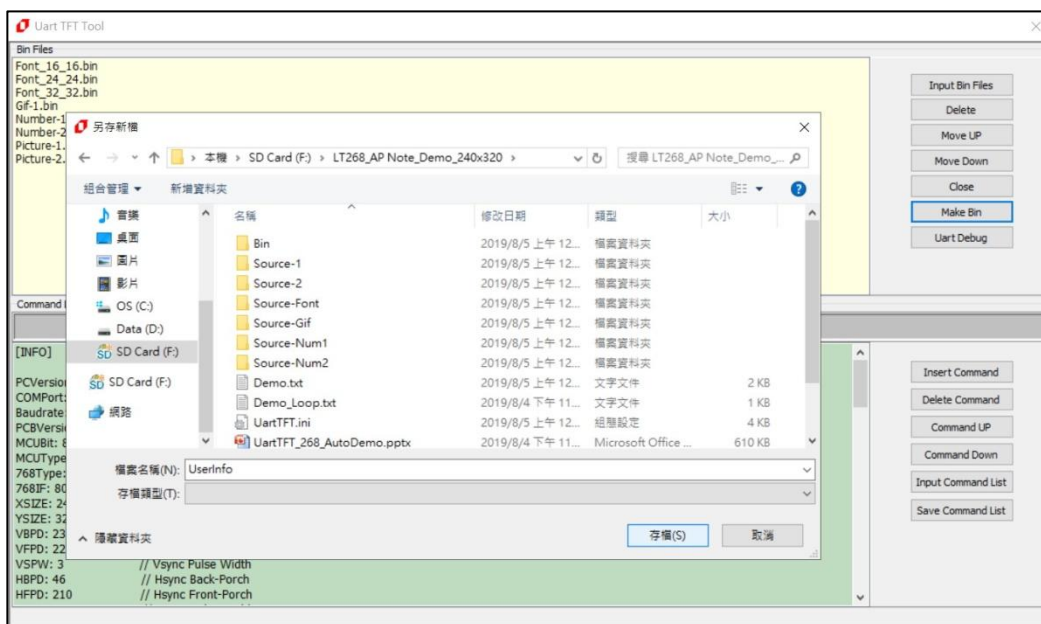


图 4-31：存入 UserInfo.bin

- 6、UartTools 自动要求接着存入 `UartTFT_Flash.bin`，同理存入指定的目录：

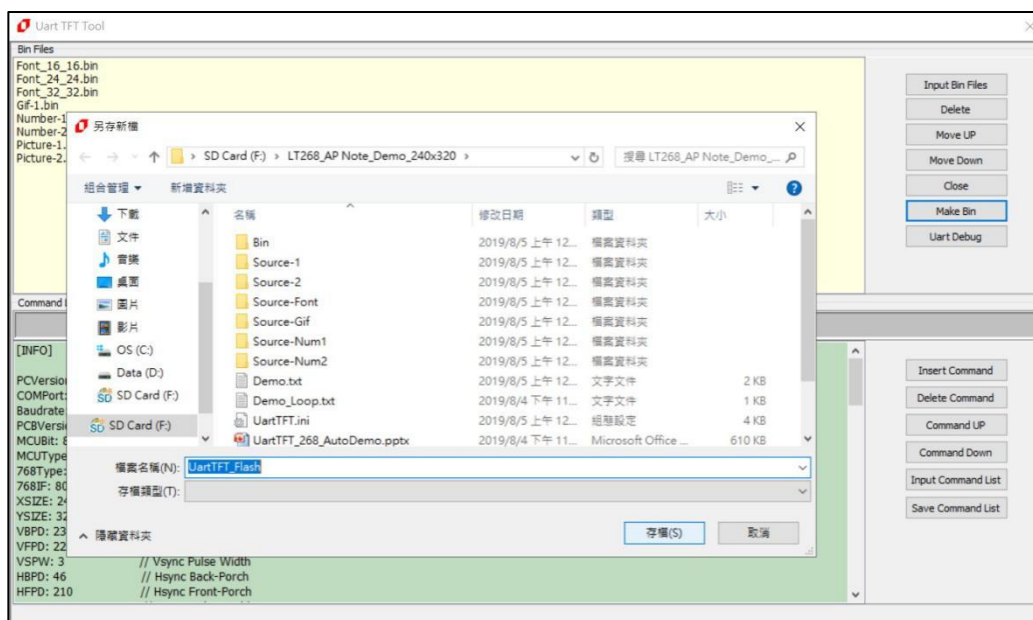


图 4-32：存入 UartTFT_Flash.bin

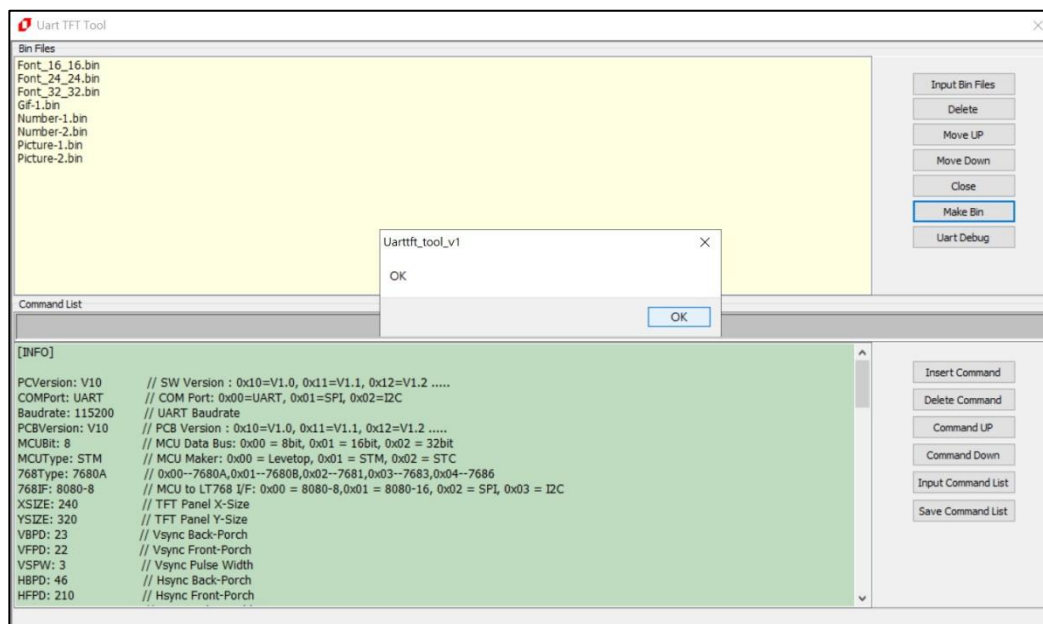


图 4-33：编译完成后的画面

- 7、使用者可以用 Command List 的编辑窗口来对指令设定文件 (UartTFT.ini) 作简易的修改、储存, 或者是由外部文件编辑器对指令设定文件 (UartTFT.ini) 修改后点击【Input Command List】重新读取指令设定文件, 修改完后回到步骤 5 重新执行【Make Bin】:

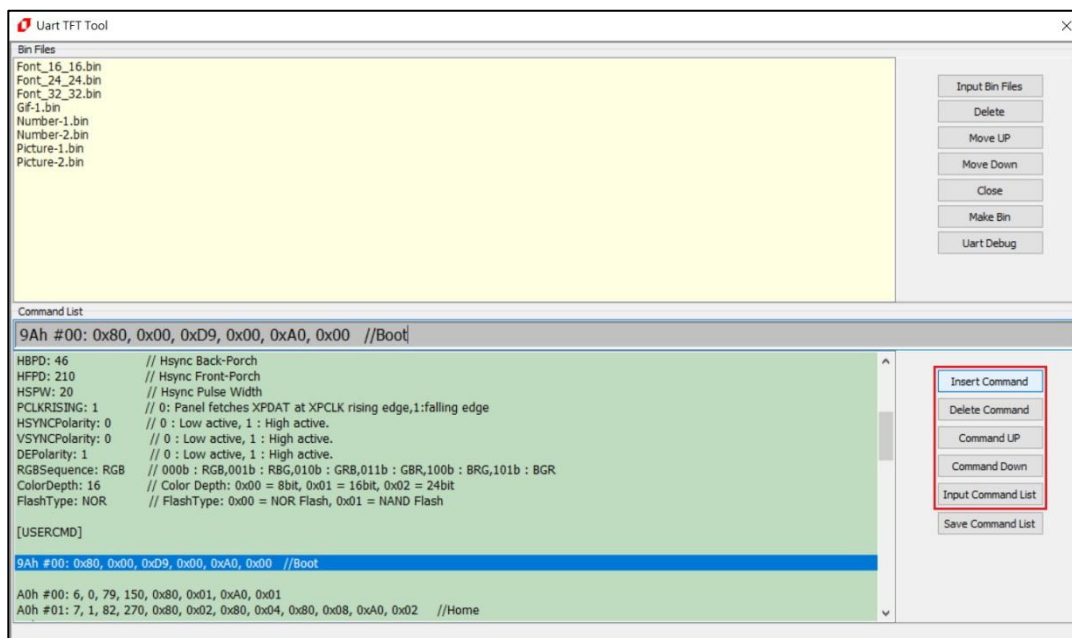


图 4-34：图文整合编译器主画面

- 8、使用 SPI Flash 烧录器将产生的 UartTFT_Flash.bin 档案烧写进 SPI Flash, 烧录完成后就代表 TFT 串口屏基本设计已经完成。

9、点击【Uart Debug】开启 UartDebug.exe 通讯软件，进行仿真主控端 MCU 的指令测试：

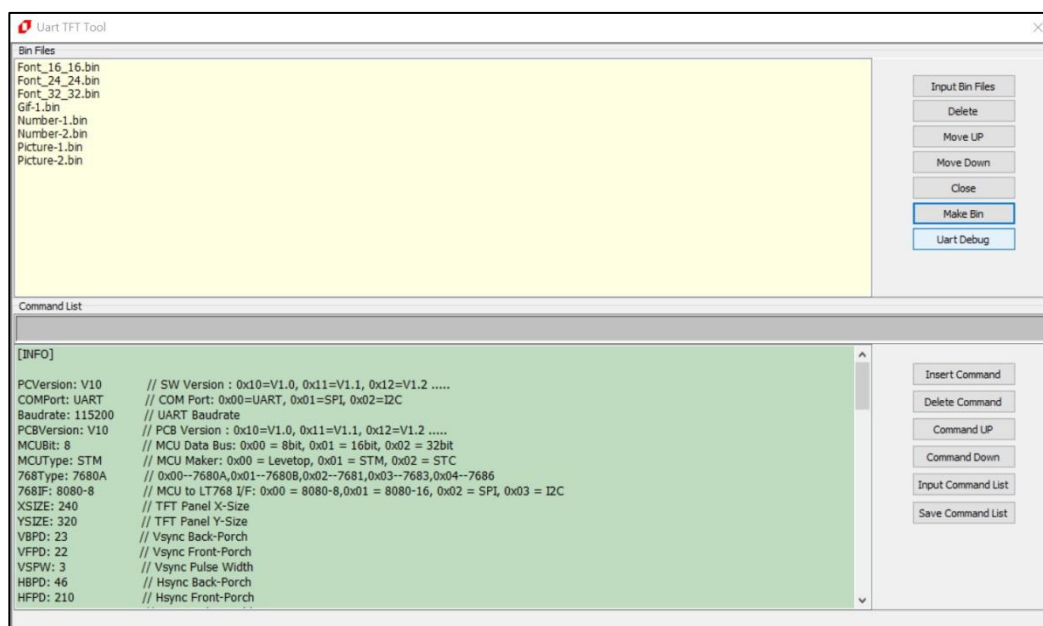


图 4-35：打开 UartDebug

UartDebug.exe 是 乐升半导体 提供的一个专用程序，主要用 PC 端的 USB 接口转成 Uart 通讯串口信号，来发送指令给 LT268B 串口屏的 Uart 接口，进而让 TFT 屏显示不同的内容，也就是在串口屏与主控端 MCU 线前做 TFT 屏显示画面的前期验证，使用者可以与本公司联系下载 LT268B 的 UartTFT_Tool Demo 文件进行实际操作，有关 UartDebug 的使用方式请参考第 5 章的说明。

- UartTFT_Flash.bin 档案比较大，主要内容是图片、字库、Gif 和指令文件的动作数据；UserInfo.bin 档案比较小约为 128K 左右，主要内容只有指令文件的动作数据，UartTFT_Flash.bin 档案实际已经包含 UserInfo.bin 档案数据，因此第一次烧写进 SPI Flash 时只需要烧写 UartTFT_Flash.bin 即可。
- 由于写进 SPI Flash 数据所花的时间较长，当使用者的图片、字库、Gif 没有变动而只修改指令文件动作，在“Make Bin”之后只需要将 128K 左右的 UserInfo.bin 档案烧写进 SPI Flash 即可，不需要将 UartTFT_Flash.bin 重新烧写进 SPI Flash，这样可以节省许多烧写 Flash 的时间及缩短开发时间。

4.4.2 制作图片的 Bin 文件

在应用端会常用到的图片，可以透过 DMA 传输方式将显示数据存到 LT268B 内建的显示内存内，这样可以降低 MCU 处理传送图像数据的负担，使用这个功能可以先将图片转成 Bin 文件，然后预先烧录在 SPI Flash 内，UartTools 提供制作图片 Bin 文件的功能，可以让使用者在 PC 端导入图片，然后生成一个图片的 Bin 文件，使用者可以参考以下的说明：

1. 打开软件 UartTFT_Tool.exe：

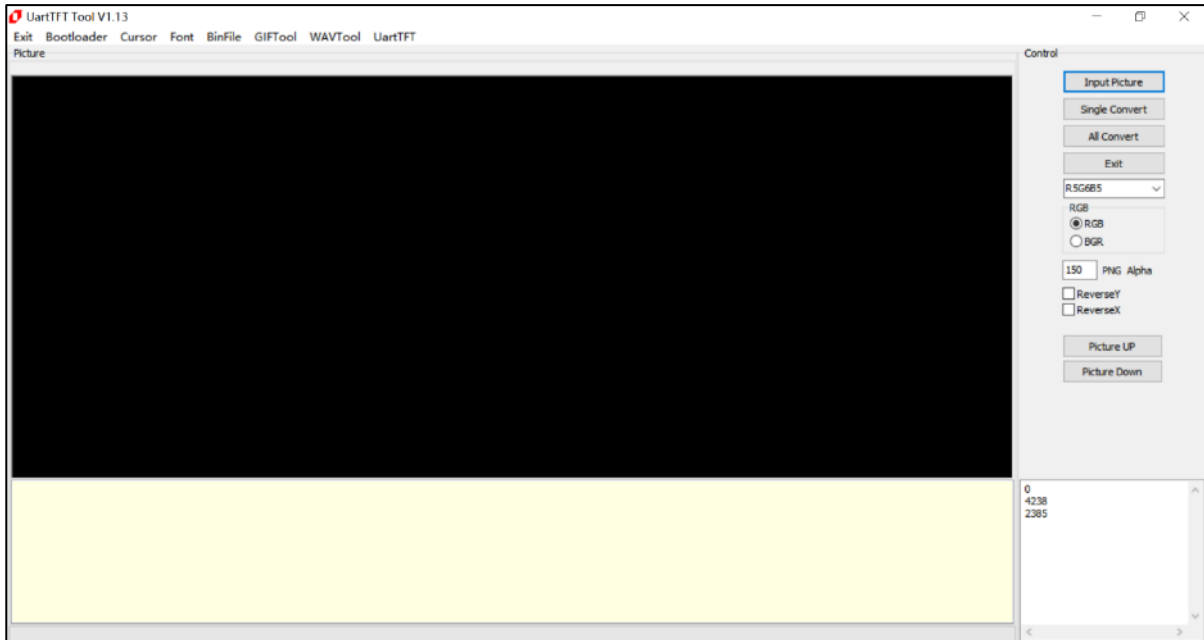


图 4-37：执行 UartTFT_Tool.exe

2. 导入图片，点击 **Input Picture** 按钮，选择需要的图片，点击打开，即可添加此文件夹下的所有图片：



图 4-38：导入图片



图 4-39：导入完成

3. 图片输出设定, 可选 16bpp 或 24bpp 或 Black_White 格式, 以及 RGB 或 BGR 格式。

注意: 若需要制作 Black_White 格式的图片 bin 文件, 源图片文件必须为只有黑色和白色的图片 (比如: 数字图片)。



图 4-40: 设定输出格式

4. 导出某一张图片或导出全部图片, 注意输入文件名时文件名中不能包含下面这些字符, 如: ? * / \ < > : " |, 否则无法保存。

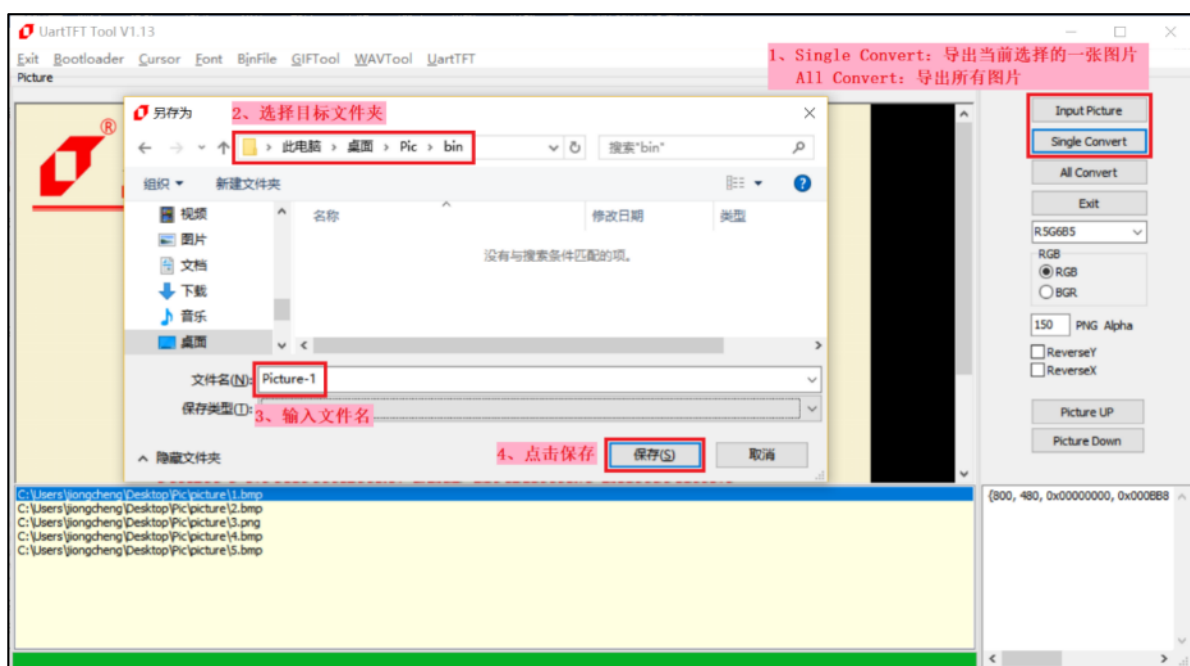


图 4-41: 导出图片 (1/2)



图 4-42：导出图片 (2/2)

5. 成功导出图片 Bin 文件：



图 4-43：导出成功

6. 导出图片后可以在目标文件夹中看到导出的 [Picture1.bin](#) 文件：

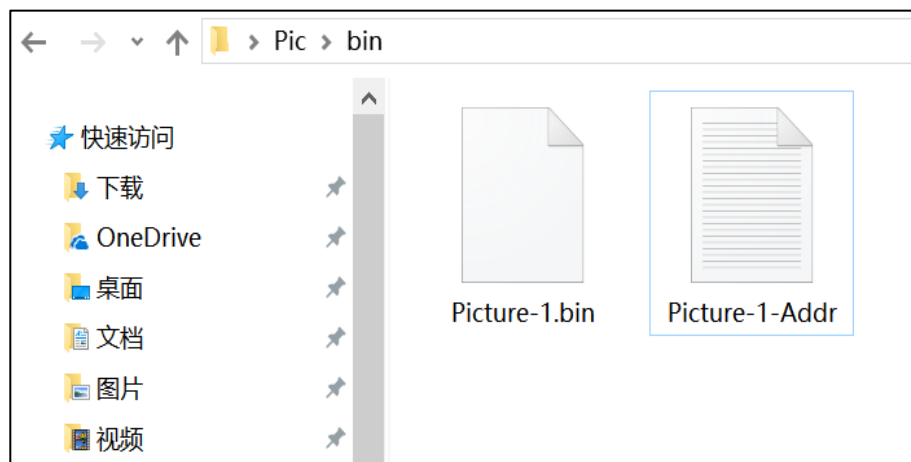


图 4-44：导出的图片 Bin 文件

4.4.3 制作 GIF 檔的 Bin 文件

应用端如果要播放 GIF 的动画，可以将 GIF 档案切割成许多图片，然后透过 DMA 传输方式将这些图片分时存到 LT268B 内建的显示内存，这样就可以达到显示动画的效果。使用者可以用 UartTools 来导入 GIF 的动画档案，然后生成一连串图片的 Bin 文件，详细步骤可以参考以下的说明：

1. 点击【UartTools 菜单>GIFTool】即可打开 GIF Bin 文件制作界面：



图 4-45：打开 GIF Bin 文件制作界面

2. 导入 GIF 图片，点击 Input Picture 按钮，选择需要的 GIF 图片，点击打开，即可添加此文件夹下的所有 GIF 图片：

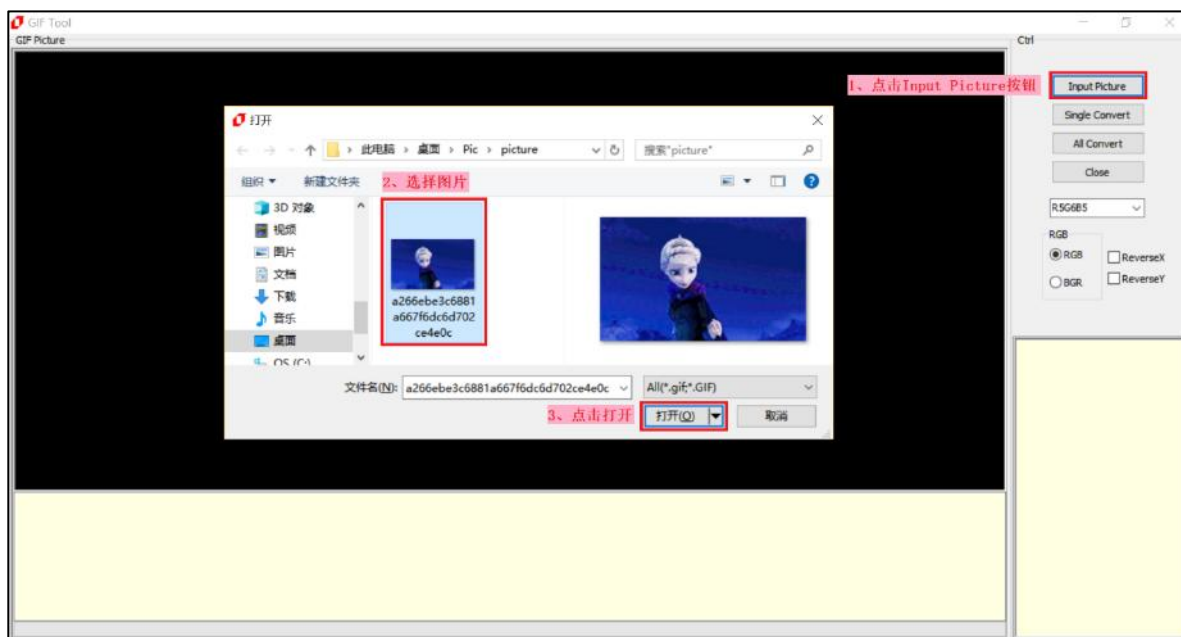


图 4-46：导入图片



图 4-47：导入完成

3. GIF 图片输出设定, 可选 16bpp 或 24bpp 格式, 以及 RGB 或 BGR 格式:



图 4-48: 设定输出格式

4. 导出某一张 GIF 图片或导出全部 GIF 图片, 注意输入文件名时文件名中不能包含下面这些字符, 如: ? * / \ < > : " | , 否则无法保存。

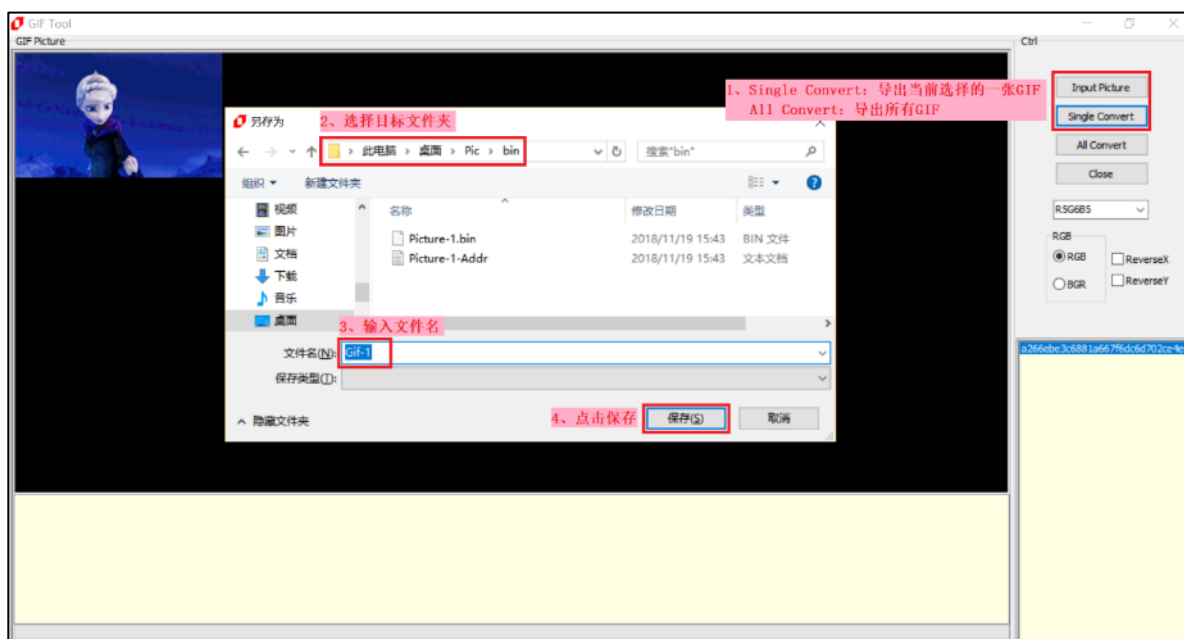


图 4-49: 导出图片 (1/2)

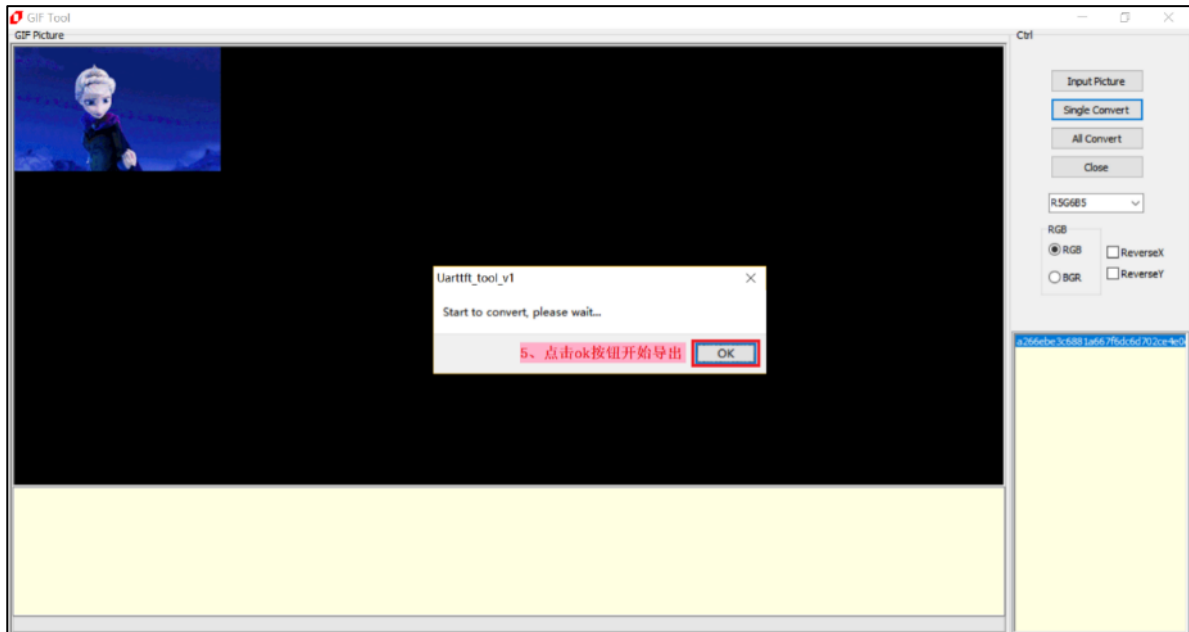


图 4-50：导出图片 (2/2)

5. 成功导出图片 Bin 文件：

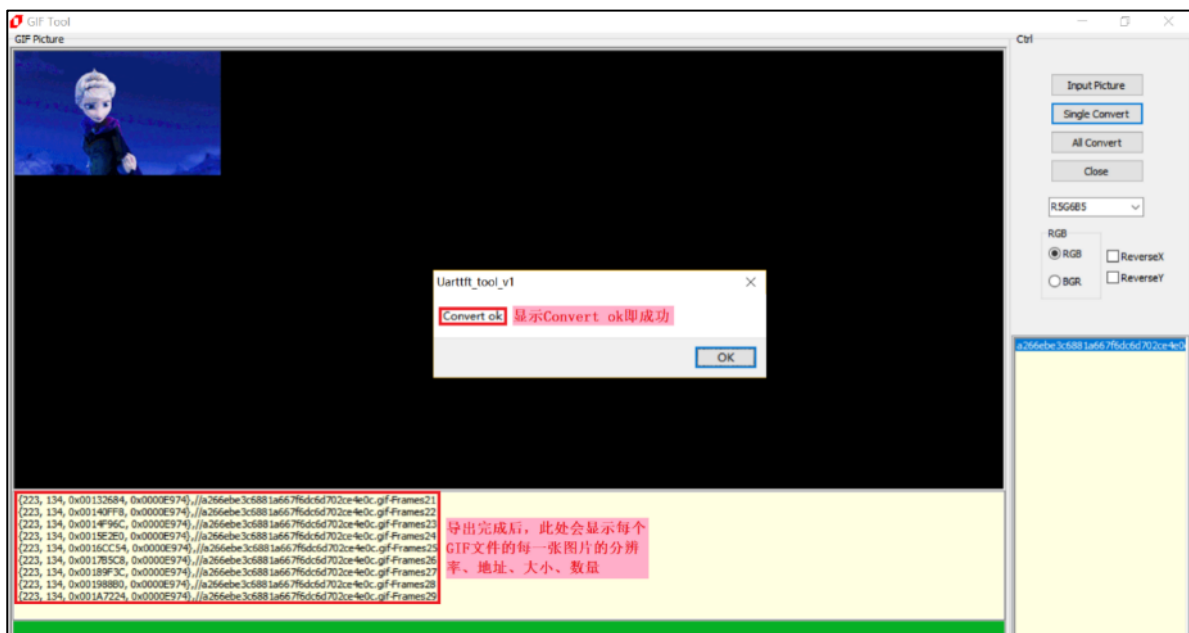


图 4-51：导出成功

6. 导出图片后可以在目标文件夹中看到导出的 load.bin 文件以及 txt 详情文档：

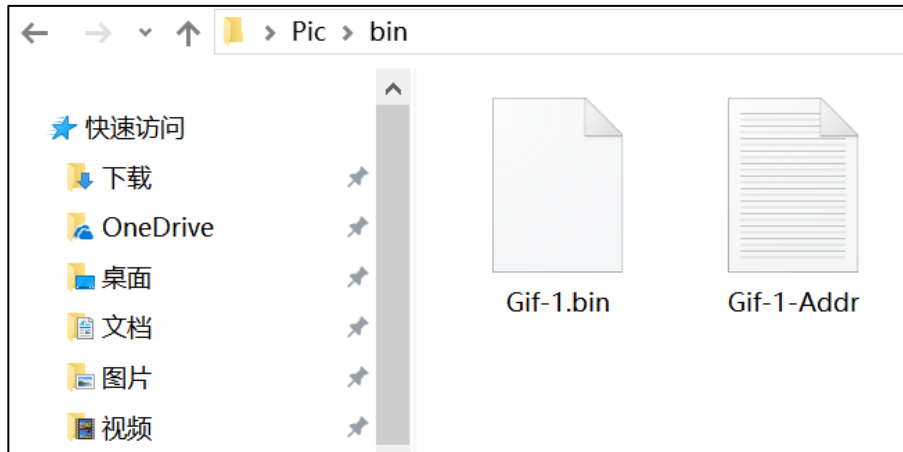


图 4-52：导出的图片 Bin 文件

在 txt 文档内能看到 Gif 内每张图片的分辨率、地址、大小以及图片数量：

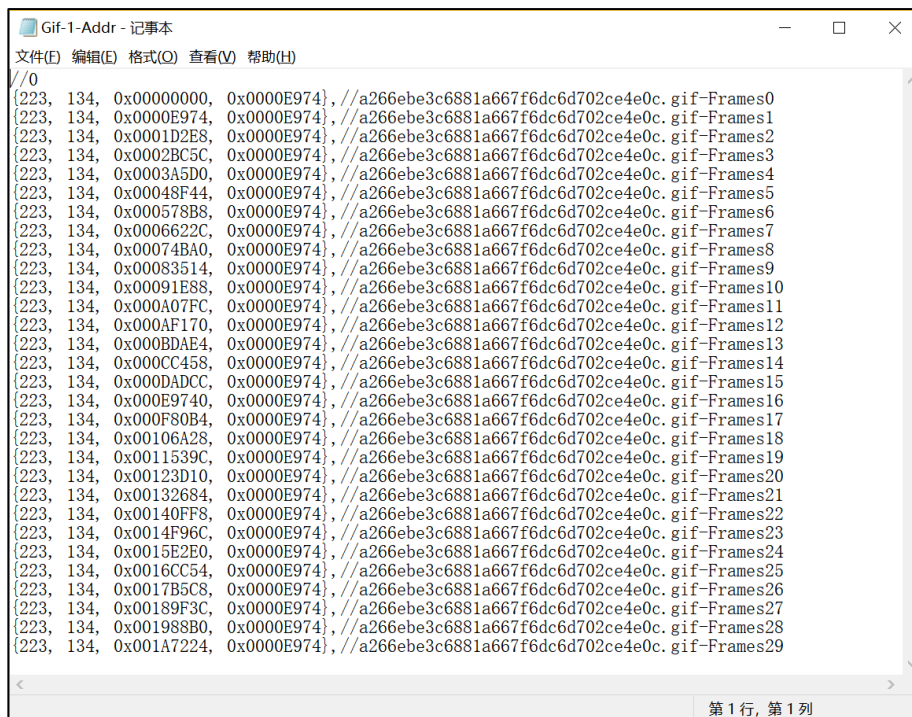


图 4-53：导出的每张图片信息

4.4.4 制作字库的 Bin 文件

4.4.4.1 全字库制作

在应用端如果要使用到中文字库，UartTools 有一项制作字库 Bin 文件的功能，能将字库信息转成 Bin 文件，然后透过 DMA 传输方式将字库数据存到 LT268B 的内建显示内存中，之后如果要在 TFT 屏上显示中文，MCU 只须送 GB 码（2 个 Bytes）就可以在设定的位置上显示出中文，因此可以提升中文显示效能，还降低 MCU 处理中文显示的负担。对于字库 Bin 文件的制作，使用者可以参考以下的说明，举例产生一个 16*16 的宋体字库 Bin 文件：

1、点击【UartTools 菜单>Font】即可打开中文字库 Bin 文件制作界面：

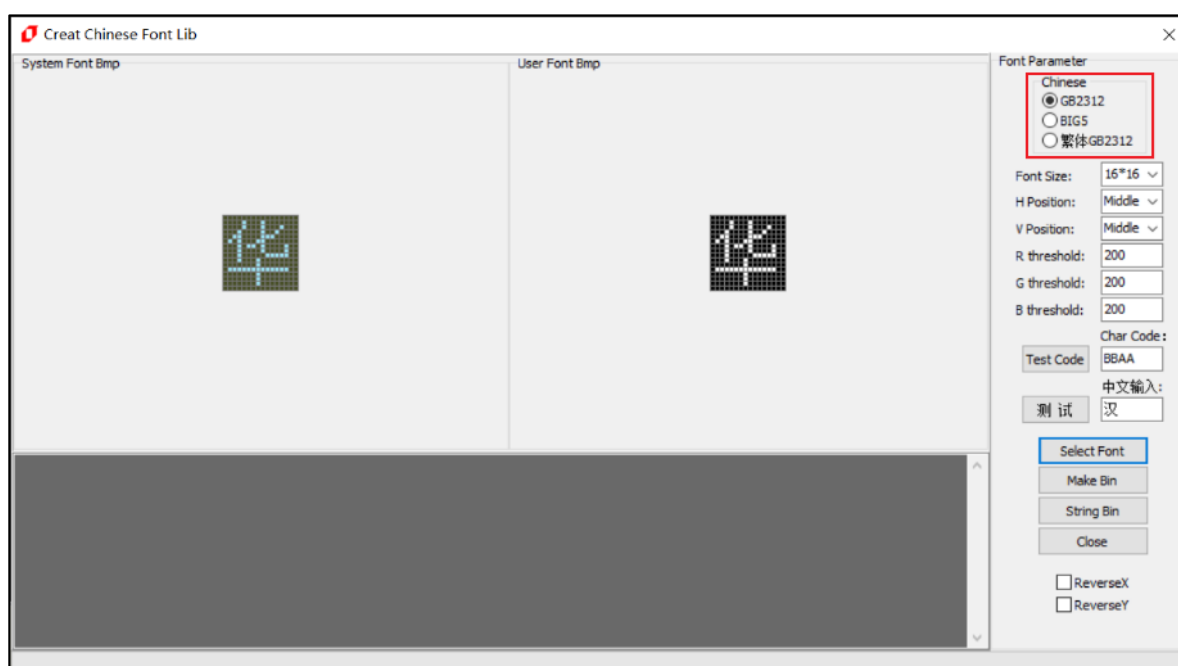


图 4-54：制作中文字库

2、点击【Select Font】按钮，可设置字体、字形、大小等，设置完毕后，按确定保存：

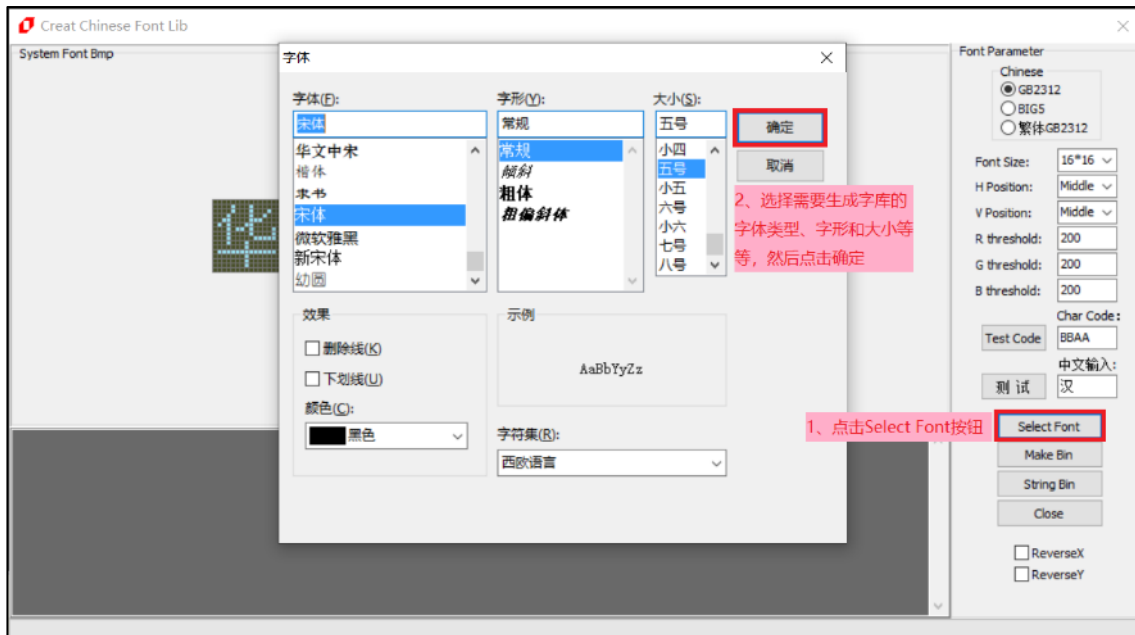


图 4-55：选择字体

3、设置字库有 16*16、24*24、32*32、48*48、72*72 五种字体大小，你也可以设置字体横向（偏左、居中、偏右）和纵向（偏上、居中、偏下）的位置、颜色阈值（0~254）、和预览文字，点击【Test Code】按钮即可查看该字符的数据。



图 4-56：设置字库

- 4、点击【Make Bin】即可输出字库 Bin 文件。注意输入文件名时文件名中不能包含下面这些字符，如：? * / \ < > : " |，否则无法保存。

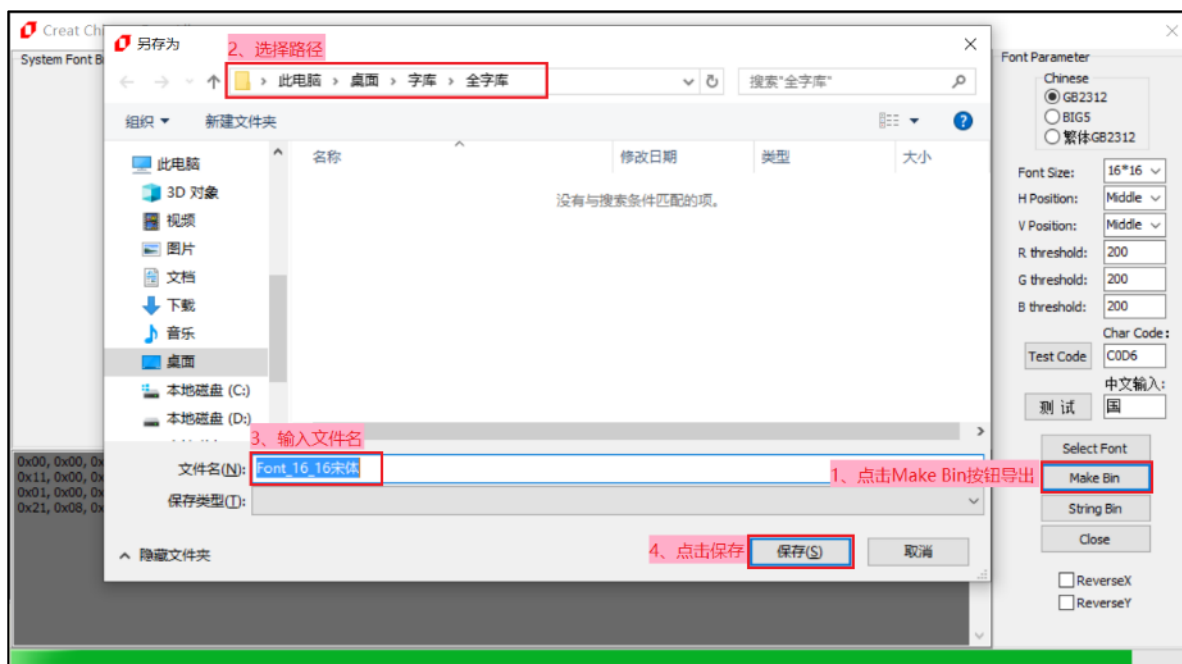


图 4-57：保存字库

当显示（字库）+Font Lib ok 时，即全字库制作成功：

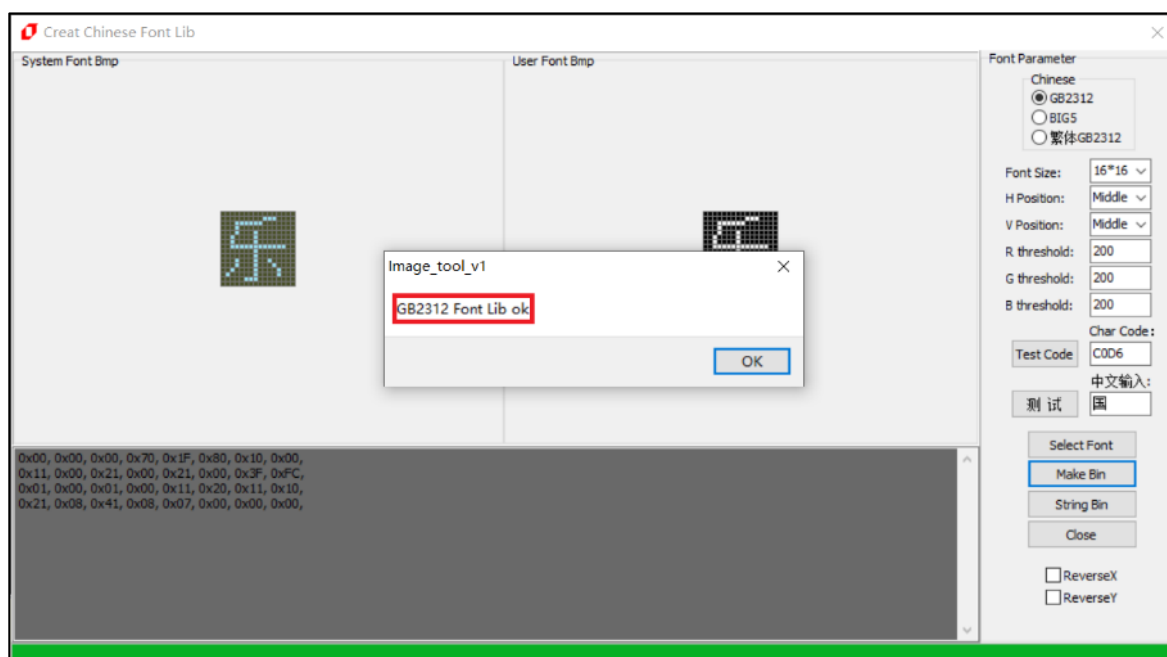


图 4-58：字库制作完成

5、制作完成后可以在目标文件夹中看到导出的 简体 16_16 宋体.bin 文件：

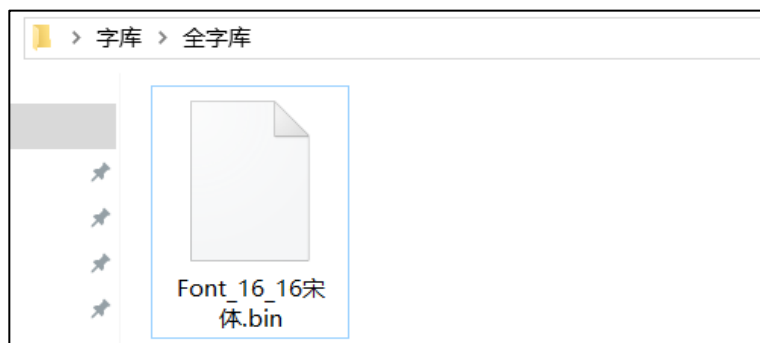


图 4-59：导出的字库 Bin 文件

4.4.4.2 自定义字库制作

由于大字库容量相对比较大，考虑到会占用 Flash 比较多的空间，因此该主控端提供了自定义字库的功能，客户可根据需要提前想好要调用的文字，从而仅仅将这部分文字打包成一个字库，进而减少 Flash 容量的浪费。对于自定义字库 Bin 文件的制作，使用者可以参考以下的说明，举例产生一个 48*48 的宋体自定义字库 Bin 文件：

1. 前期选择字体大小和形状，步骤与 4.2.5.1 全字库制作相同，点击【String Bin】后选择一个存放要调用文字的 txt 文档：

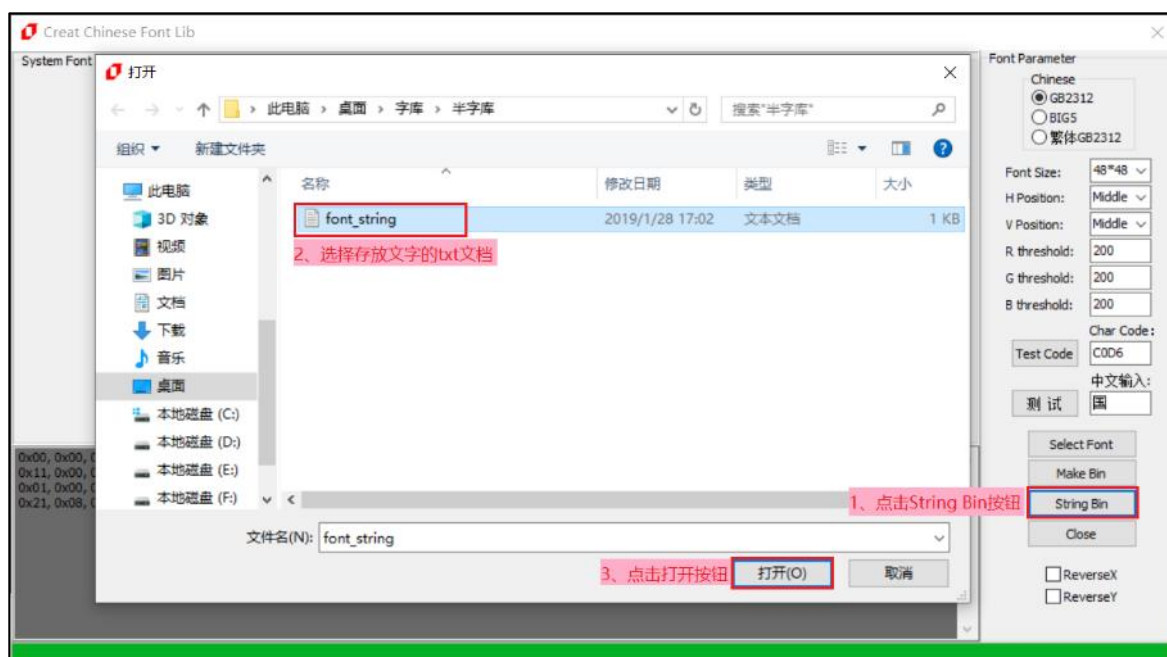


图 4-60：选择 txt 文档

2. 提前将需要调用显示的文字存放在一个记事本下：

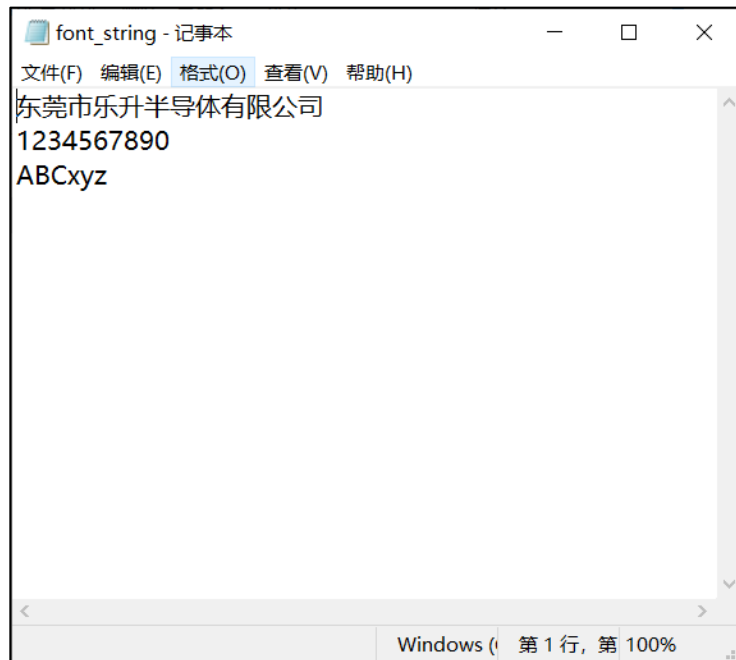


图 4-61：存放文字的 txt 文档

3. 指定一个路径存放将要生成的自定义字库，命名好后点击保存：

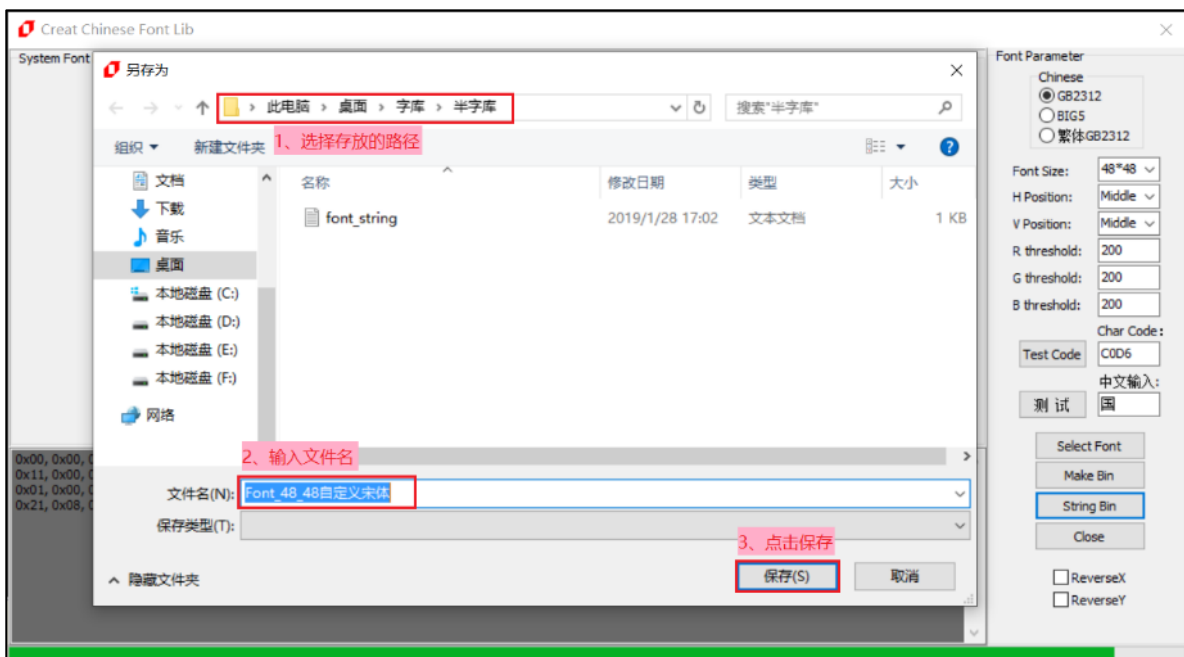


图 4-62：保存字库

当显示 String Font Lib ok 时，即自定义字库制作成功：

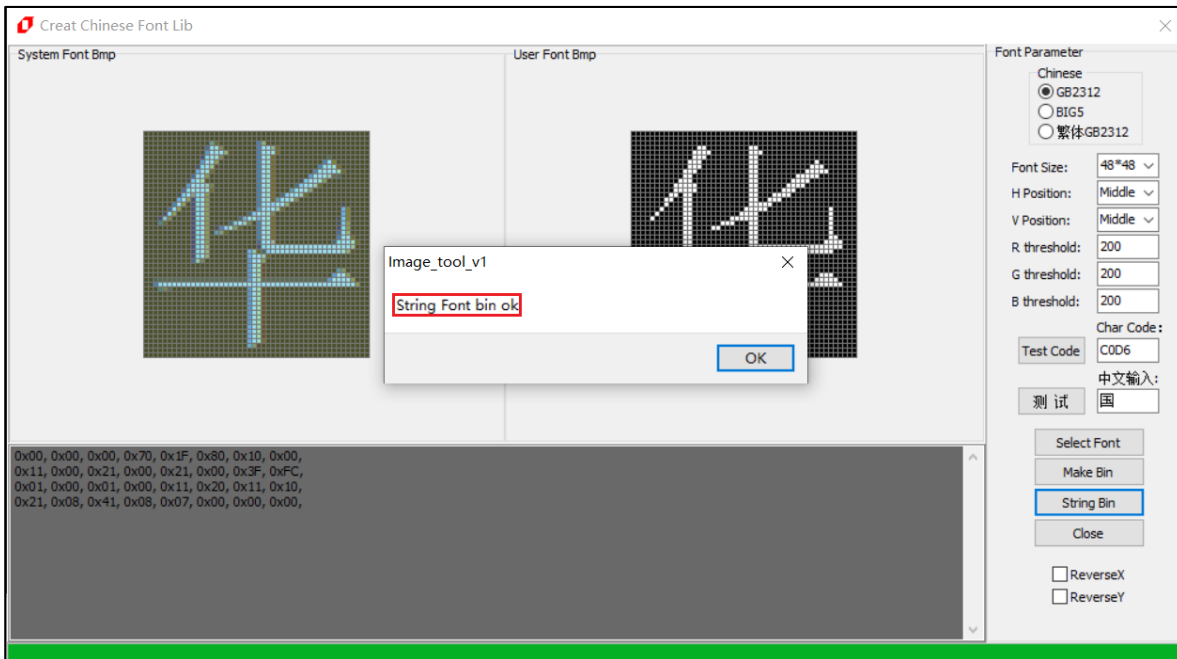


图 4-63：字库制作完成

- 完成后可以在目标文件夹中看到导出的 `Font_48_48 自定义宋体.bin` 文件：

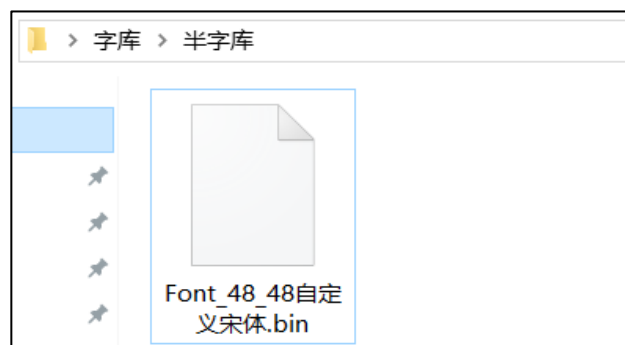


图 4-64：导出的字库 Bin 文件

4.4.5 制作 Wav 檔的 Bin 文件

4.4.5.1 音频文件转 WAV

1. 若音频素材格式不是 WAV 格式，则需要通过格式转换来获取 WAV 格式文件。下面以“格式工厂”免费版为转换平台进行操作。首先打开软件，选择音频，选择“-> WAV”，进入添加文件界面。

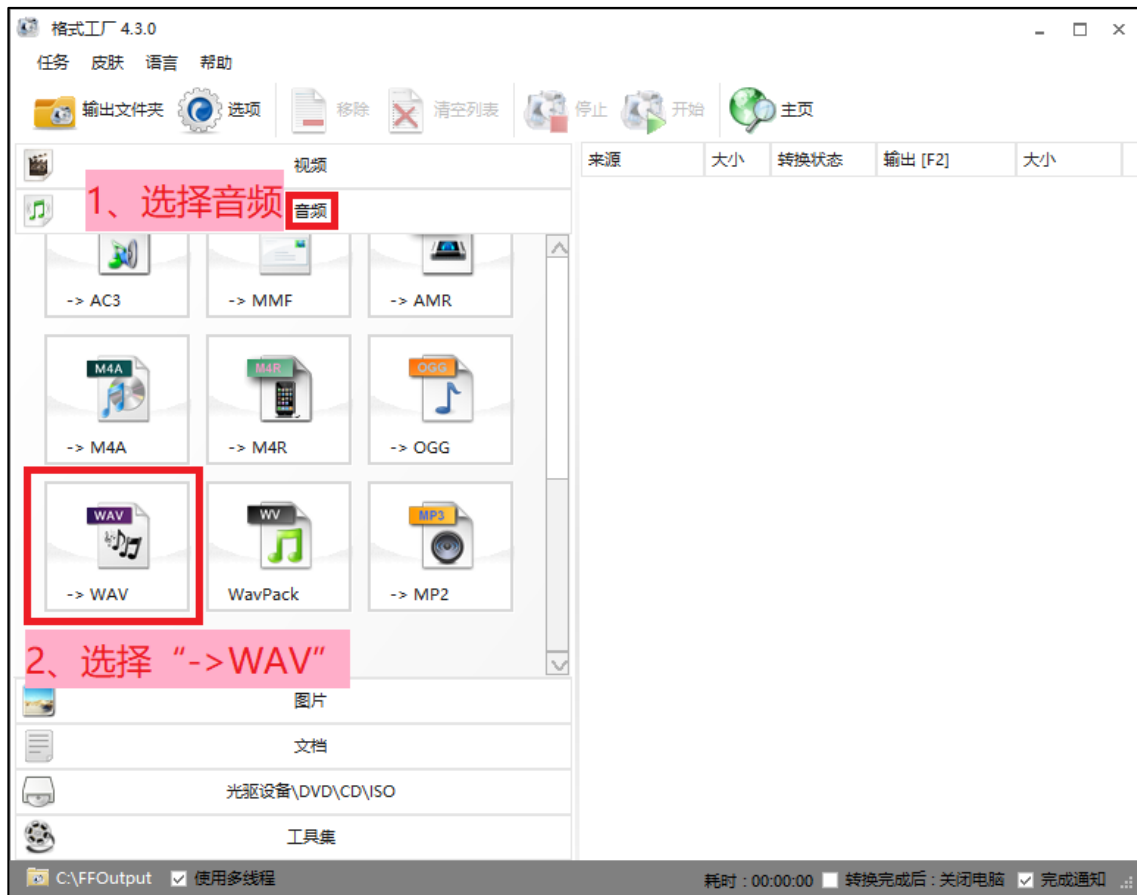


图 4-65：选择功能



图 4-66: wav 功能界面

2. 点击添加文件按钮，选择需要转换的音频文件。

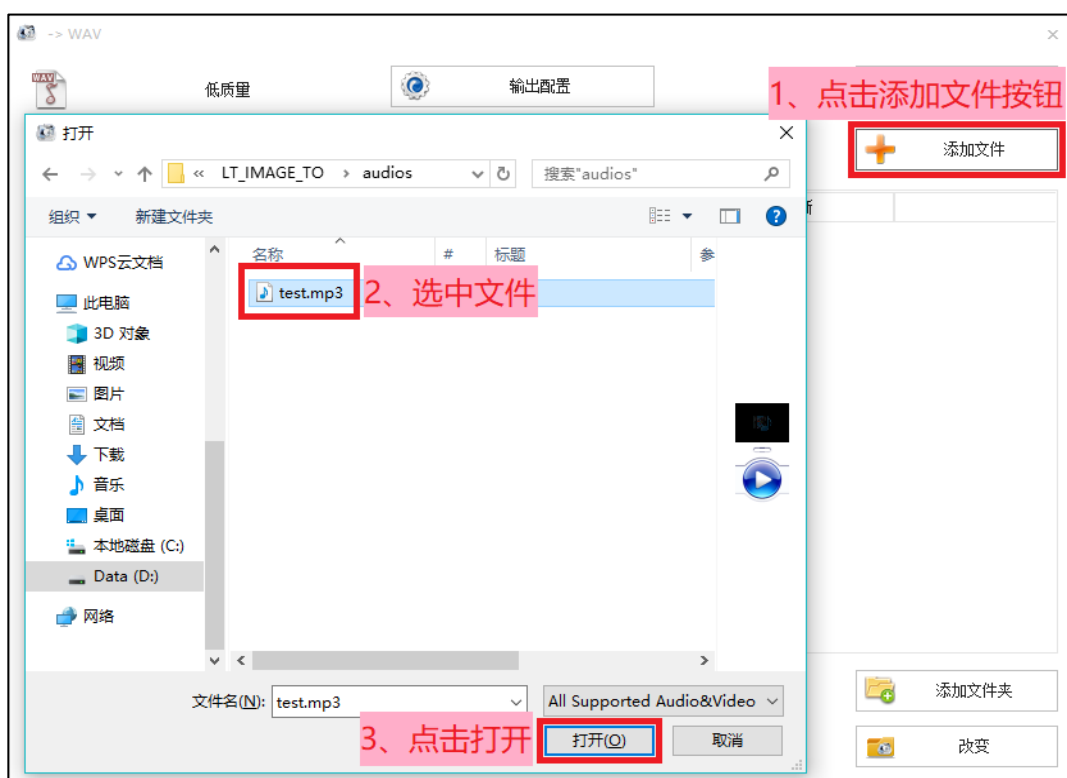


图 4-67: 添加 wav 文件

3. 点击输出配置按钮，进行音频设置，其中，采样率可选 11025 或 22050。由于采用低成本功放方案，同时为了节省 bin 文件存放空间，推荐选择采样率为 11025。



图 4-68：输出配置

4. 选中文件，点击剪辑按钮，进入音频剪辑界面，可选择需要的音频段。如不需要，请直接转到步骤 5。



图 4-69：进入剪辑功能

剪辑音频，可调节音量大小，以及截取其中的片段，剪辑完成后，点击确定按钮保存：

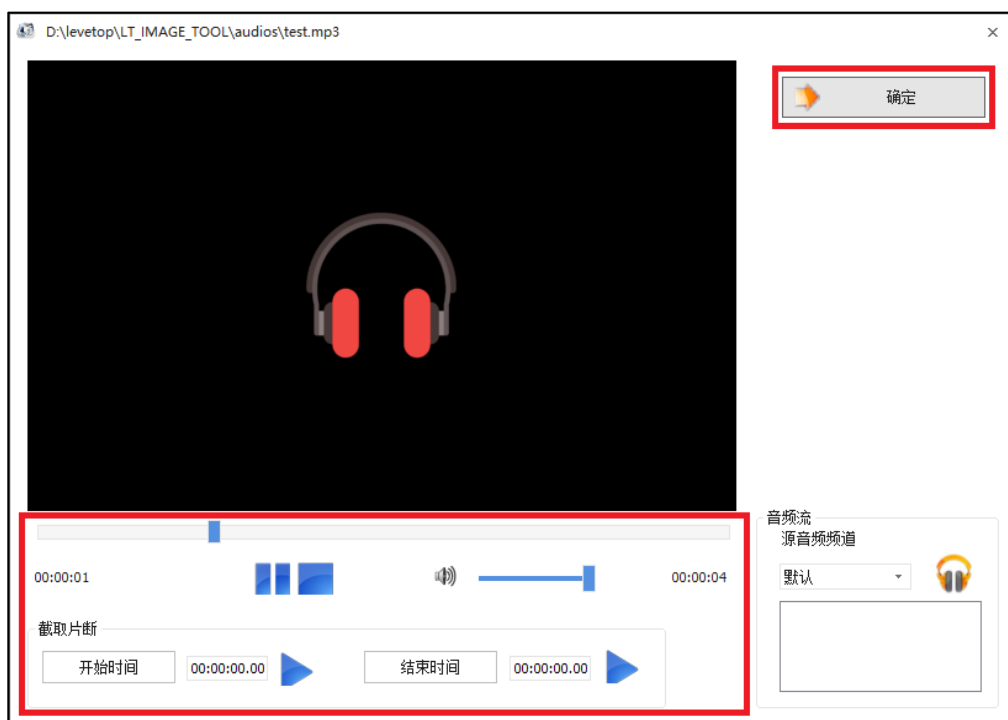


图 4-70：剪辑界面

5. 点击右下角的改变按钮，可重新选择输出目标文件夹，点击确定按钮添加任务。

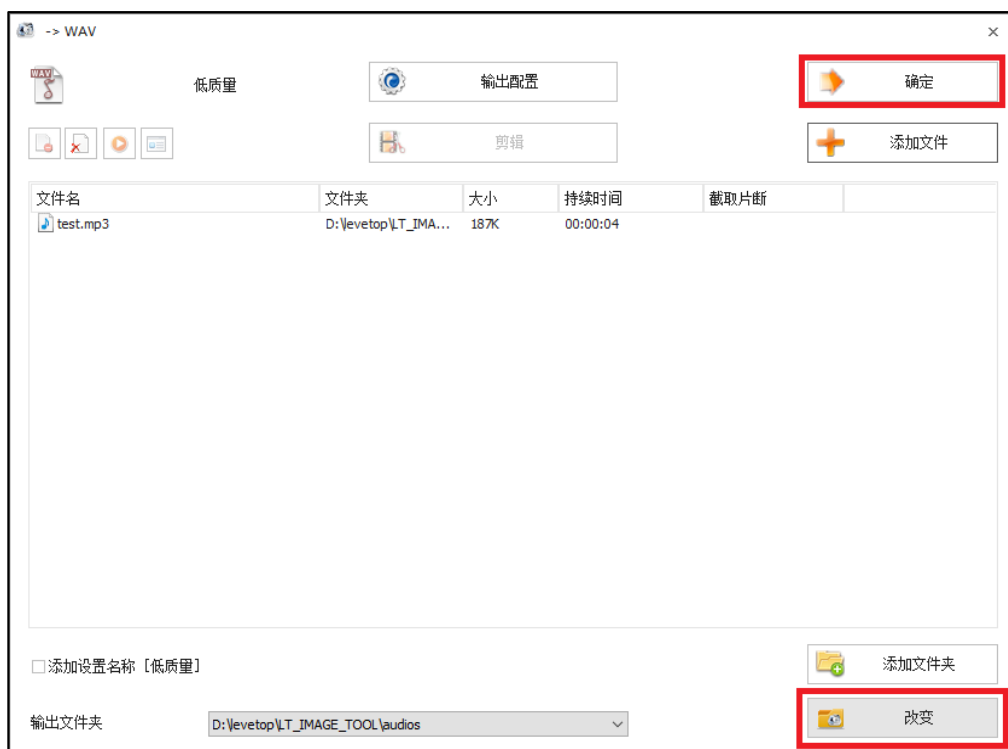


图 4-71：选择输出文件夹

6. 点击开始按钮，开始转换，转换完毕后，可在目标文件夹查看导出的 WAV 文件。

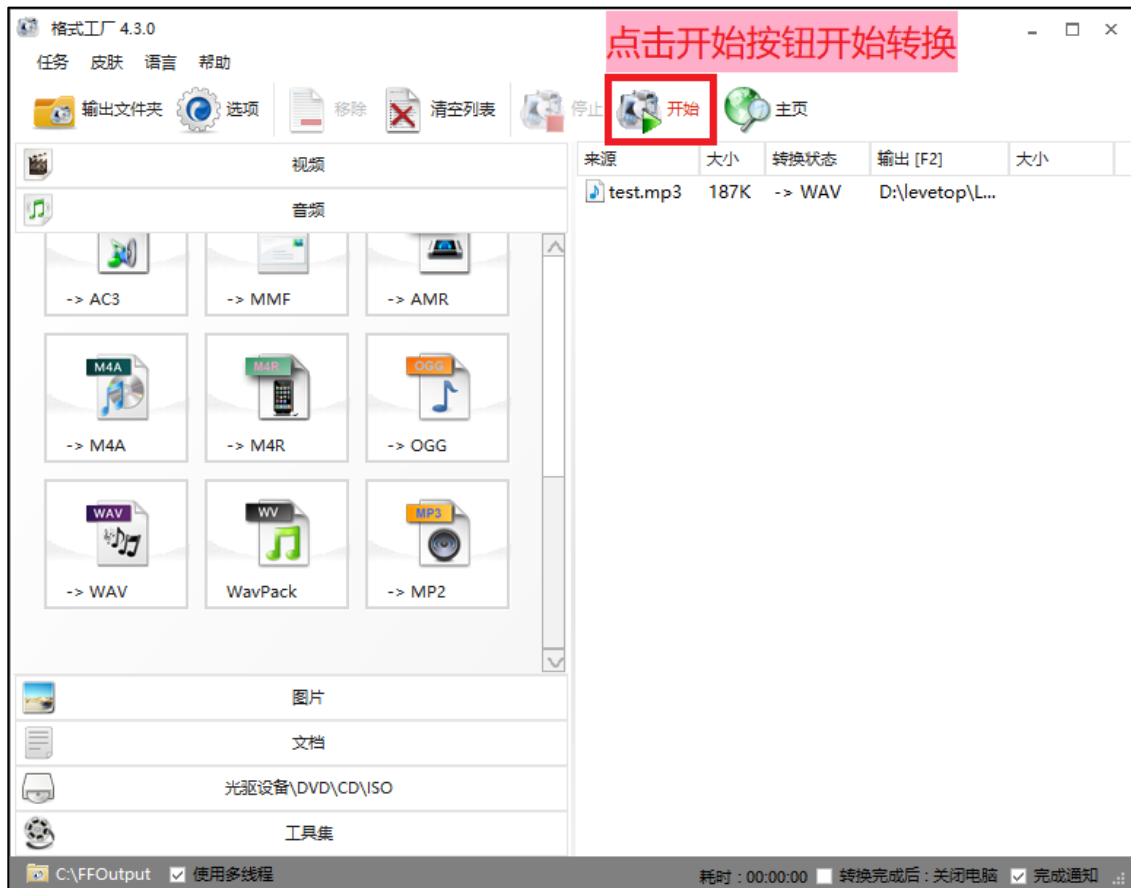


图 4-72：开始转换



图 4-73：导出的 wav 文件

4.4.5.2 制作 WAV Bin 文件

1. 点击【UartTFT_Tool 菜单>WAVTool】即可打开 WAV Bin 文件制作界面：

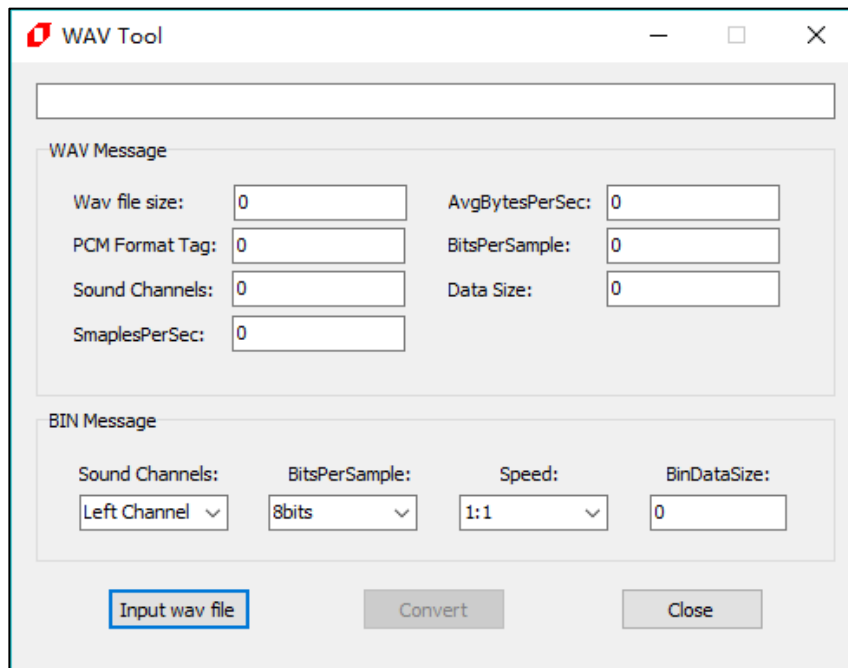


图 4-74：WAV Bin 文件制作界面

2. 导入 WAV 文件，点击 Input wav file 按钮，选择需要转换的 WAV 文件，点击打开，即可添加：

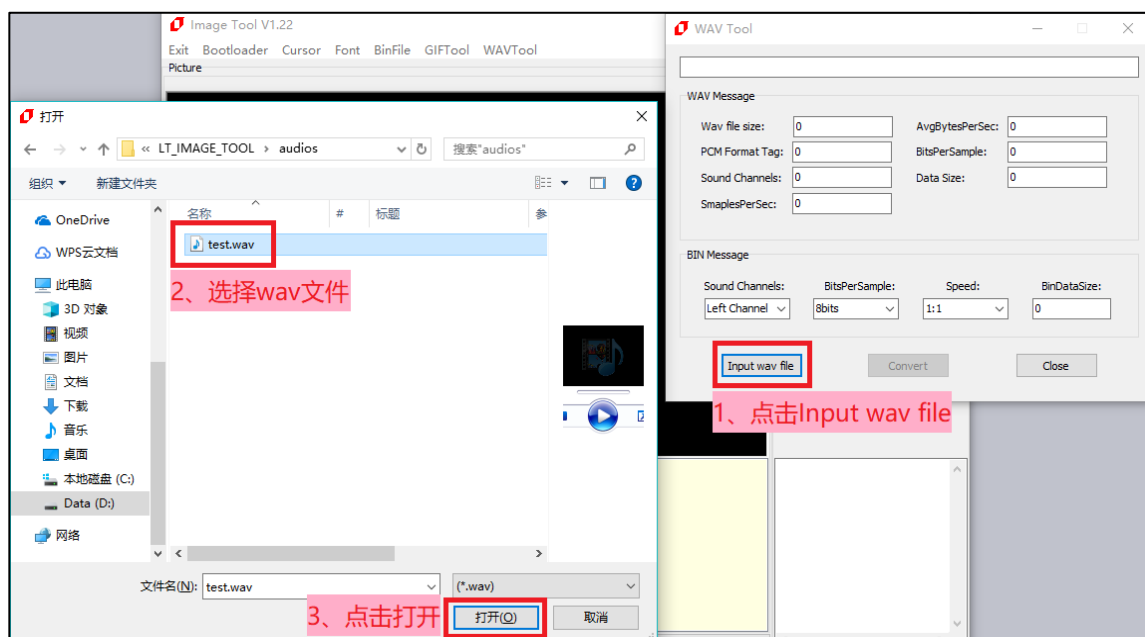


图 4-75：添加 wav 文件

添加成功后, 在界面上方, 可以看到文件所在路径, 并显示 wav 文件的相关数据 (如下图)。如果采样率 (SamplesPerSec) 不是 11025 或 22050, 建议改变 Speed 的比率, 或是先用其他音频软件将采样率变成 11025 或 22050。

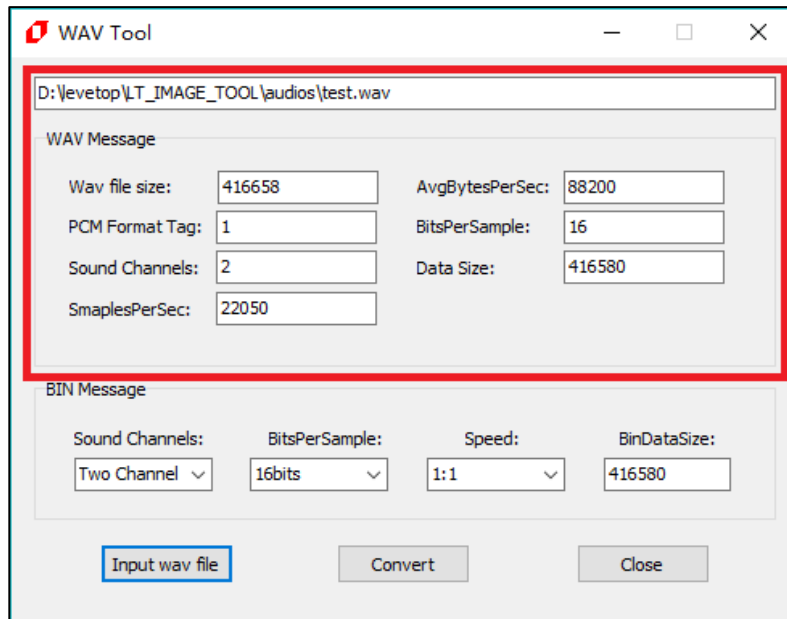


图 4-76: 添加成功

3. 设置 bin 文件的参数, Sound Channels 选项可选左声道、右声道或双立体声道, BitsPerSample 选项可选 8bits 或 16bits。一般选用单声道、8bits 就足够, bin 文件较小, 便于储存。Speed 选项选择生成 bin 文件时的采样速度, 速度越高, 音质也会相对下降, 同时需要改变程序的定时器更新数据的时间。

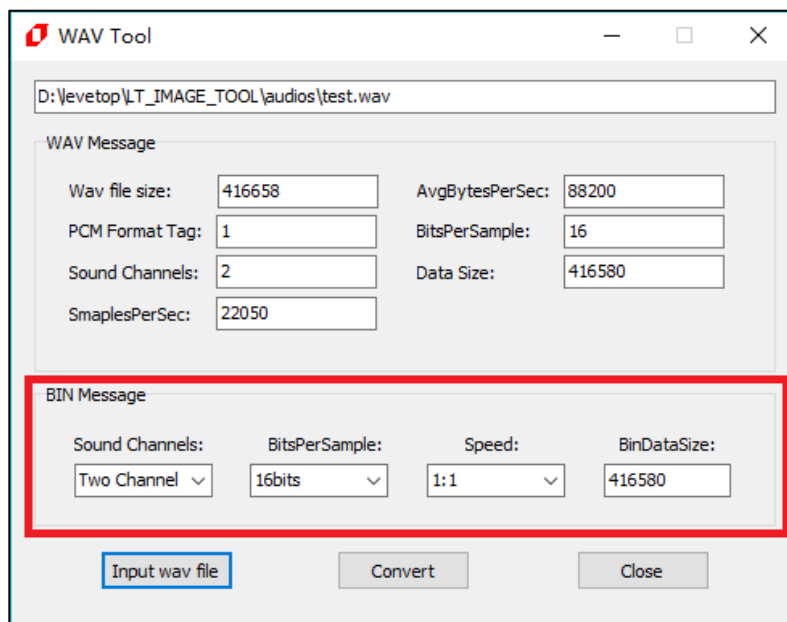


图 4-77: WAV Bin 文件制作界面

4. 导出 bin 文件, **注意**: 输入文件名时文件名中不能包含下面这些字符, 如: ? * / \ < > : " |, 否则无法保存。

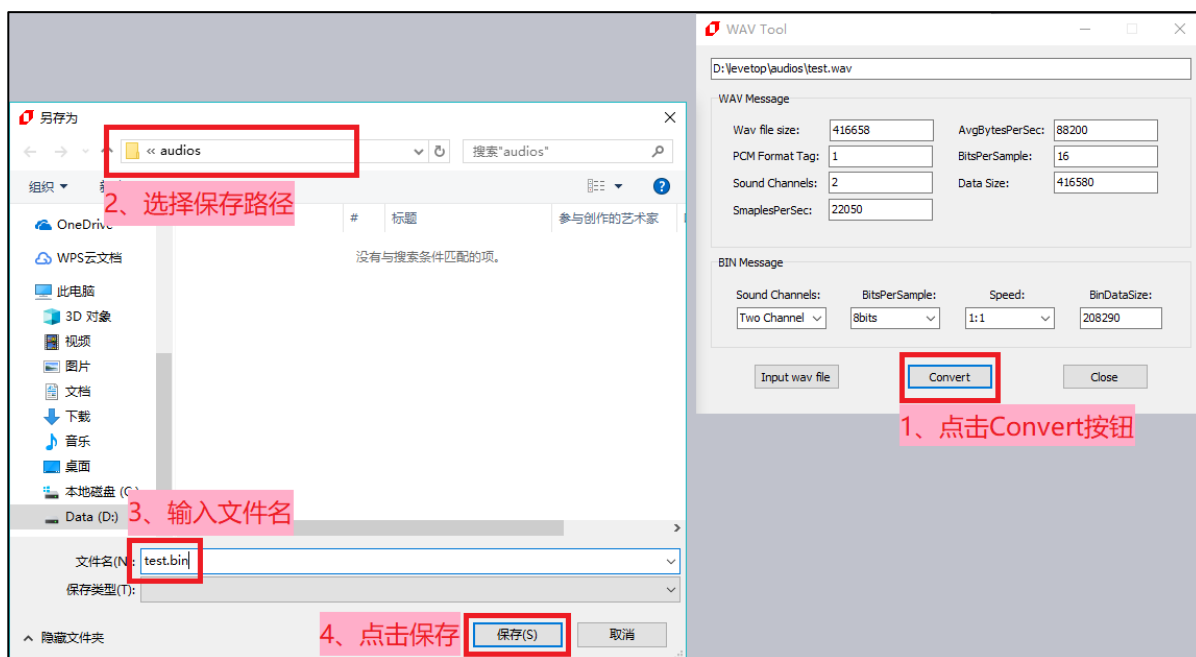


图 4-78: 导出 bin 文件

若该音频文件与 bin 设置参数不符合, 则会提示 Can't support this bits, 请按照步骤 3 重新设置:

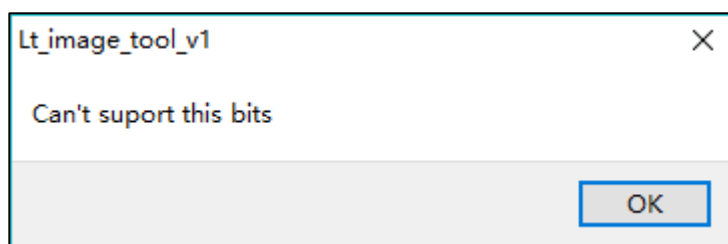


图 4-79: 错误提示

若符合要求, 则点击 OK 按钮开始导出:

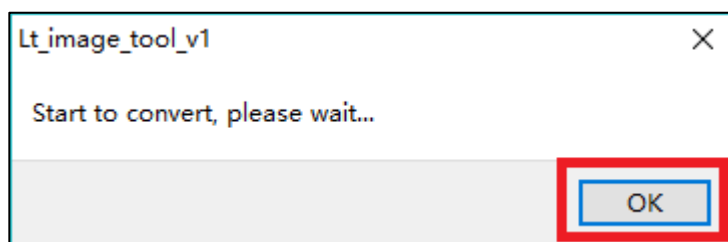


图 4-80: 点击 OK 导出

- 成功导出 WAV bin 文件，并可在目标文件夹看到导出的 test.bin 文件：

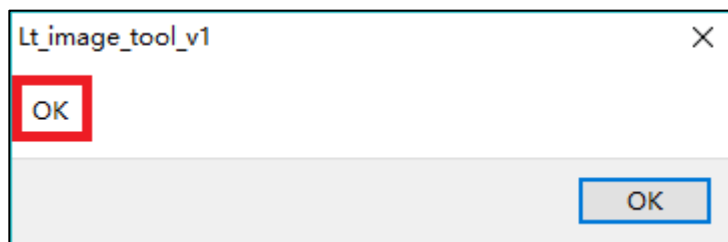


图 4-81：导出成功

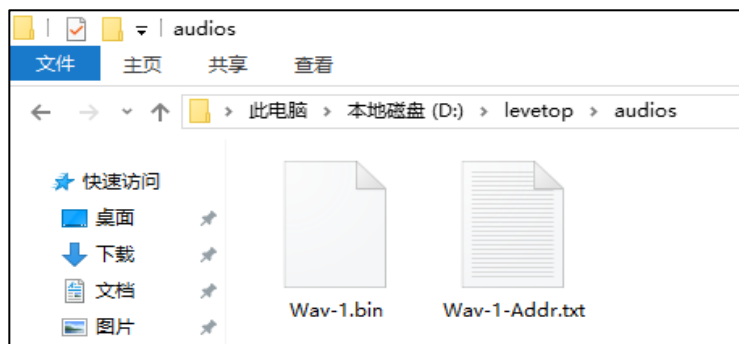


图 4-82：生成的 bin 文件

4.4.5.3 典型 PWM 音频驱动电路

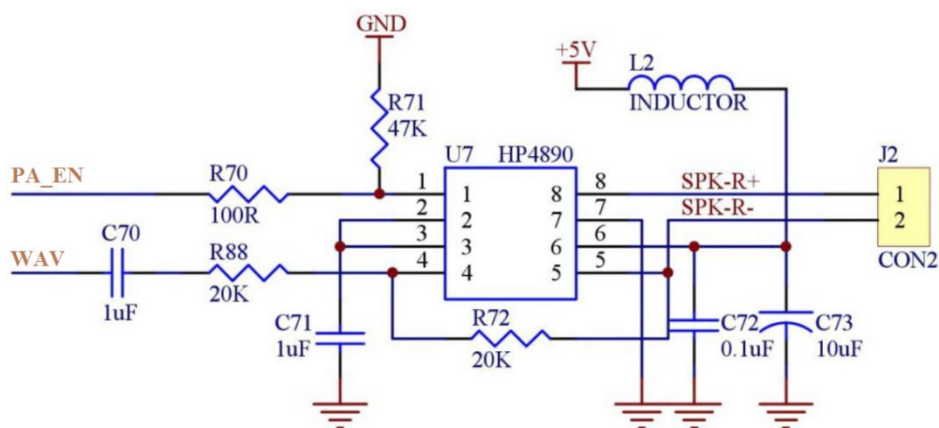


图 4-83：典型 PWM 音频驱动电路

4.4.6 Bin 文件的结合

在产生许多不同的 Bin 文件后，最终要写入到 SPI Flash 内，因此 UartTools 提供一个 Bin 文件整合的功能，可以让使用者在 PC 端将不同的 Bin 文件结合成一个 Bin 文件。详细步骤可以参考以下的说明：

- 1、点击【UartTFT_Tool 菜单>Binfile】文件整合界面，最多能整合 6 个 Bin 文件，点击【File 1~6】可依次添加。注意，Bootloader.bin 文件需放在地址 0 位置，即 File 1。

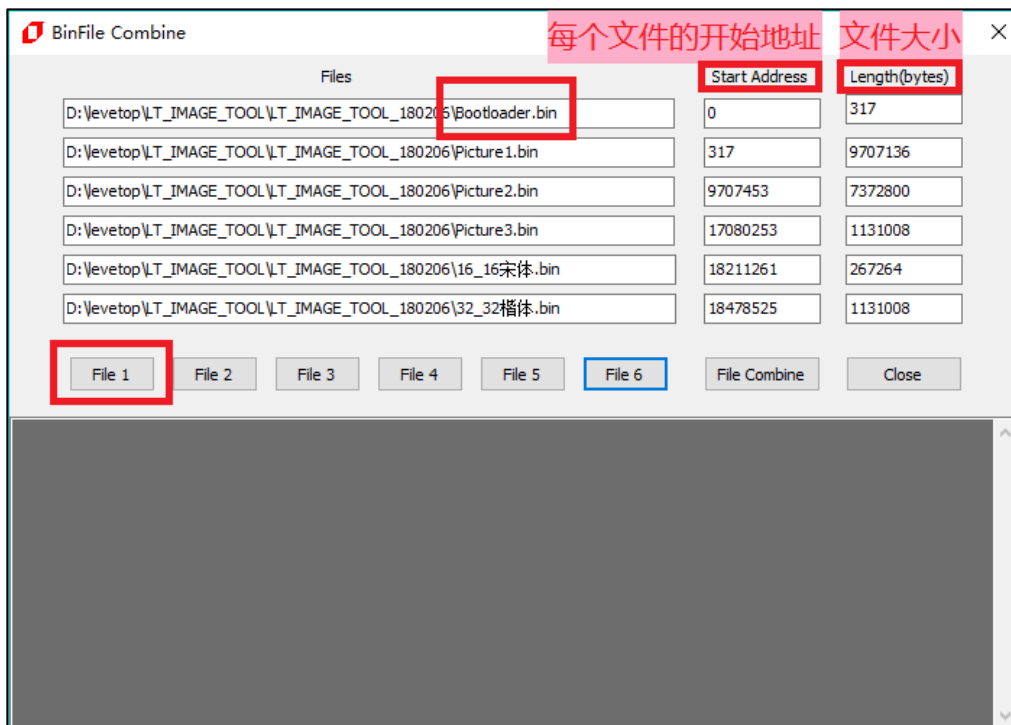


图 4-84: Bin 文件整合

- 2、点击【File Combine】按钮保存整合文件，注意输入文件名时文件名中不能包含下面这些字符，如：? * / \ < > : " |，否则无法保存。

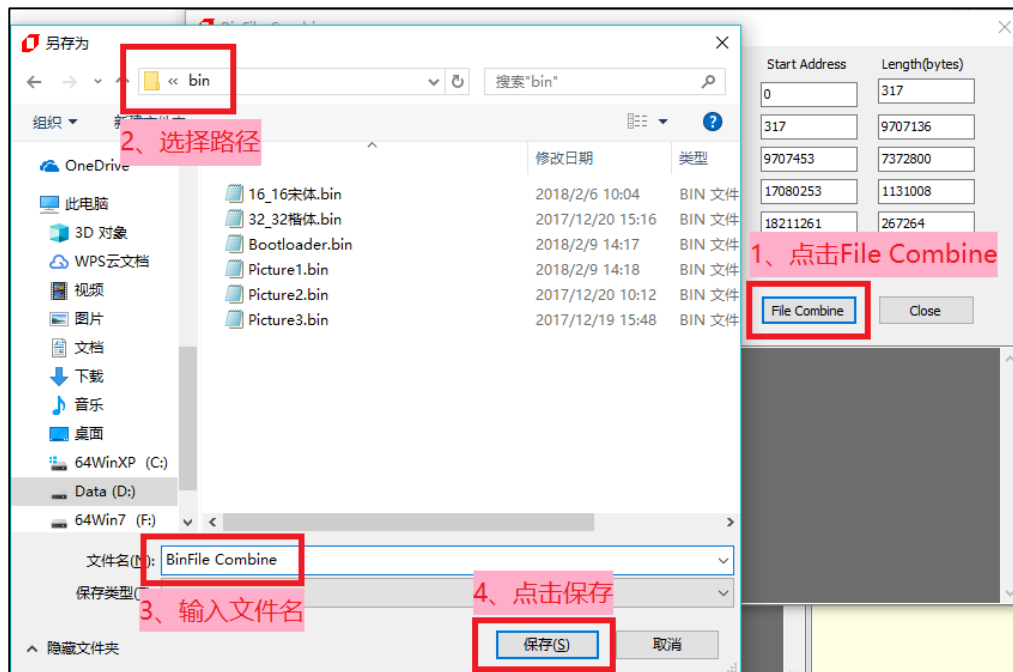


图 4-85：保存整合文件

当显示 Combine over 时，即整合成功，并显示每个源文件的地址和大小，同时生成一个 BinFile Combine-Addr.txt 文件，便于查阅每个源文件的地址、大小等详细信息。

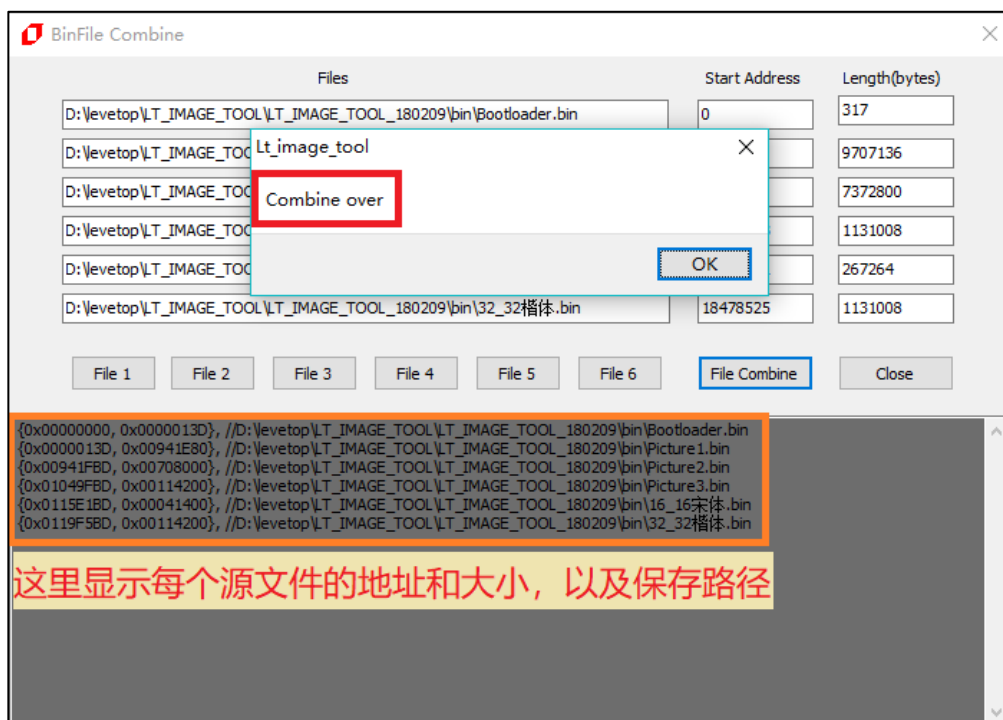


图 4-86：整合成功

3、生成的 BinFile Combine-Addr.txt 文件：

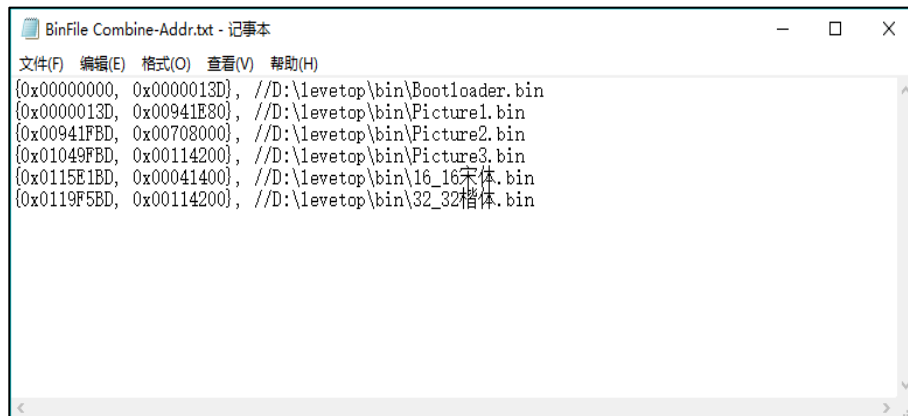


图 4-87：保存文件信息

整合完成后可以在目标文件夹中看到导出的 **BinFile Combine.bin** 文件，然后使用者可以用电脑 USB 接口及 **LT268B_ISP_Vxx.exe**，或是专用的 SPI Flash 烧录器将此档案烧录到连接至 LT268B 的 SPI Flash。

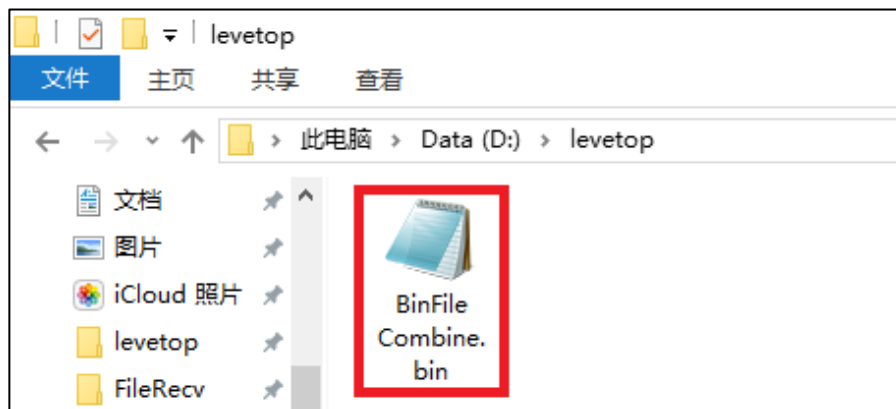


图 4-88：导出的 Bin 整合文件

5. 串口通讯软件(UartDebug.exe)

UartDebug.exe 是 乐升半导体 提供的一个专用程序，主要用 PC 端的 USB 接口转成 Uart 通讯串口信号，来发送指令给 LT268B 串口屏的 Uart 接口，进而让 TFT 屏显示不同的内容。由 UartTFT_Tool 点击【Uart Debug】就可以连接到 UartDebug.exe 及进入此通讯软件，其使用方式由以下几部分组成：

- (一) 打开 UartDebug 软件注意事项
- (二) UartDebug 软件界面的简介
- (三) 打开 COM 口
- (四) 加载与更新 UserInfo.bin 文件
- (五) 指令的加载
- (六) 指令的发送
- (七) 指令列表的更改与保存
- (八) 清除指令框指令与接收框数据
- (九) UserInfo.bin 的生成

5.1 打开 UartDebug 软件注意事项

使用 UartDebug 软件时，最好以管理员身份运行，不然 UartDebug 软件发送接收数据可能会出错。以管理员身份运行有两种操作如下：

- 1) 直接在 UartDebug 图标上单击右键，然后在跳出的框中找到以管理员身份运行这一栏单击即可，选择该方式的话每次使用该软件都要进行上述操作，具体操作如下图：



图 5-1：设置以管理员身份运行

- 2) 直接在 UartDebug 图标上单击右键，然后在跳出的框中找到属性这一栏单击，如下图 5-2，单击属性后会跳出一个新窗口，在新窗口上单击兼容性，在特权等级下面以管理员身份运行此程序前面的框中打钩，最后单击确认，如图 5-3。配置好之后以后使用该软件只要双击打开即可。



图 5-2：设置程序属性

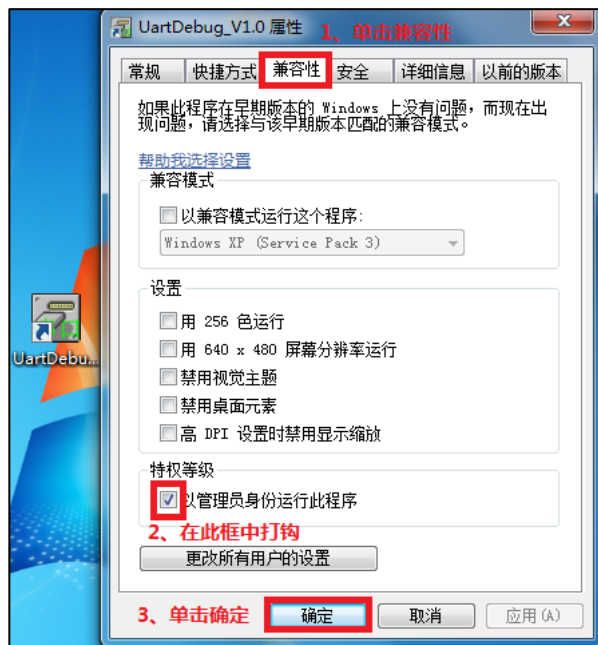


图 5-3：勾选以管理员身份运行

- 3) 直接执行 UartDebug.exe 打开程序，或是通过 UartTFT_Tool.exe 软件打开。首先双击打开 UartTFT_Tool 软件，然后然后单击【UartTFT】会跳出一个 Uart TFT Tool 窗口，再单击【Uart Debug】即可打开 UartDebug 软件，如前一章第 4.4.1 节的步骤 9。具体如下图：

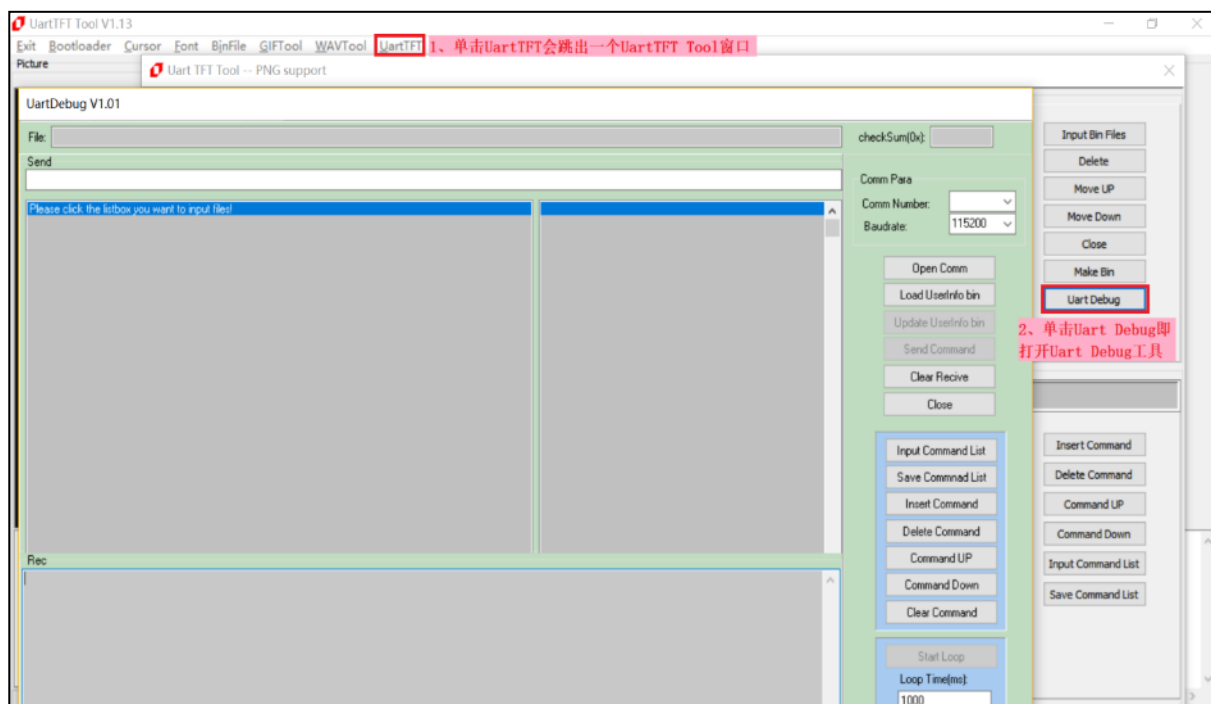


图 5-4：在 UartTFT_Tool 软件上打开 UartDebug 软件

5.2 UartDebug 的功能简介

进入 UartDebug 软件后的界面及 UartDebug 软件界面的简介如下图所示：

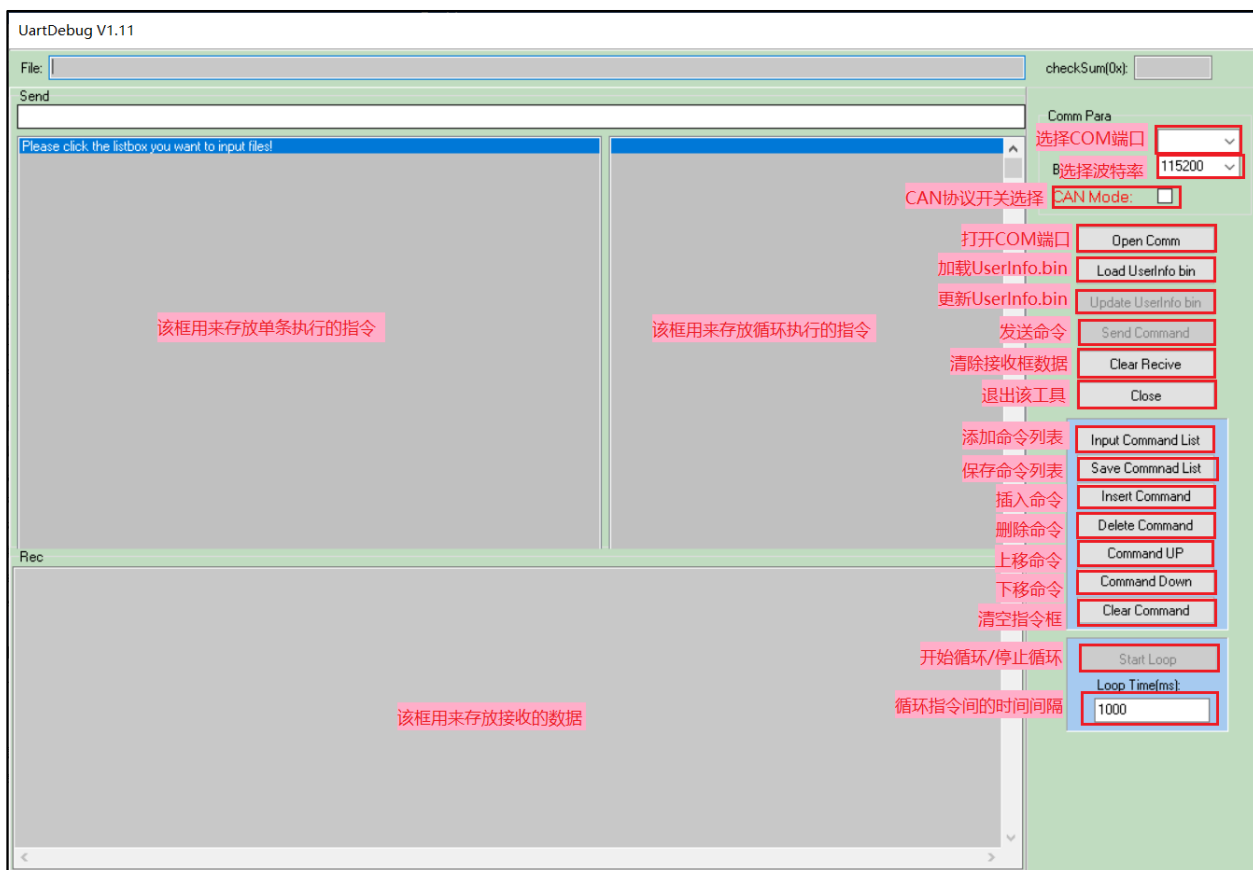


图 5-5: UartDebug 软件界面

由图 5-5 可知：UartDebug 软件界面左边框用来存放单条执行指令，右边框用来存放循环执行指令，下边框用来存放接收的数据，最右边个小框由上到下分别为：选择端口、配置波特率、COM 口的打开或关闭、加载 UserInfo.bin 文档、更新 UserInfo.bin 文档、发送指令，清除接收、添加指令列表、保存指令列表、插入指令、删除指令、指令上移、指令下移、清除指令、开始或停止循环和两条循环指令执行的时间间隔。

5.3 仿真指令的加载

1. 首先确认电脑（USB 转 Uart）与 LT268B 串口屏的 UART 已经连接，然后点击【Comm Number】选择出现对应的 Com Port，再点击【Open Comm】完成联机：

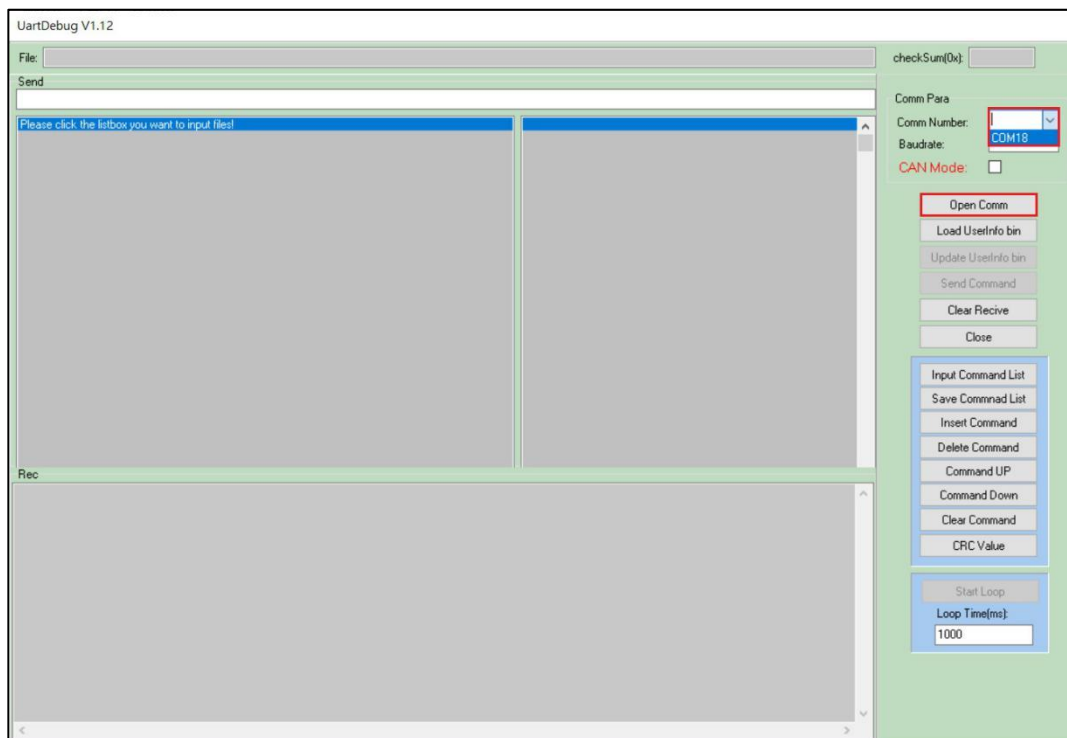


图 5-6：选择对应的 Com Port

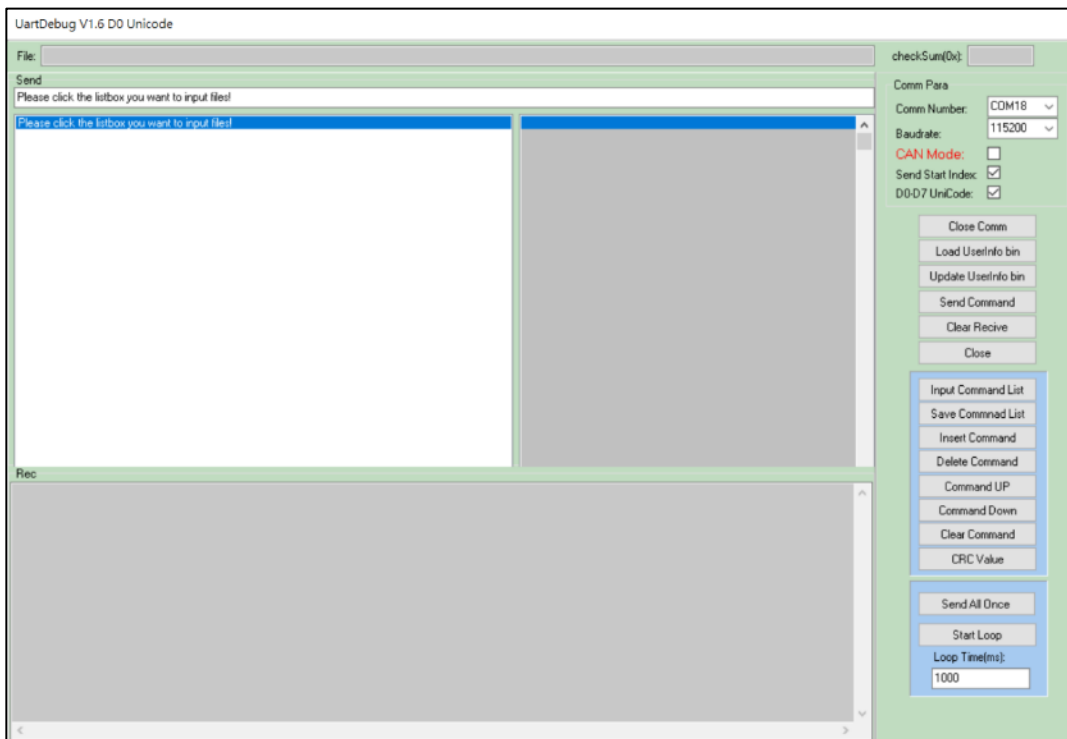


图 5-7：点击【Open Comm】完成联机

如果没单击指令存放框，直接就单击【Input Command List】就会跳出一个 “Please select a list box for input file” 框提醒你选择输入文件的列表框，如下图：

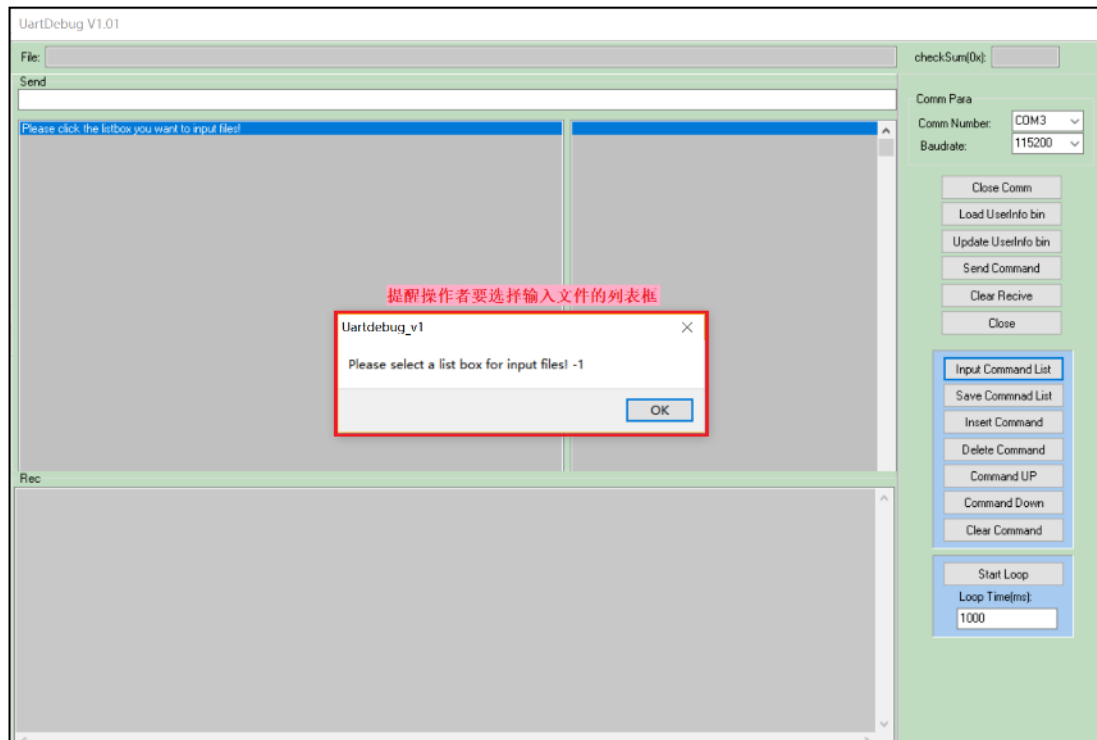


图 5-10：选择输入文件的列表框

3. 使用者如果使用 LT268B 的 UartTFT_Tool Demo 文件进行操作，在电脑 (USB 转 Uart) 与 LT268B 串口屏的 UART 连接供电时会出现如下图显示画面，那是因为 **UartTFT.ini** 指令文件有 9A 开机指令：（第 4.4.1 节 TFT 串口指令编译的步骤 4）开机时执行了一个秀图指令、一个滚动图片指令、一个按键指令：

9Ah #00: 0x80, 0x00, 0xD9, 0x00, 0xA0, 0x00 //Boot



图 5-11：9A 开机指令

5.4 仿真指令的发送

用户可以双击仿真指令，也可以先单击选中要发送的指令，再单击【Send Command】发送指令，电脑就会发出该仿真指令送达给串口屏，例如双击第 2 行 80 01（如下图 5-12），串口屏就会显示另一张图片（如下图 5-13A），再双击第 88 00，串口屏就会显示一 GIF 动画（如下图 5-13B），用户可以通过这种方式——验证在 UartTFT.ini 指令设定文件内的设计是否正确。

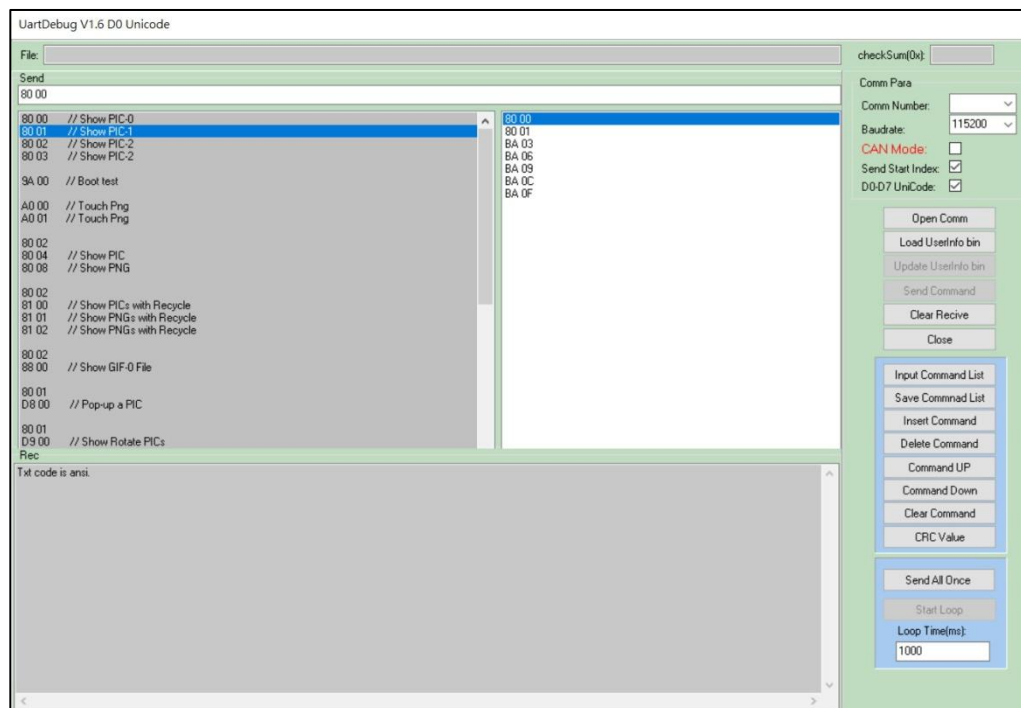


图 5-12：发出仿真指令给串口屏



图 5-13A：显示图片

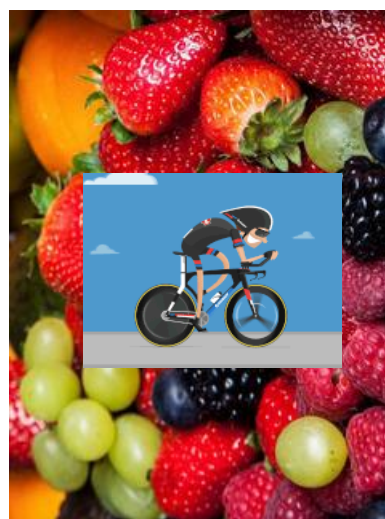


图 5-13B：显示 GIF 动画

针对仿真指令，用户可以用右侧的编辑窗口来对仿真指令文件（[Demo.txt](#)）做简易的修改、储存，或者是由外部文件编辑器对仿真指令文件修改后点击【Input Command List】重新读取该文件。

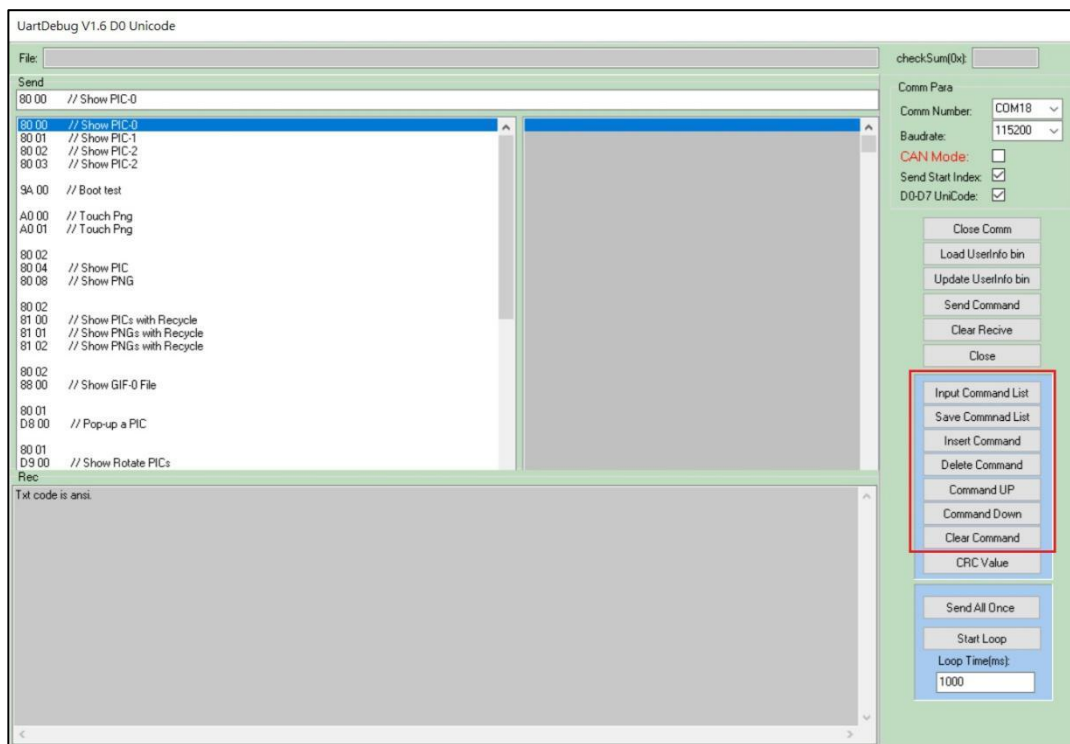


图 5-14：仿真指令文件的修改

根据第二章的表 2-2 主控端与 TFT 串口屏协议，主控端的系统或是主板透过 UART 串口传递显示指令给 LT268B 串口屏时，除了 指令码、序号、指令参数 外还要加上 1 个 Byte 的 起始码(固定为 0xAA)、2 个 Byte 的 CRC 码、4 个 Byte 的结束码（固定为 0xE4、0x1B、0x11、0xEE），但是在 UartDebug 的仿真指令下不需要加上起始码、CRC、和 4 个 Byte 的 结束码，因为 UartDebug 在程序运行上会自动加上，例如点击上面例题的 88 00，实际上 UartDebug 传给串口屏的信息为 0xAA + 88 + 00 + CRC(1) + CRC(2) + 0xE4 + 0x1B + 0x11 + 0xEE，这样可以简化用户编辑仿真指令文件的复杂度。

5.5 发送循环仿真指令

- 1) 首先单击右边白色循环仿真指令存放框，然后单击【Input Command List】会跳出一个新窗口，选取要添加的循环仿真指令文档，确认文档选择正确后单击打开即可。具体操作如下图：

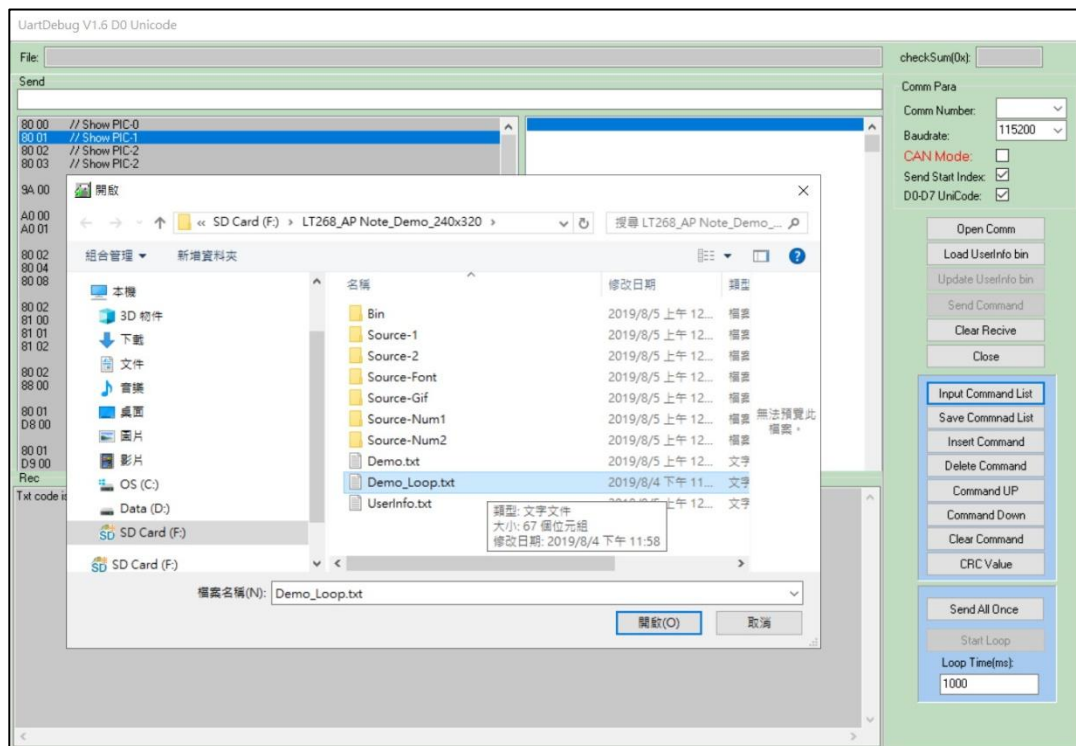


图 5-15：选取循环仿真指令文档

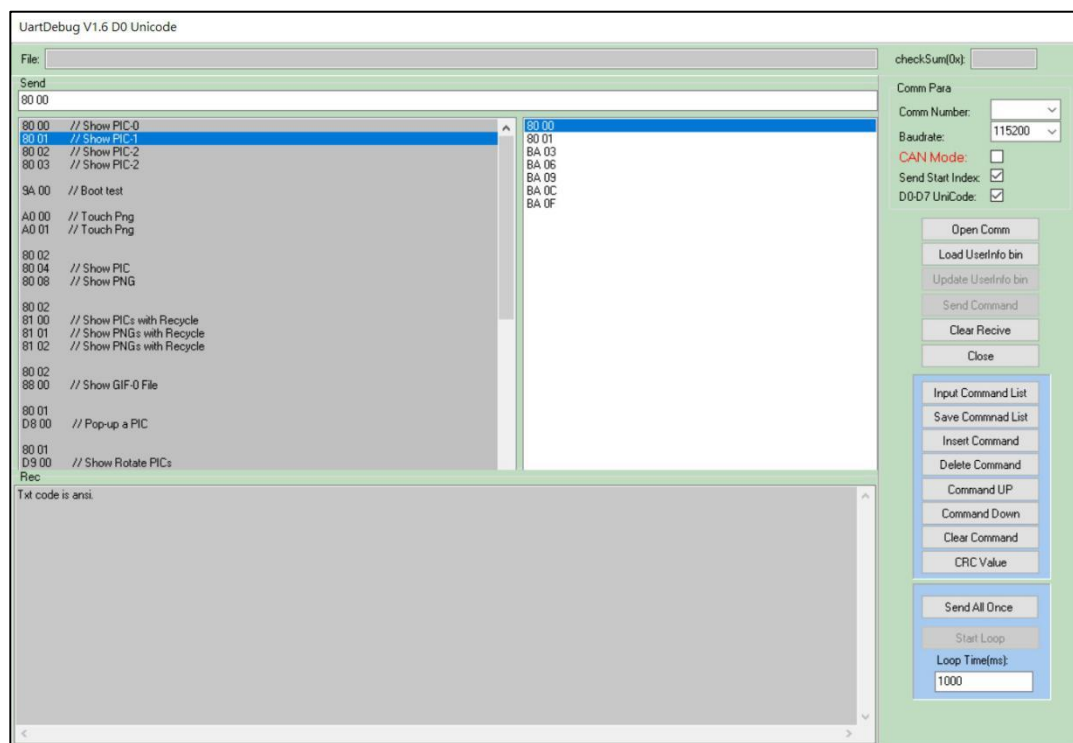


图 5-16：选取循环仿真指令文档后

- 2) 选取循环仿真指令文档的另一种方式，就是通过单击左边的指令选择要添加到右边的循环仿真指令，然后按右键会出现一个 “Add to Loop” 的框，单击该框即可将该指令添加到循环仿真列表中，具体操作如下图：

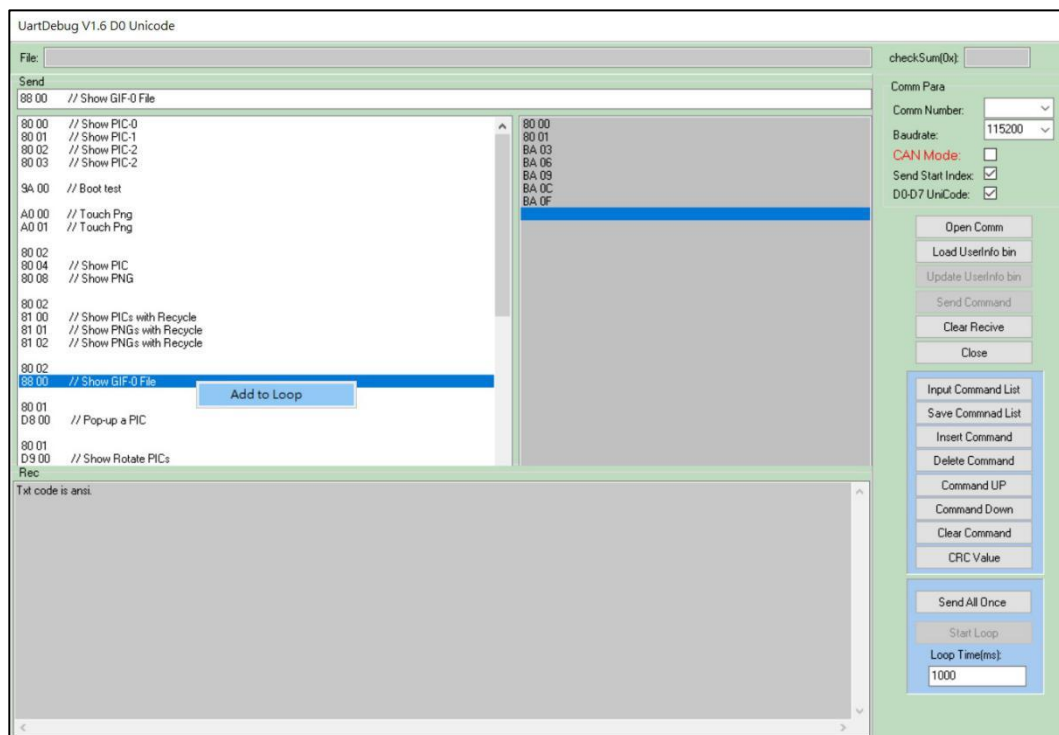


图 5-17：添加循环仿真指令

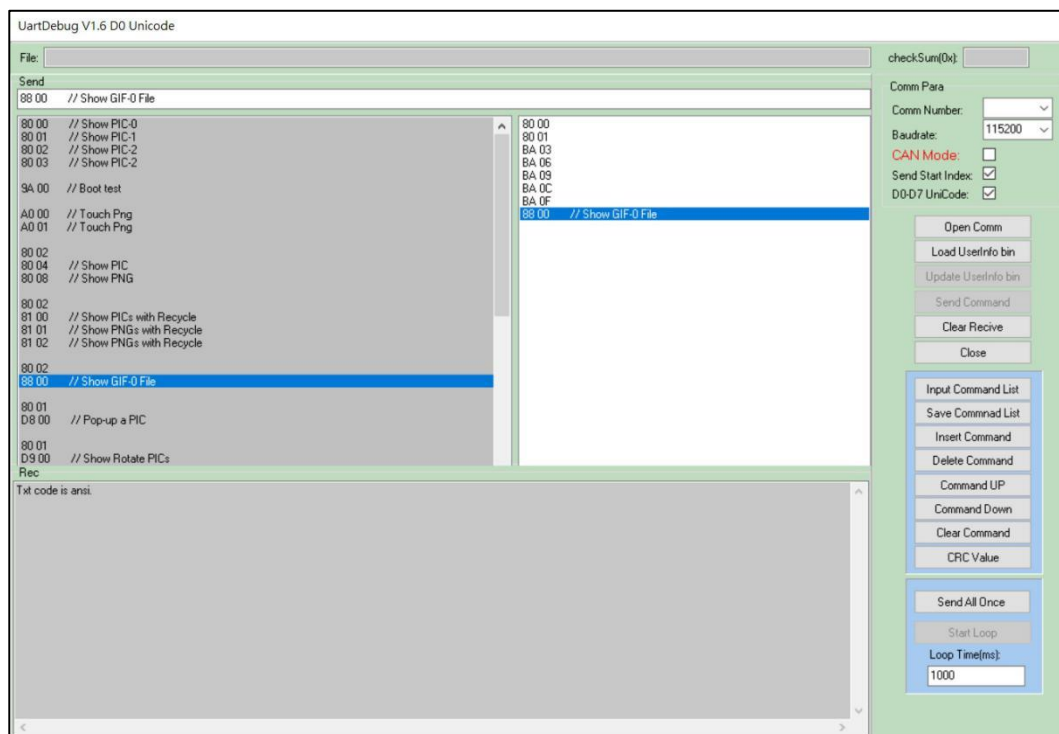


图 5-18：添加循环仿真指令后

3) 单击【Start Loop】即可实现仿真指令的循环发送，具体操作如下图：

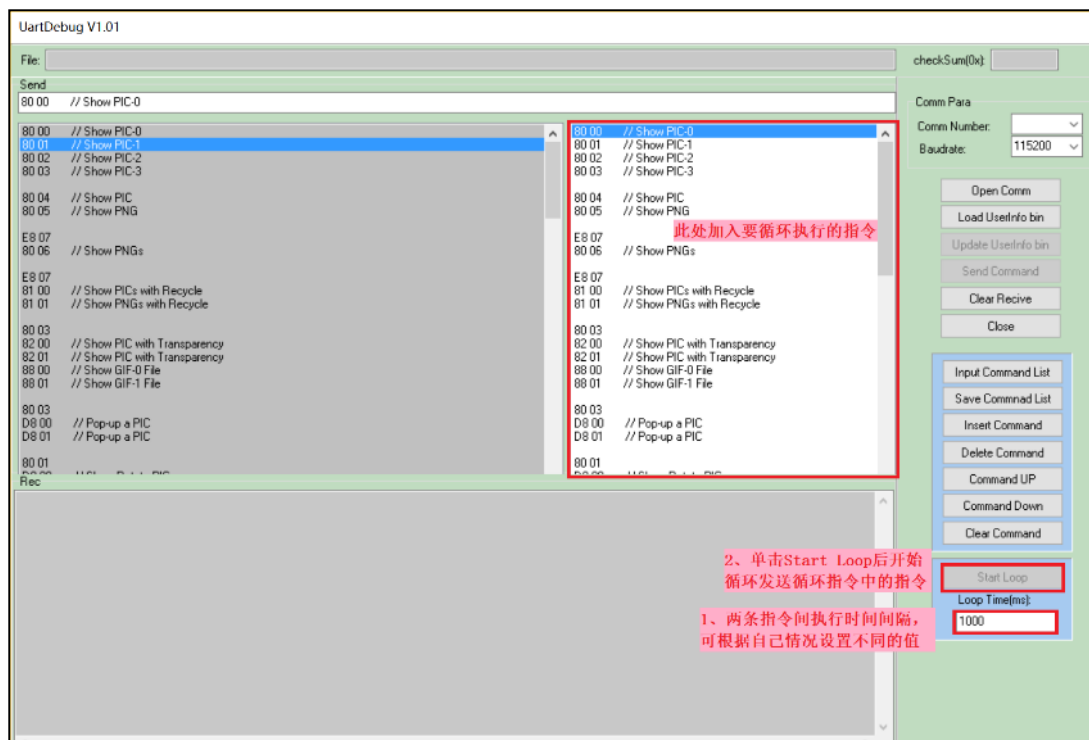


图 5-19：进行仿真指令的循环发送

4) 循环发送过程中，Start Loop 框会变成 Stop Loop 框，要停止循环发送只需单击该框即可。

5.6 指令列表的更改与保存

- 1) 指令的更改，首先单击要修改的指令，然后该指令会在 Send 下面的空白框显示出来，在 Send 下面的空白框对指令进行修改，修改好后按 Enter 键即可保存更改后的指令，具体操作如下图：

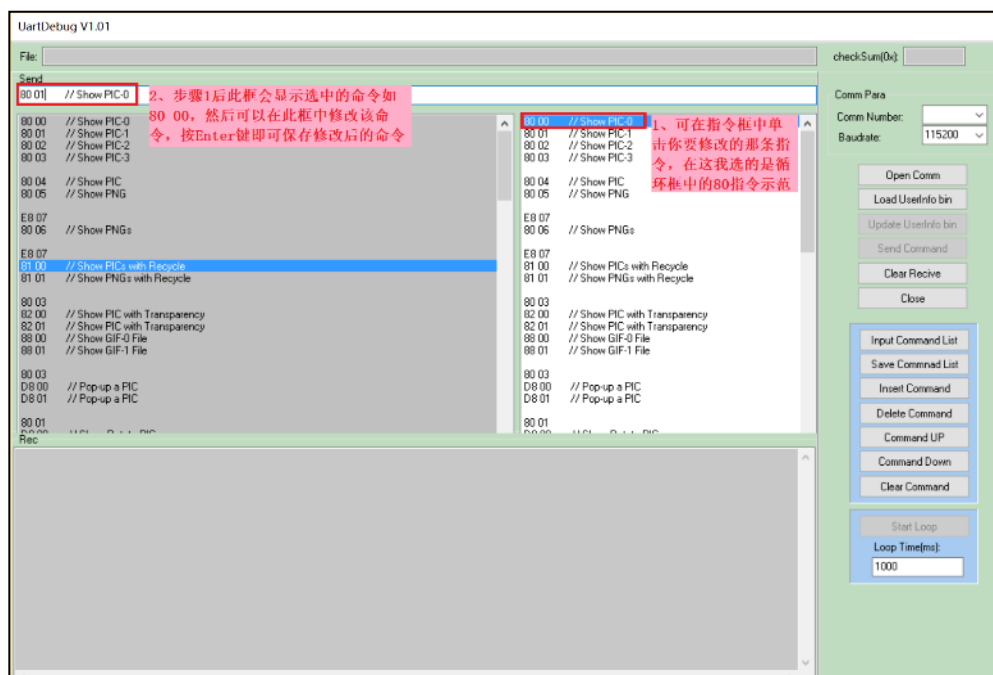


图 5-20：指令的更改

- 2) 插入指令，首先单击选择要插入指令的位置，然后单击 Insert Command，最后在 Send 下面的空白框中将指令改为自己要插入的指令，按 Enter 键保存即可完成指令的插入，具体操作如下图：

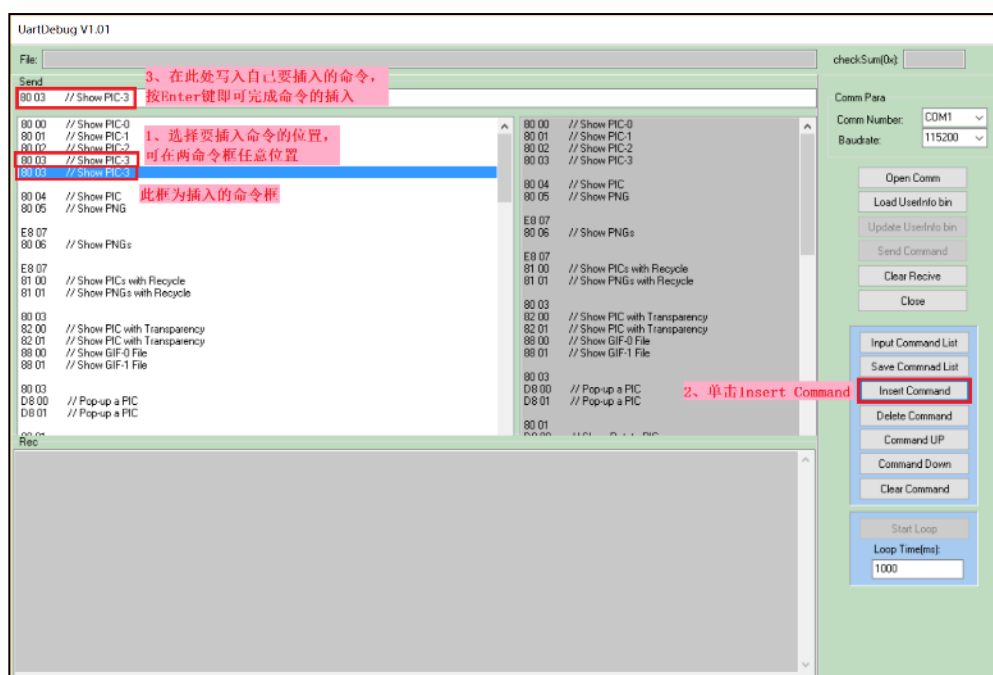


图 5-21：指令的插入

- 3) 单击选中要删除一条指令，然后单击 Delete Command 即可完成删除，具体操作如下图：

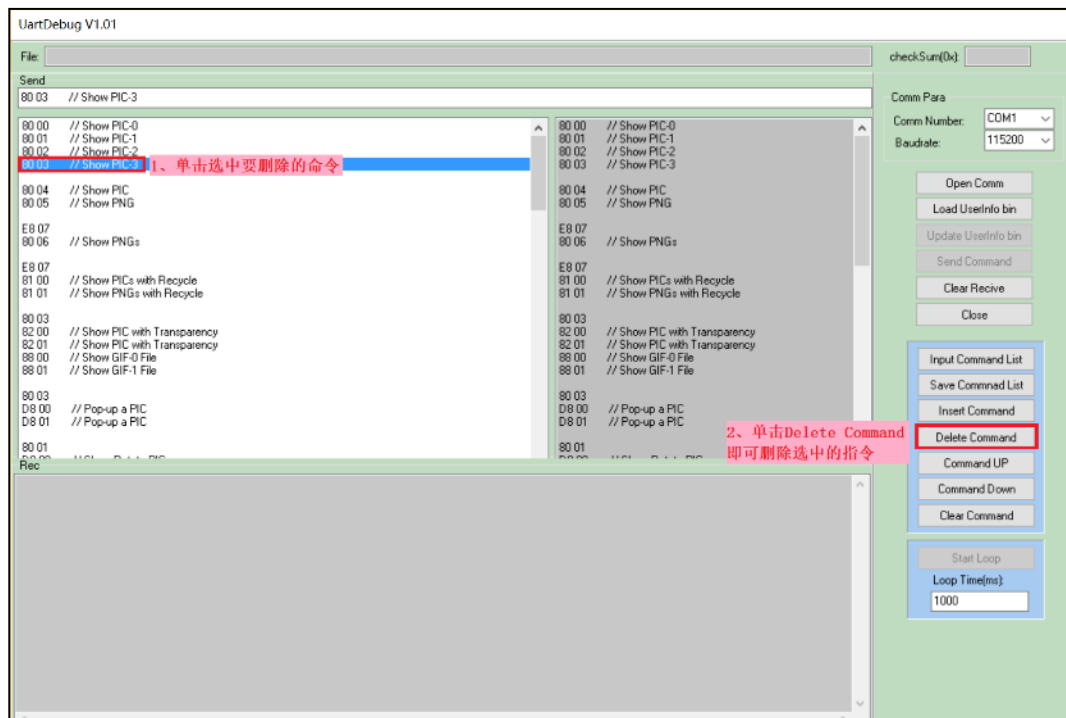


图 5-22：删除一条指令

- 4) 指令上移或下移，首先单击选中要上移或下移的指令，然后单击 Command UP 即可实现指令的上移，单击 Command Down 即可实现指令的下移。具体操作如下图所示：

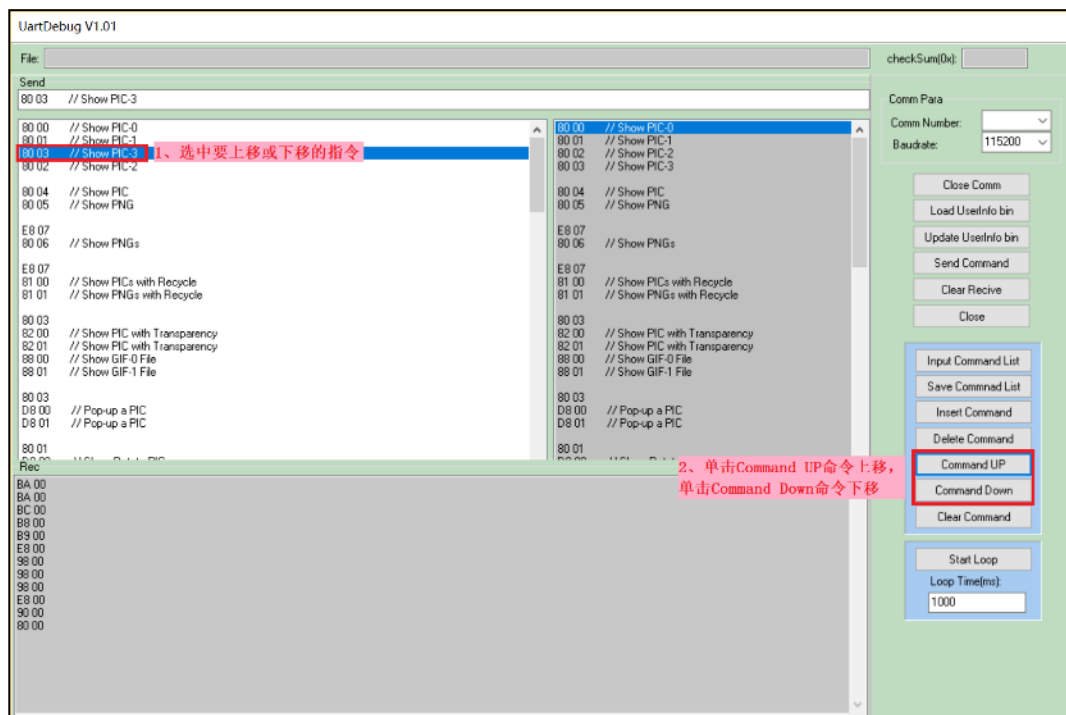


图 5-23：指令的上移或下移

- 5) 首先在两指令框中单击要将框中的指令保存为文档的指令框，单击选中要将框中的指令保存为文档的指令框后，单击 Save Command List 会跳出一个窗口，选择要保存的路径，给要保存的文档命名，最后单击保存即可实现将两指令框中其中一个框中的指令保存为文档。注意不能一下子直接保存两个框中的指令，要将两框中的指令保存为文档，只能一个框一个框的保存。具体操作如下图所示：

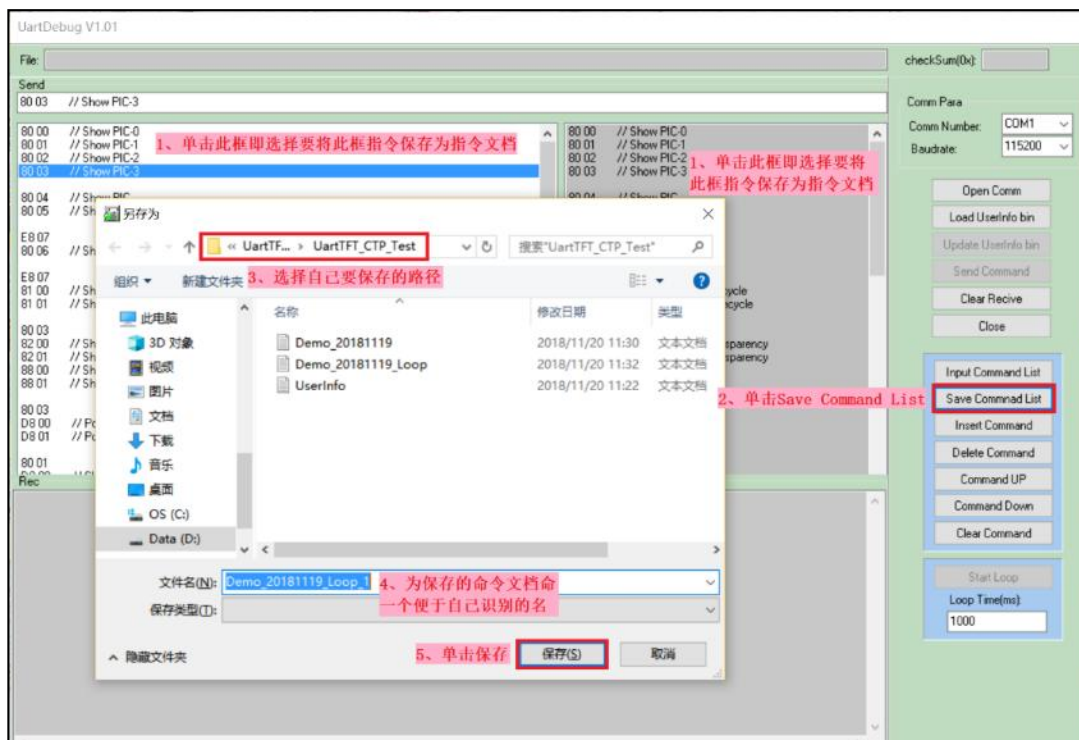


图 5-24：指令的保存

5.7 清除指令框指令与接收框数据

- 清除指令框中的指令，首先选中要清除指令的指令框，然后单击 Clear Command 后会跳出一个新窗口提醒你是否真的要清除指令，确定要清除指令的话单击 OK，不想清除的话单击 Cancel。具体操作如下图所示：

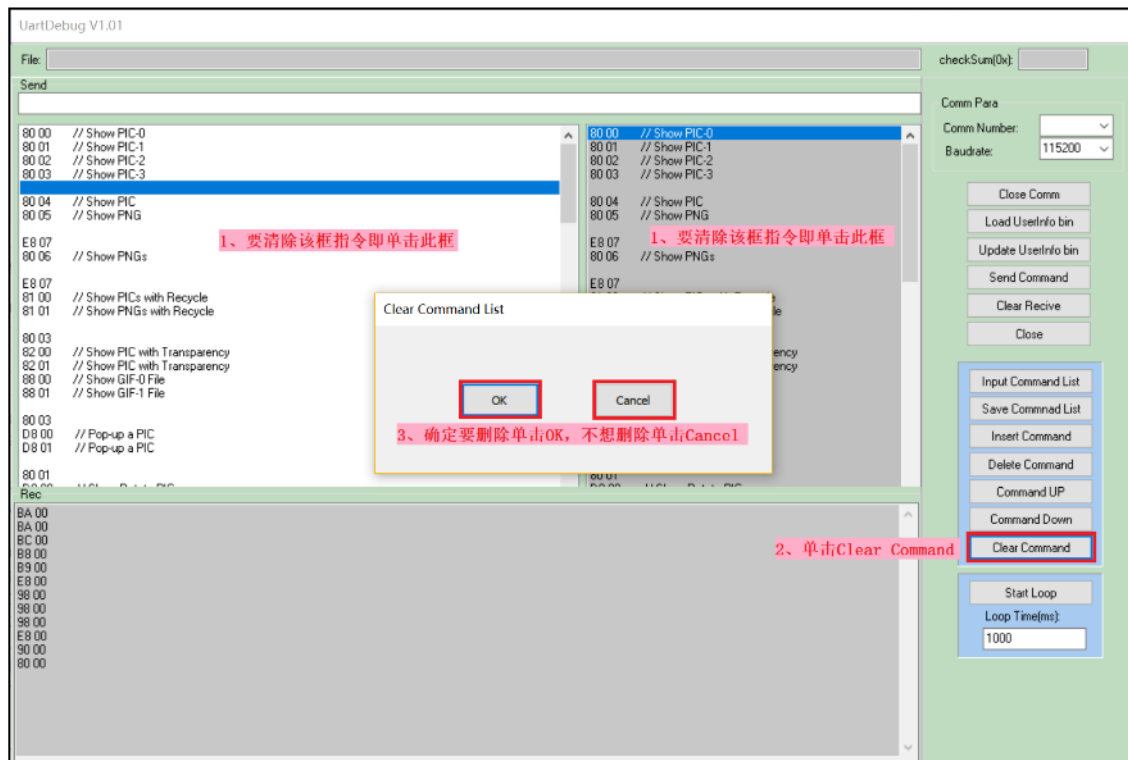


图 5-25：清除指令框中的指令

2) 清除接收的数据，只要单击 Clear Receive 即可清除接收的数据。具体操作如下图：

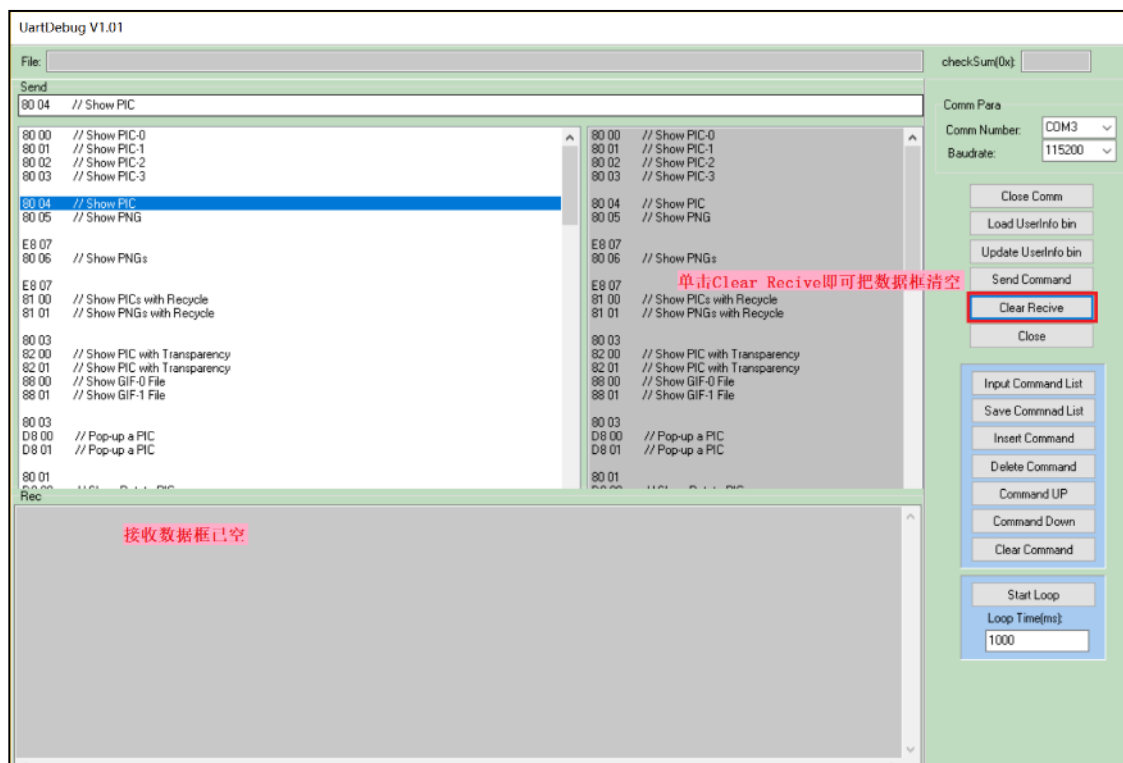


图 5-26：清除接收的数据

5.8 加载与更新 UserInfo.bin 文件

- 1) 在 COM 口打开后, 点击 Load UserInfo Bin 后会弹出一个窗口, 进入存放 UserInfo.bin 文档的文件夹, 选择要加载的 UserInfo.bin, 确认选择正确后单击打开即可加载 UserInfo.bin 文档, 如下图:

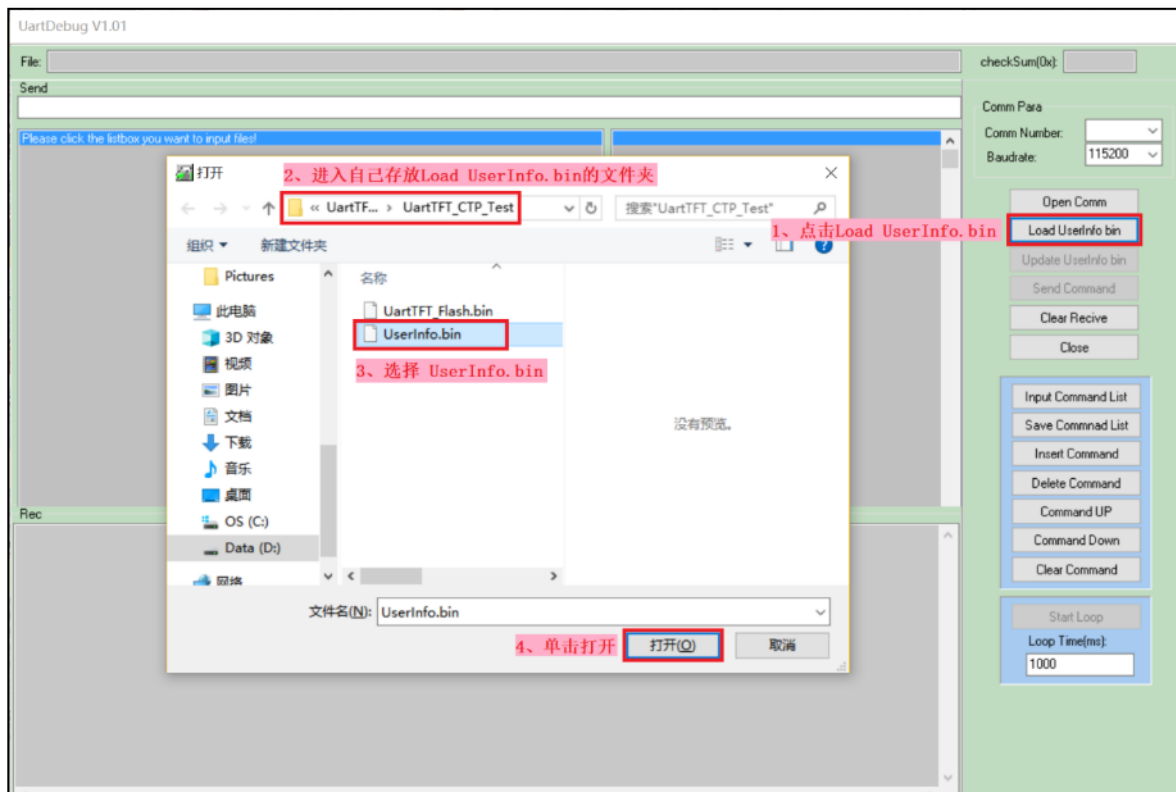


图 5-27: 加载 UserInfo.bin 文档

- 2) 执行完加载指令后, 点击 Update UserInfo.bin, 就可以更新 UserInfo.bin, 如图 5-28; 更新时 TFT 屏会显示如图 5-29。

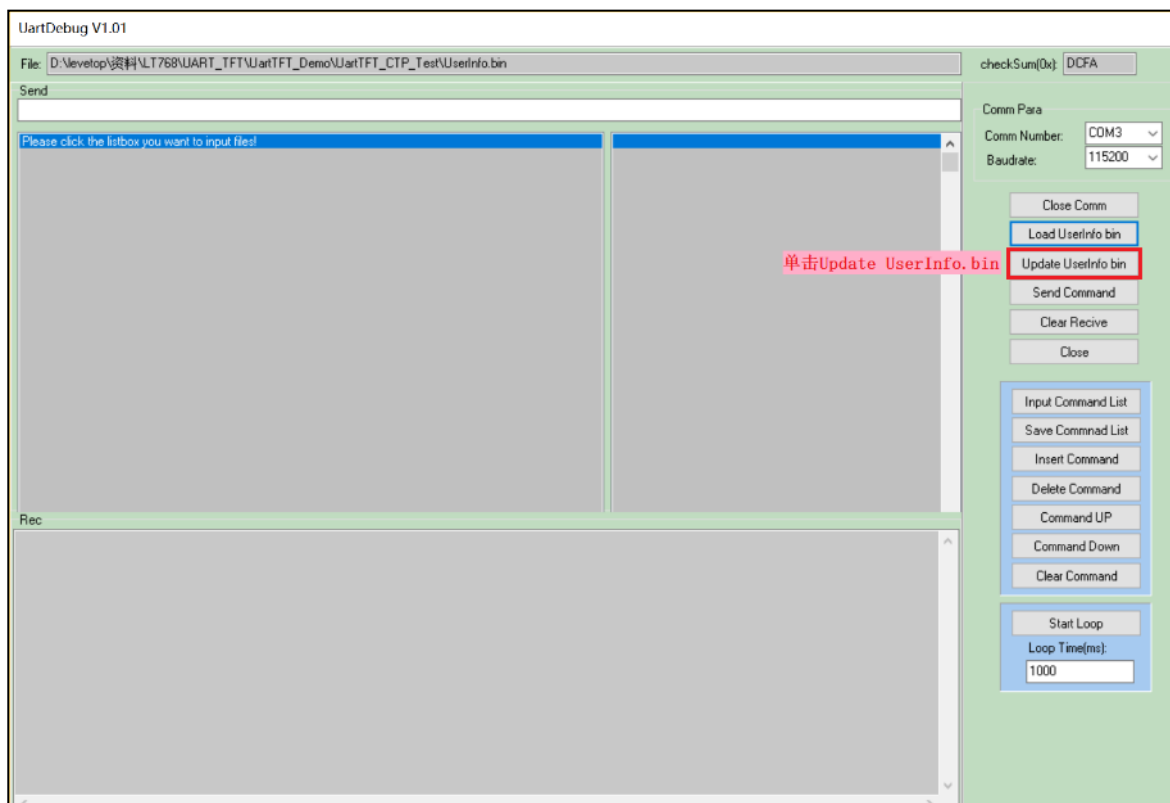


图 5-28: 更新 UserInfo.bin 文档

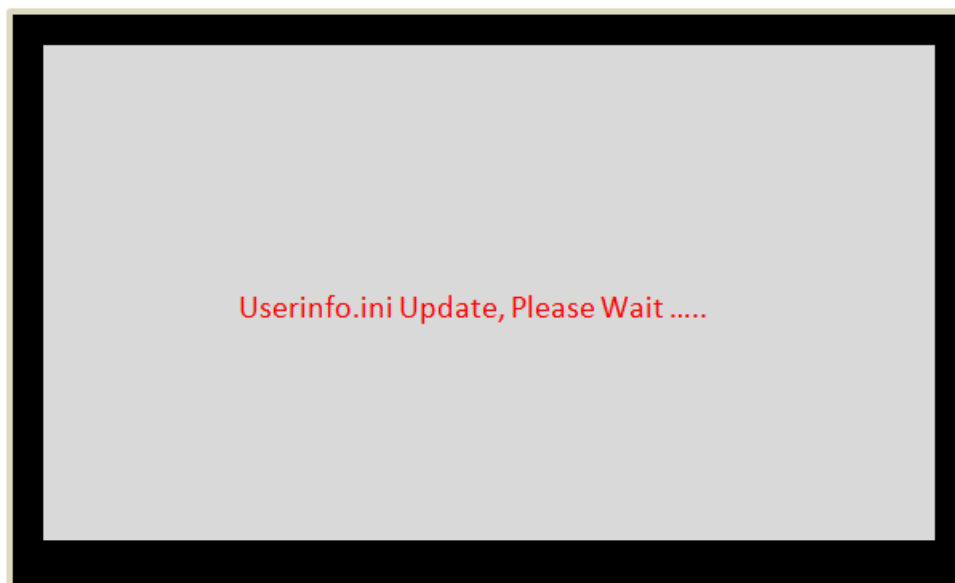


图 5-29: UserInfo.bin 文档更新中

3) 更新成功后 TFT 屏会显示指令更新成功，如下图：

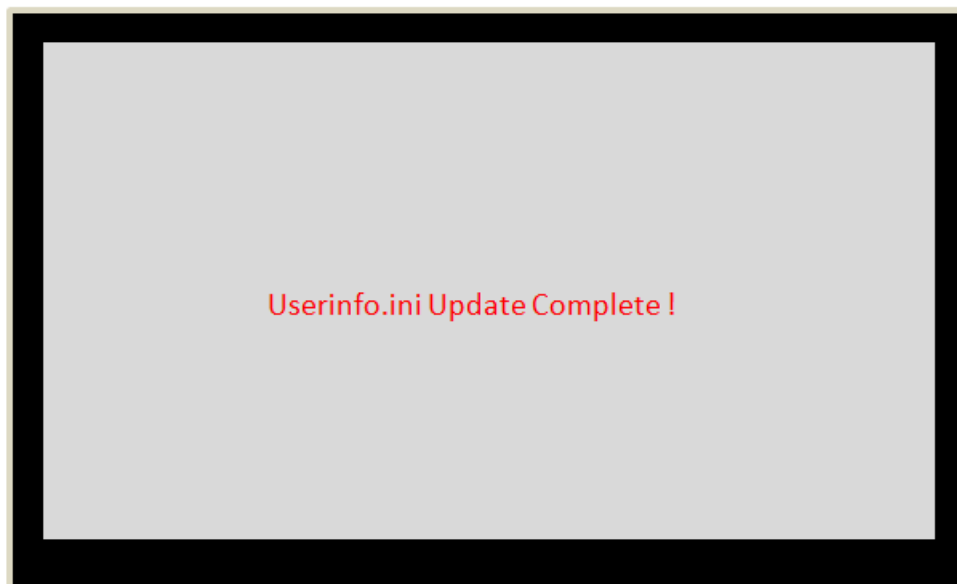


图 5-30：UserInfo.bin 文档更新成功

注意：UserInfo.bin 是透过 UartTFT_Tool.exe 软件生成，具体生成步骤请参考本手册第 4.4.1 节 TFT 串口指令编译。

6. MCU 码与 Flash 更新

6.1 LT268B 的主程序更新

LT268B 具有 USB 接口，同时内部含有更新程序，可以让用户用电脑由 USB 接口对 TFT 屏上的 SPI Flash 或是 LT268B 串口程序进行更新。首先至本公司网页(www.levetop.cn)下载 **LT268B_ISP_Vxx.rar**，然后解压缩生成 **LT268B_ISP_Vxx.exe** 档案。更新 LT268B 的 MCU 程序前需要将 LT268B 板上的 “BUSY” 引脚拉低，进入 USB_Update 模式。然后用 USB 线连接板子与电脑，如下图所示：



图 6-1：用 USB 更新

执行电脑上软件 LT268B_ISP_Vxx.exe，应用程序会自动选择 LT268B 的串口号，只需点击 Open Port。（若选择错误的串口号，无法进行下一步操作，防止选错串口号）。如下图：

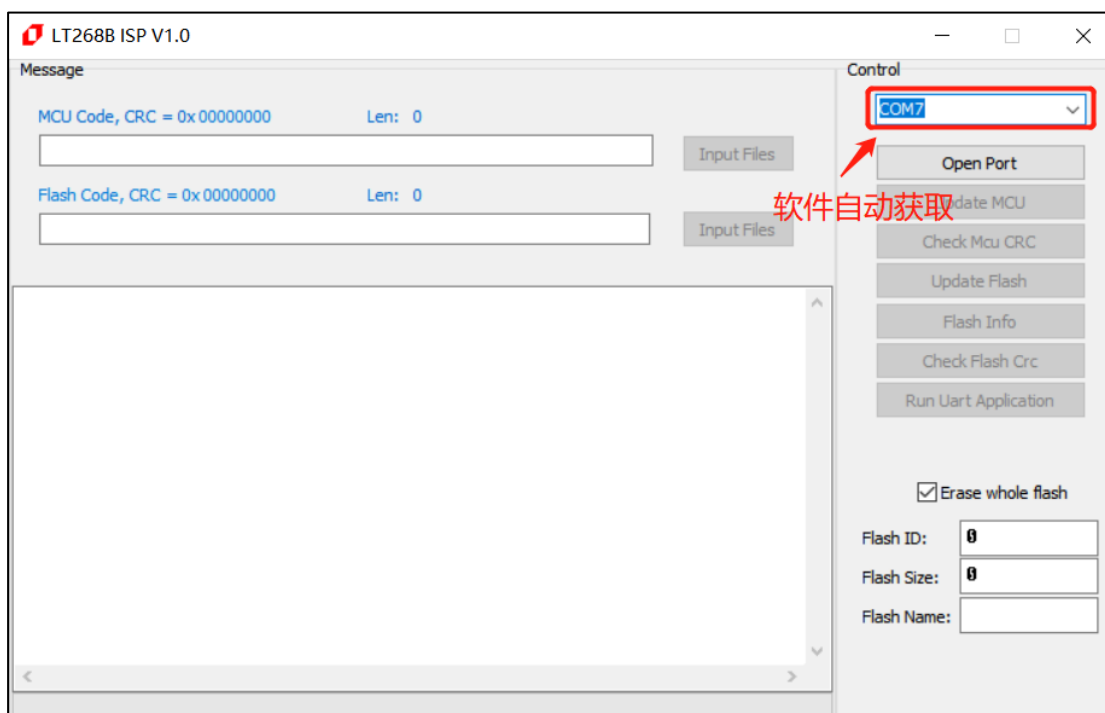


图 6-2：开启软件 LT268B_ISP_Vxx.exe

点击“Open Port”开启通信，软件界面如下图所示：

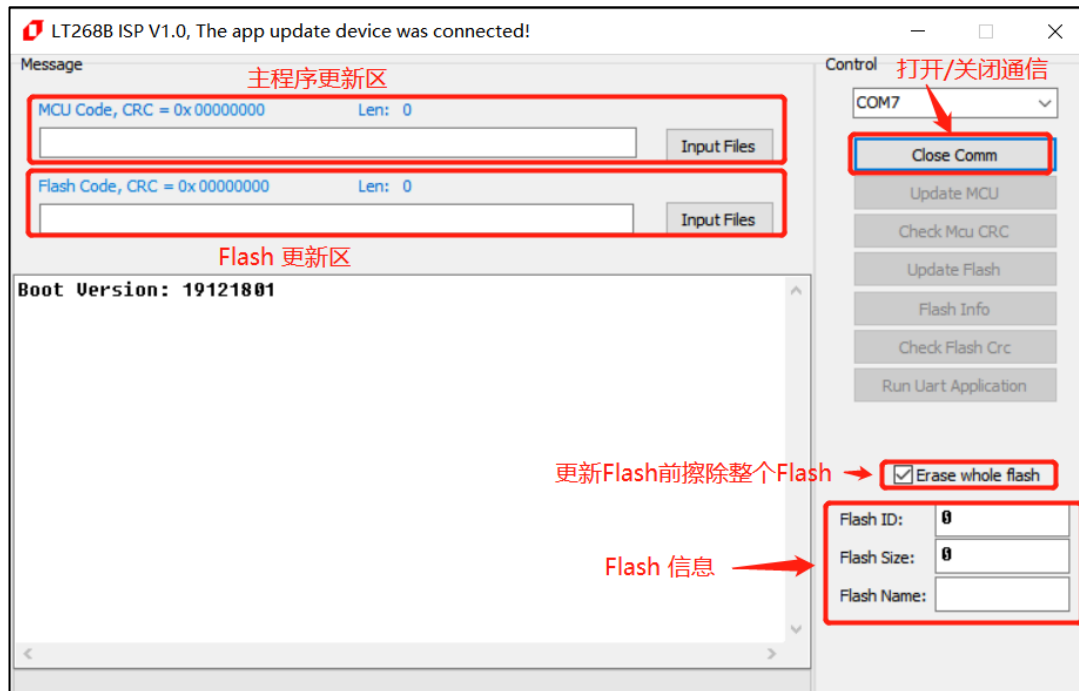


图 6-3：选择更新项目

在主程序更新区，点击 Input Files，打开主程序文件，如 LT268B_Demo_V1.1.bin 文件，显示如下
图：

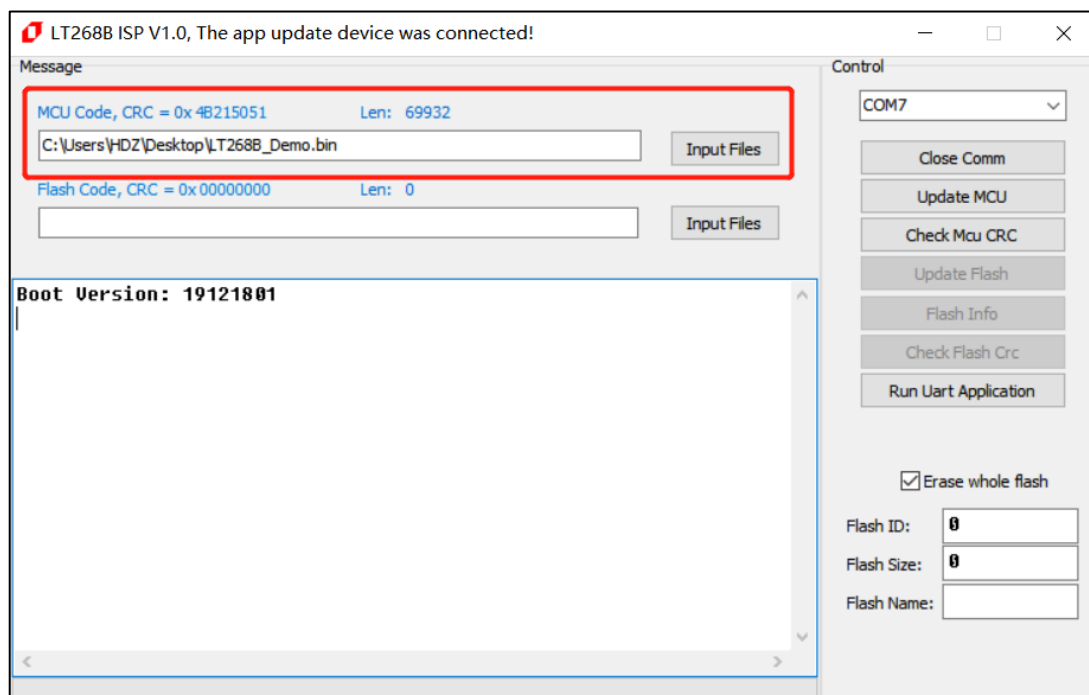


图 6-4：选择更新 LT268B 内部的 MCU 程序

点击“Update MCU”进行更新 MCU 程序，烧录成功显示如下图：

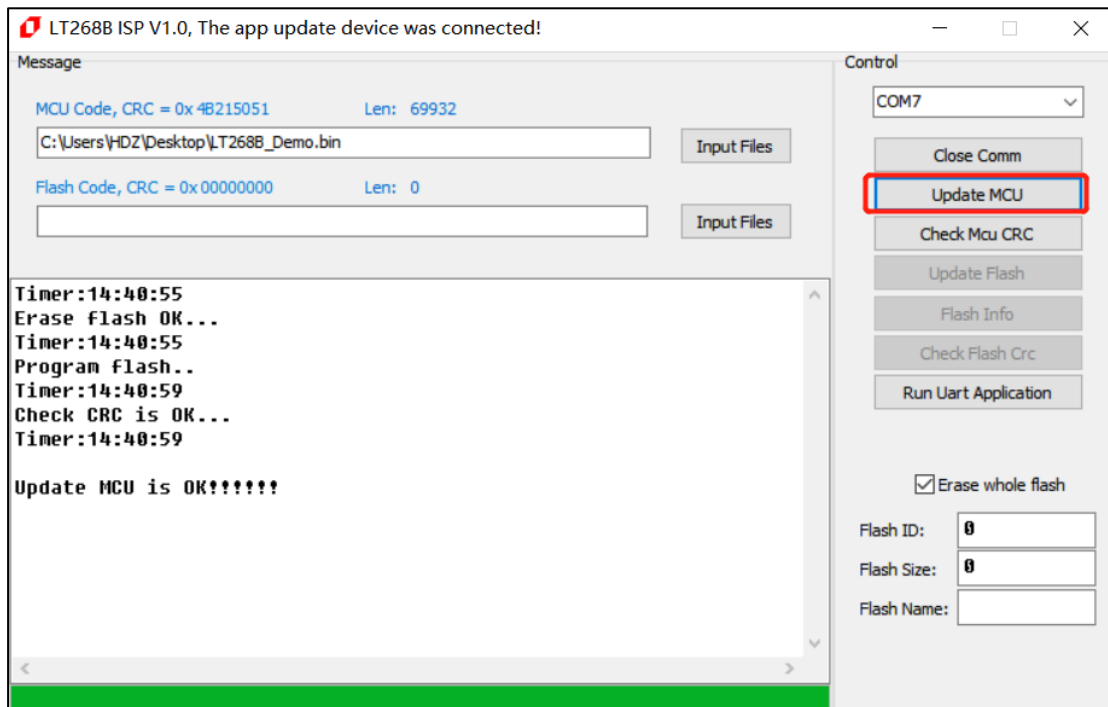


图 6-5：LT268B 内部 MCU 程序更新完成

通过 Check MCU CRC 按钮，可以检查导入文件与当前 MCU 设置是否一致，方便校验版本。（上面 Update MCU 已经包含 Check，无需再 Check MCU CRC），当 CRC 不一致时，返回信息如下图：

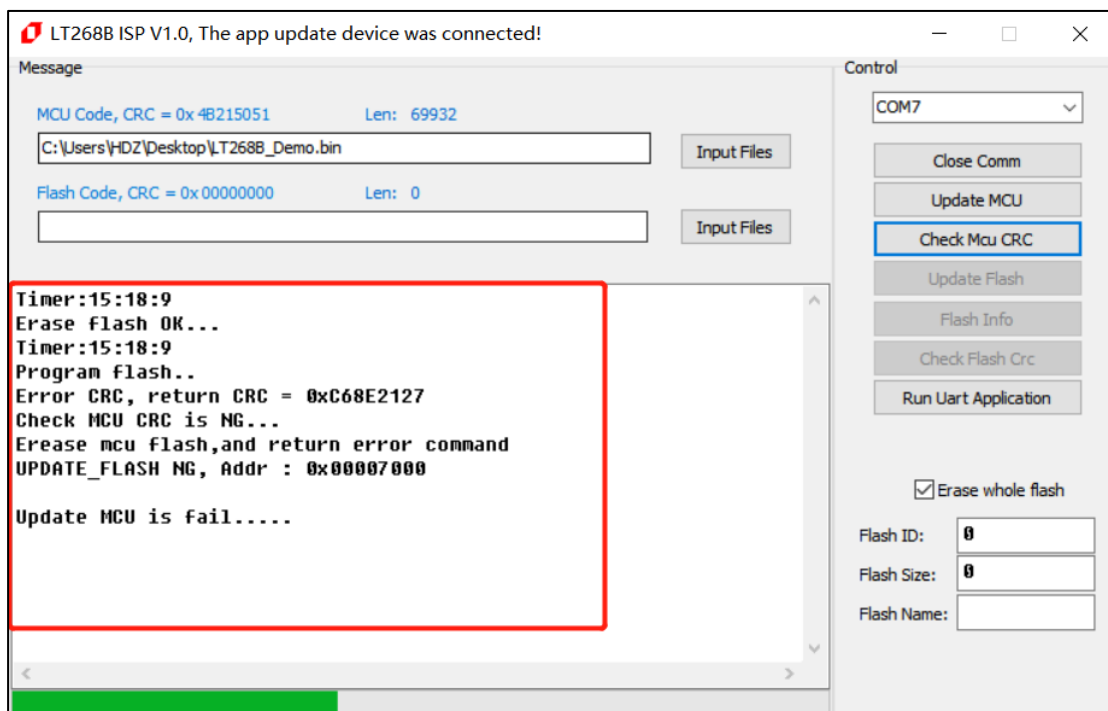


图 6-6：检查导入文件与当前 MCU 设置是否一致

烧录完成后可点击“Run Uart Application”进行重置和运行程序，也可重新上电或复位进行重置和运行程序。（注：进行“Run Uart Application”操作时会使 MCU 退更新模式，使软件不能识别串口，若要重新进入更新模式需按下 RST0 按键进行复位。），如下图：

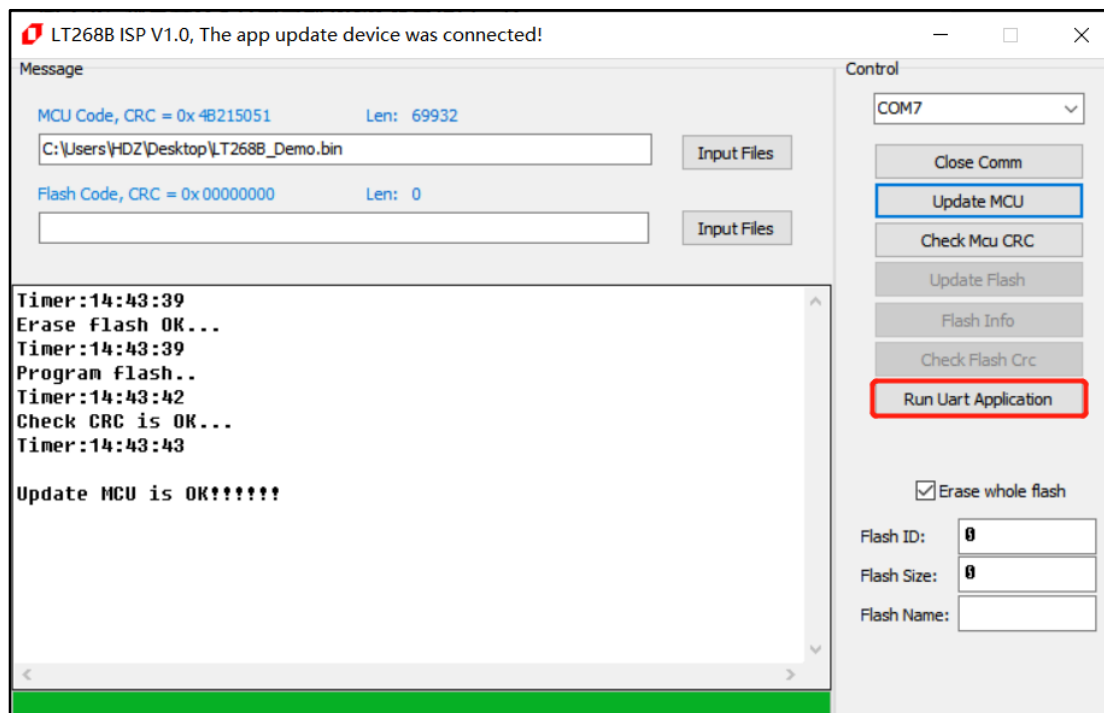


图 6-7：更新完成后进行重置和运行程序

6.2 USB 更新外部的 SPI Flash

先添加 Flash 更新文件，文件类型为.bin 格式，再点击“Flash Info”可查询 Flash 信息。如下图：



图 6-8：查询 Flash 信息

点击“Update Flash”对 Flash 进行更新。更新大文件时，选择擦除整个 Flash 速度会更快，更新小文件是选择不擦除整个 Flash 速度更快。如下图：



图 6-9：更新 Flash

更新大文件时, 选择擦除整个 Flash (擦除 16M 的 Flash 时间为 30S 左右, 擦除 32M 的 Flash 为 60S 左右), 更新完成后会自动检测 Flash CRC(勾选了 Check Nor CRC), 全部完成后显示如下图:

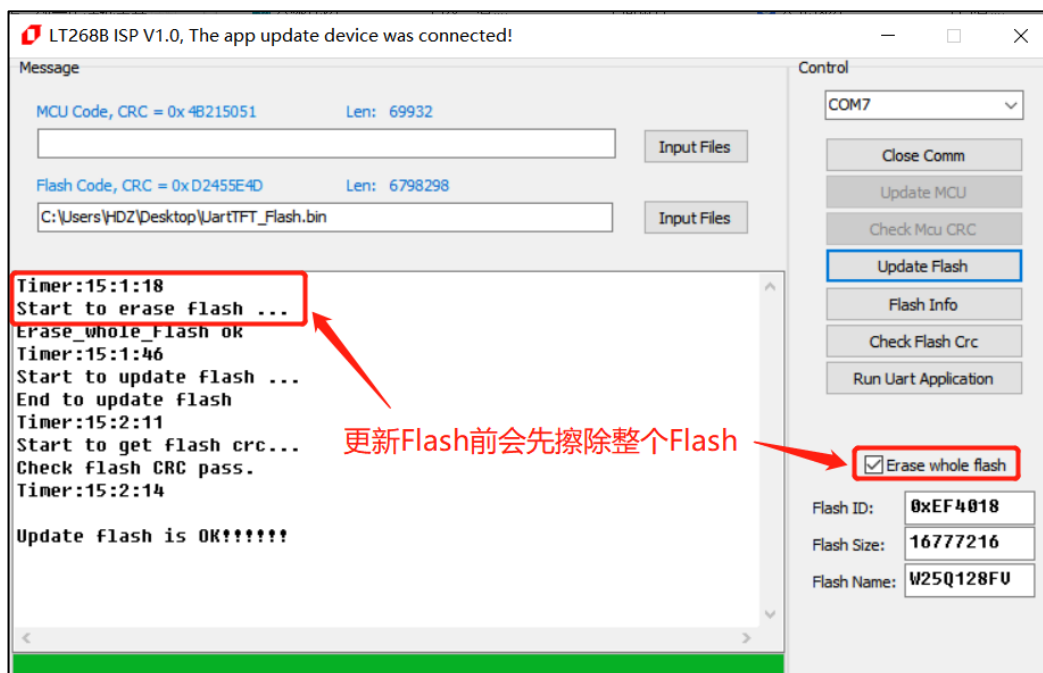


图 6-10: 更新 Flash 完成

更新小文件时, 不选择擦除整个 Flash,更新完成后会自动检测 Flash CRC(勾选了 Check Nor CRC), 全部完成后显示如下图:

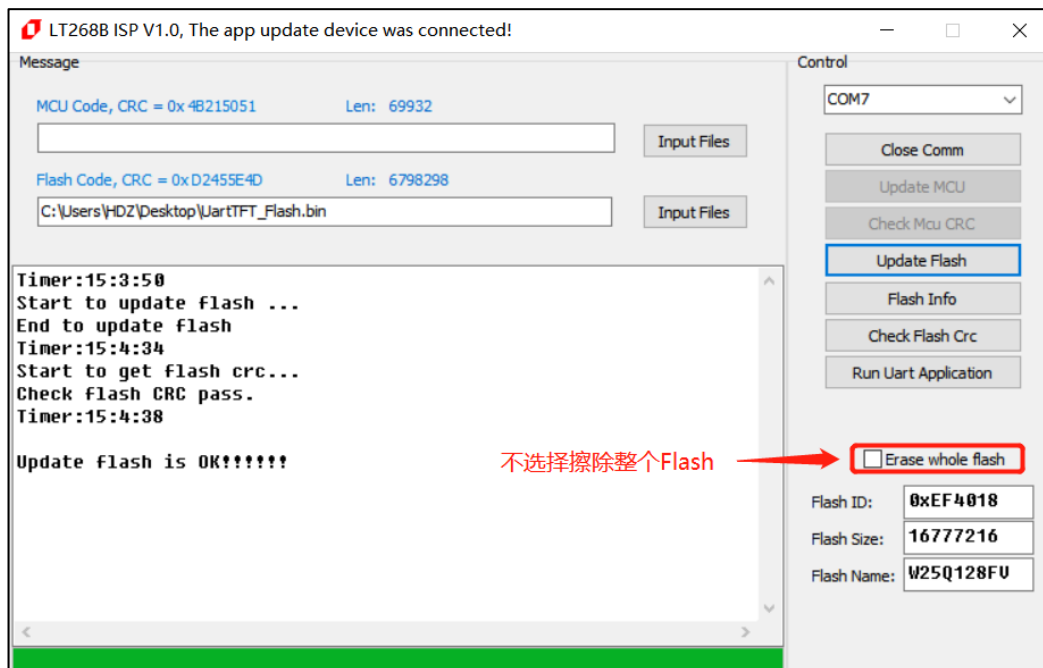


图 6-11: 更新 Flash 完成

目前软件已兼容大多数 Winbond 的 NOR Flash, 可在软件文件夹中的 Flash.ini 文件中自行添加 Flash 信息和修改 Flash 的片选, 在软件的同目录下用记事本的方式打开 Flash.ini 文件, 如下图:

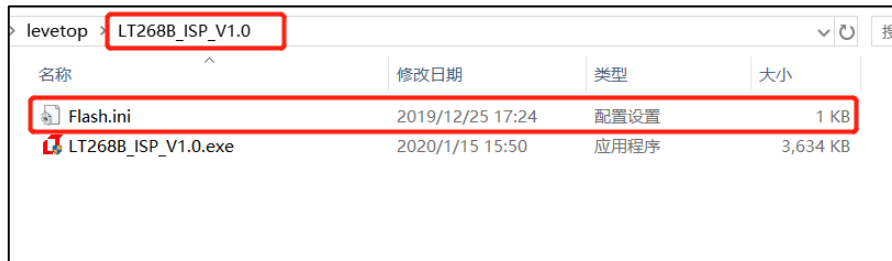


图 6-12: 软件文件

Flash.ini 文件内容, 可以按照格式添加 Flash ID 往后的内存信息, 如下图:



图 6-13: 添加 Flash ID

6.3 SD 卡更新外部的 SPI Flash

LT268B 也可以用 SD 卡更新外部的 SPI Flash，首先用电脑将要更新的 SPI Flash 数据档案“UartTFT_Flash.bin”存放在一个 SD 卡的根目录上，而 LT268B 模块上的“BUSY”引脚不需要拉低，将 TFT 模块直接插入电源后，再把 SD 卡插入模块上的 SD 卡巢内，LT268B 的程序会自动侦测然后进入更新画面，如下图：

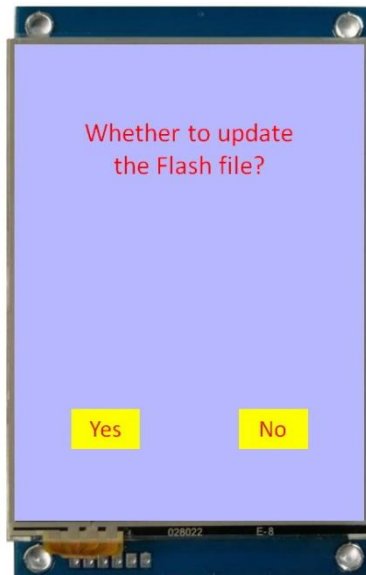


图 6-14：进入更新画面

按下“Yes”后 LT268B 就开始读取 SD 卡上的“UartTFT_Flash.bin”数据，开始进行对 SPI Flash 的刻录动作，如下图所示，刻录结束后 LT268B 会自动重新启动。



图 6-15：SPI Flash 数据更新中

7. 版本记录

表 7-1：应用手册版本记录

版 别	发 布 日 期	改 版 说 明
V1.0	2020/01/03	Preliminary version (初版)。
V1.1	2020/6/27	1. 更新 C0~C3 指令。

8. 版权说明

本文件之版权属于 深圳市乐升半导体 所有，若需要复制或复印请事先得到 乐升半导体 的许可。本文件记载之信息虽然都有经过校对，但是 乐升半导体 对文件使用说明的规格不承担任何责任，文件内提到的应用程序仅用于参考，乐升半导体 不保证此类应用程序不需要进一步修改。乐升半导体 保留在不事先通知的情况下更改其产品规格或文件的权利。有关最新产品信息，请访问我们的网站 [Http://www.levetop.cn](http://www.levetop.cn) 。