乐升半导体
LEVETOP

LEVETOP

UI_Editor-III

User Manual

V1.10

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

Revision

Version	Date	Description
V1.0	2024/12/11	First Edition
V1.01	2025/03/05	User Manual: 1. Add the description of UI_projectname.h 2. Add the description of OTA Difference Upgrading 3. Add the description of the tool for converting UI_Editor-II projects to UI_Editor-III formats (LT_UI_Convertor_II_to_III.exe) Software: 1. Generate the UI_projectname.h after the compilation 2. Add OTA Update Address Table 3. Add the tool for converting UI_Editor-II projects to UI_Editor-III formats (LT_UI_Convertor_II_to_III.exe)
V1.1	2025/09/09	User Manual: Add the description of the SlideMenu_Pro widget Software: Add SlideMenu_Pro widget

Copyright

Contents

Figure

Table

1. Introduction

UI_Editor-III is the 3rd generation UI editing tool for UartTFT panels. It is designed for LT7589 UartTFT controllers. This manual is to illustrate how users can utilize this tool to implement UI designs.

1.1. UartTFT Panel Hardware & Software Structure

A UartTFT panel combines MCU, UartTFT controller, and TFT display into one product. Via serial Uart interface, a UartTFT panel can receive commands from the host MCU, and display predefined pictures and contents to the connected TFT panel. It can improve the efficiency while displaying pictures/contents to the TFT panel and reduce the time it takes for the host MCU to process graphic data.

Figure 1-1 shows an application example of UartTFT Panel using LT7589.
If long distance communication is needed, it usually needs to add RS232 or RS485 driver chip.

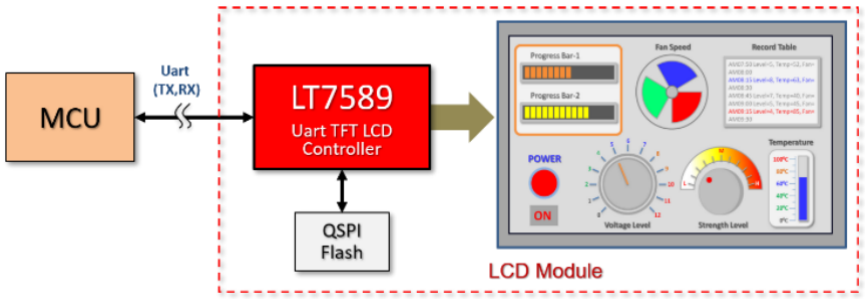


Figure 1-1: Application Diagram

A UartTFT panel has to be setup properly before put into use. Developers shall utilize UI Editor-III to setup the panel, and plan the flow of the display contents, including pictures, text, animations, and effects they want to achieve. After the design is done, developers may export a Bin file through UI_Editor-III. Developers can then program the bin file to SPI Flash via an SD card, a USB disk, or a dedicated SPI Flash programmer. And then the Host MCU can communicate with the Uart TFT panel through the Uart interface. Developers may also simulate their design via a USB-to-Uart (RS232) control cable, which means users can pre-verify the display result.

1.2. UI_Editor-III Settings & Development Steps

There are five steps when creating a new project with UI_Editor-III:

1、Get the UI material ready, refer to *Material Format*;

2、Create a new project, refer to *Create a New Project - Procedure*;

3、Design the UI pages, refer to *Widget* for various widget explanation;

4、Compile the project. Developers may check their design on UI_Emulator-III, a simulation tool for UI_Editor-III projects. Refer to *UI_Emulator-III*;

5、Programming to a UartTFT panel, refer to *Appendix 1 - Programming*.

2. UI_Editor-III Installation

2.1. Download the software package

Developers may download UI_Editor-III software package at Levetop official site: https://www.levetop.cn/en/index.aspx

As shown in Figure 2-1, click on [Download]

Figure2-1：Download

Click on [UartTFT Development Tools], and then click [UI_Editor/Emulator]. Select [UI_Editor-II tool kits (V3.11)] to download it.

Figure 2-2：Select UI_Editor-III tool kits

Figure 2-3 shows an example of the downloaded tool kits. Please note that the latest version number may not be the same as the example here.

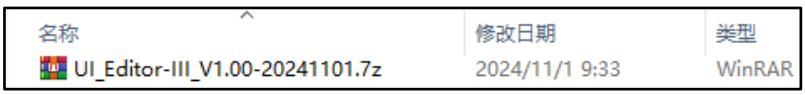


Figure 2-3：UI_Editor-III tool kits

2.2. Unzip the toll kits

Unzip the tool kits to retrieve the software folder, as shown in Figure 2-4:

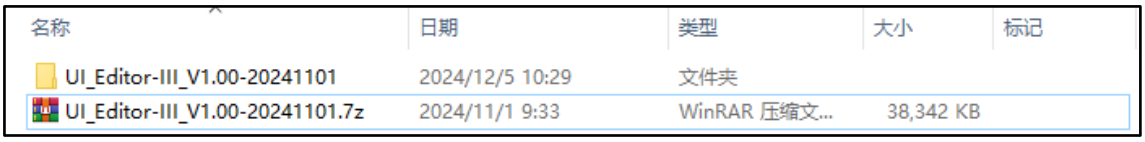


Figure 2-4: Unzip the tool kits

2.3. UI_Editor-III Tool Kits

The contents of the unzipped file folder are as shown below:

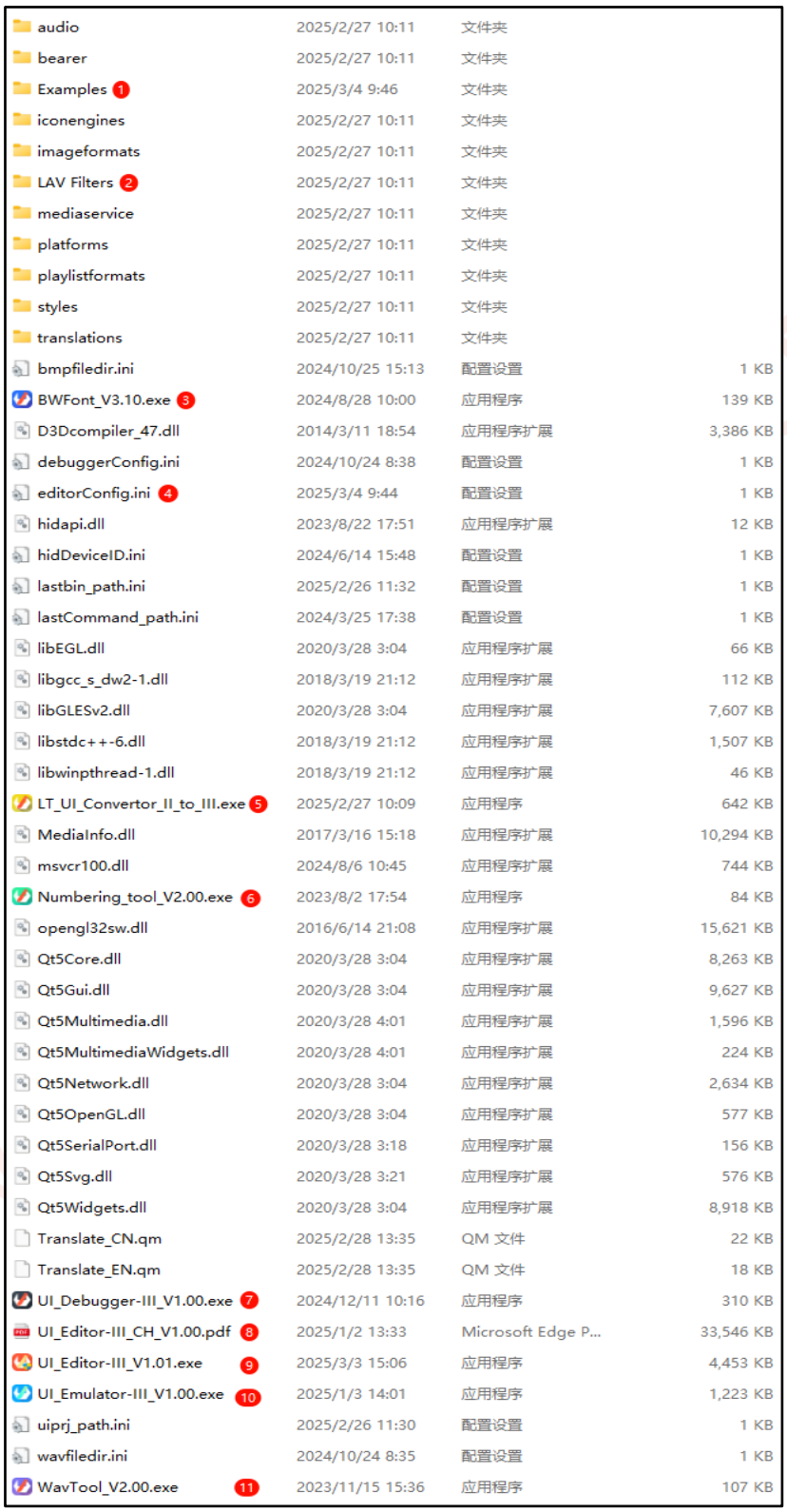


Figure 2-5: UI_Editor-III File Folder

1. **Example Folder**：A demo project is available in this folder.
2. **LAV Filters**：A video decoder tool used by UI_Emulator-II is stored in this folder.
3. **BWFont_Vx.x.exe**：A tool for customized Fonts, refer to *Font Tool*
4. **editorConfig.ini**：Configuration file of UI-Editor.
5. **LT_UI_Convertor_II_to_III.exe**：Used to convert UI_Ediotr-II projects to UI_Editor-III format，refer to *Convert a UI_Editor-II Project to a UI_Editor-III Project*
6. **Numbering_Vx.x.exe**：A tool for numbering the material files, refer to *Numbering Tool*
7. **UI_Debugger-III_Vx.xx.exe**：A tool for testing and monitoring the communication between PC and the UartTFT controller, refer to *UI_Debugger-III* and *Uart Communication*
8. **UI_Editor-III_EN_Vx.xx.pdf**：User manual.
9. **UI_Editor-III_Vx.xx.exe**：Main program. Please right click on it and select [Run as administrator]
10. **UI_Emulator-III**：An emulator for the projects created by UI_Editor-III, refer to *UI_Emulator-III*
11. **WavTool_Vxx.exe**：A tool for converting Wav files to bin files, refer to *WavTool*

2.4. Activate UI_Editor-III

Locate UI_Editor-III_Vx.xx.exe in the file folder:

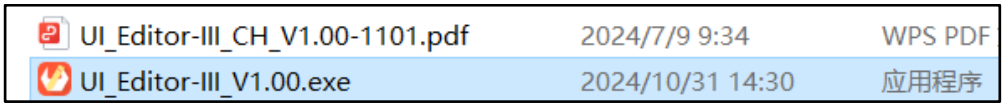


Figure 2-6: Locate UI_Editor-III

Right click on **UI_Editor-III_Vx.xx.exe,** and select [Run as administrator].
Preferred OS: Win10 or above

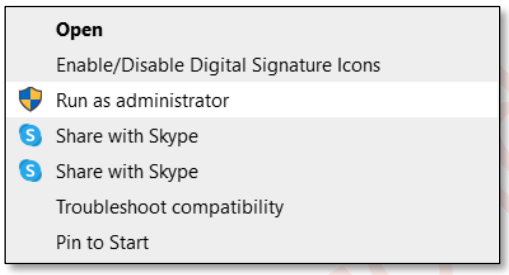


Figure 2-7: Select [Run as administrator]

2.5. Scaling Problem

After starting UI_Editor-III, click on [File], and then click on [Project Setting] to open the setup page:

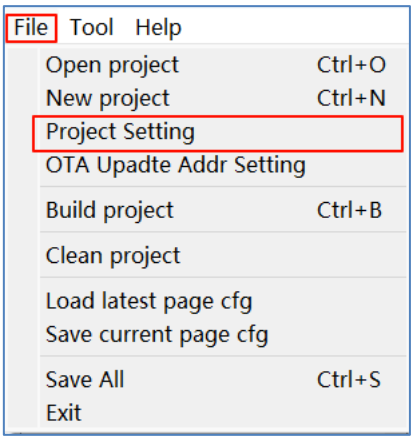


Figure 2-8: Click on [Project Setting]

Check if all the characters are displayed correctly. If characters are displayed incompletely (as shown below), please refer to *Display Scaling* for proper solutions.

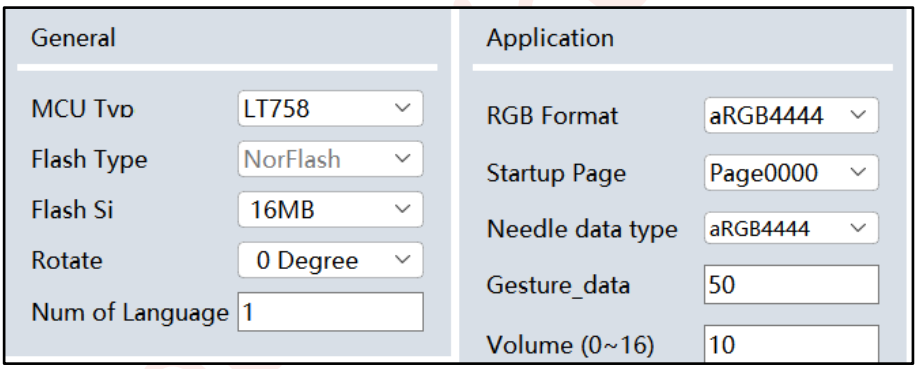


Figure 2-9: Scaling Problem

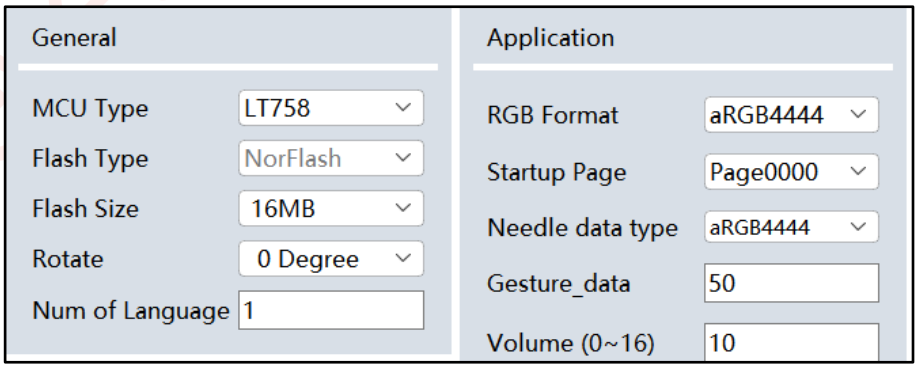


Figure 2-10: Normal Display Example

2.6. Create Shortcut

Right click on UI_Editor-III_Vx.xx.exe, and select [Create Shortcut]

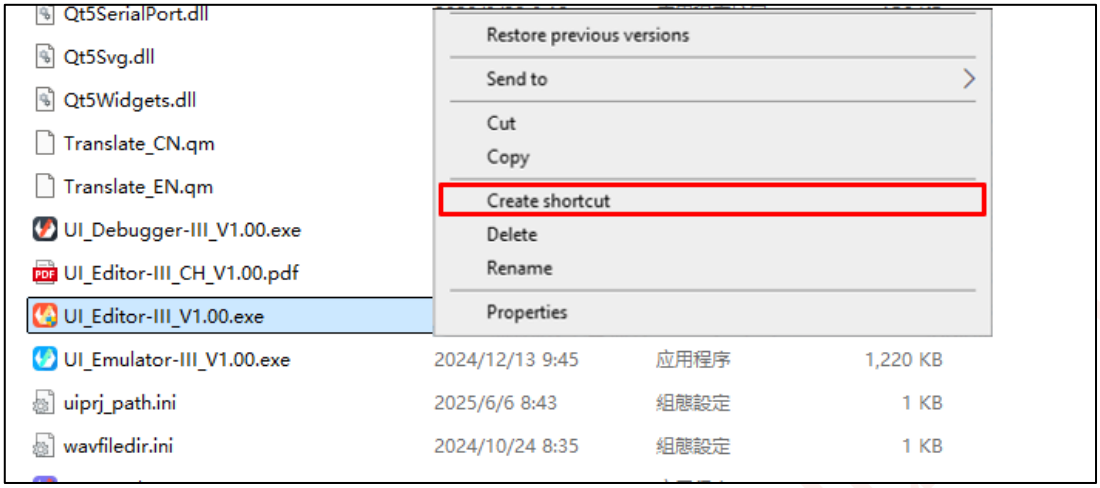


Figure 2-11：Create Shortcut

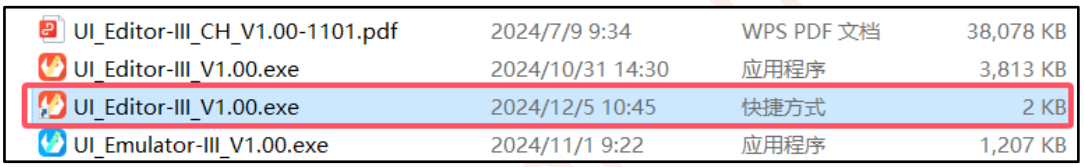


Figure 2-12：Shortcut Created

Right click on the shortcut, select [Copy] or [Cut]

Figure 2-13：[Copy] or [Cut] the Shortcut

Go to Desktop, and paste the shortcut to it.

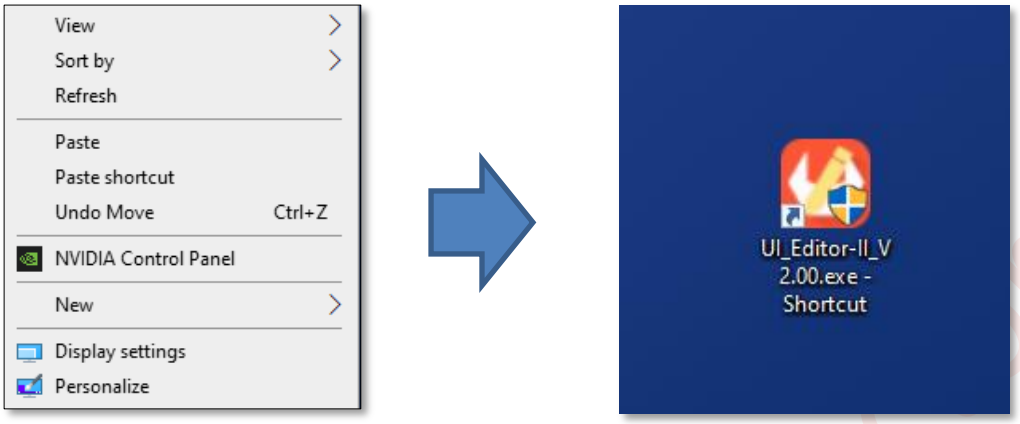


Figure 2-14: Paste the shortcut to Desktop

3. UI_Editor-III Menu & Operation

3.1. Main Screen

Figure 3-1 shows the main screen of UI_Editor-III.

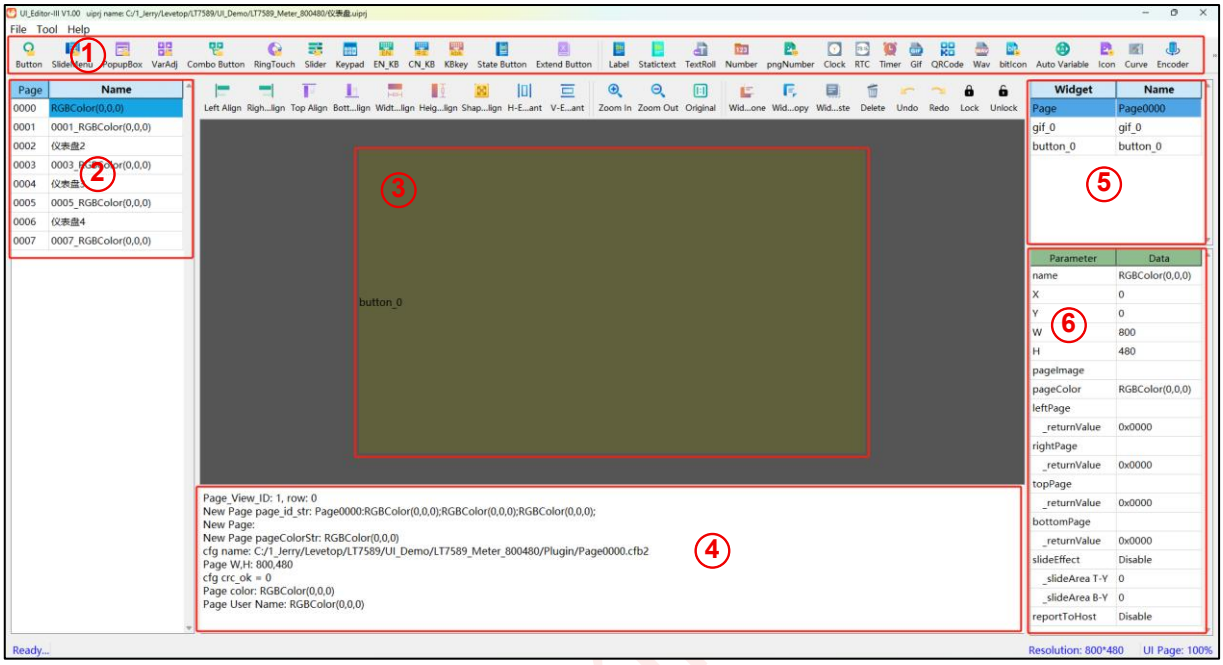


Figure 3-1: Main Screen

1. Tool Bar:

1. As shown in Figure 3-1, ① , developers may click on the icons to add various widgets, such as button, picture, text, and more. Hover the mouse cursor on an icon, the name of the icon will pop-up. Left click on an icon, the mouse cursor will then be switched to Cross style. Developers may then start to add the designated widget to the editing area, and drag it to adjust its width and height. Widgets may be added continuously as long as the mouse cursor remains Cross style. Right click the mouse on the editing area to exit the selection mode, and the mouse cursor will be switched back to Arrow style.

   The tool bar can be classified into 4 parts, as illustrated in Figure 3-2:

   ❶ Widgets with touch function

   ❷ Widgets with display function

   ❸ Widgets for layout and alignment

   Refer to *Widget* for more detail about widgets

Figure 3-2: Tool Bar

2. Page ID and Name List

As shown in Figure 3-1, ②, the left column represents Page ID (unchangeable), and the right column represents the name of the page (user definable). Refer to *Page Operation* for more detail.

3. Page Editing Window

As shown in Figure 3-1, ③, developers may edit (e.g. adding widgets) within the base map.

4. Status Window

As shown in Figure 3-1, ④, every operation process will be listed here in a timely manner. Developers may check the processed results in the status window when making bin files.

5. Widget List

As shown in Figure 3-1, ⑤, this area lists all the available widgets in the designated page. Click the listed name to quickly locate the desired widget in the page editing window

6. Widget – Parameter Setting Window

As shown in Figure 3-1, ⑥, parameters for selected widget can be setup here, including but not limited to name, address, and coordinates etc.

3.2. Function Menus

As shown below，there are three function menus: File, Tool, and Help

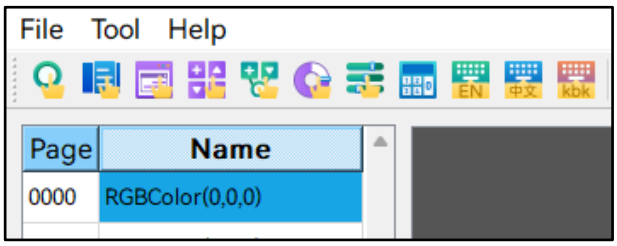


Figure 3-3：Function Menus

3.2.1. File

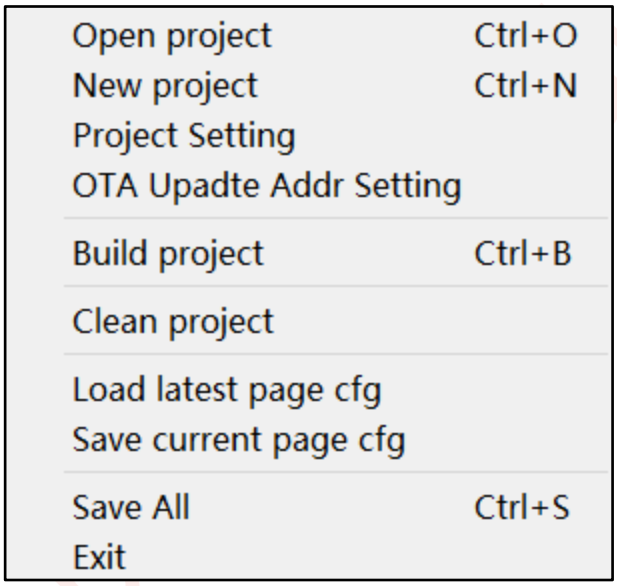


Figure 3-4：File

1. **Open project**：Open an existed project
2. **New project：** Create a new project
3. **Project Setting**：Refer to *Project Setting*
4. **OTA Upadte Addr Setting：** Set the OTA Update address. Refer to *Appendix 7 – OTA Difference Upgrading*
5. **Build project**：Compile the current project and export the UartTFT-V3_Flash.bin
6. **Clean project**：Deleted all bin files (except for font and wav bin files)
7. **Load latest page cfg**：Load the latest cfg file
8. **Save current page cfg**：Save the parameters to a cfg file
9. **Save all**：Save all changes
10. **Exit**：Exit the program

Note: A cfg file records the final configuration of a page. Each page has one and only one cfg file. A cfg file will be saved/updated when

1. [Build project] is clicked（all pages）
2. [Save current page cfg] is clicked
3. [Save all] is clicked
4. users choose to save the changes before exit the project.

3.2.2. Tool

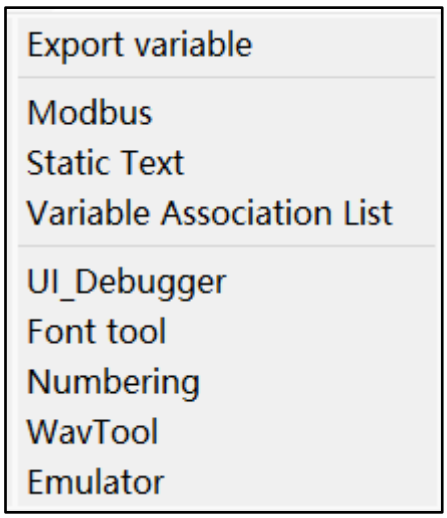


Figure 3-5: Tool

1. Export variable: Export two variable tables, DisplayWidget.csv and TouchWidget.csv, in csv format.

(1) DisplayWidget.csv: A table that lists the parameters of display widgets. Its contents include the address, length, ID, and name of the widgets.

(2) TouchWidget.csv: A table that lists the parameters of touch widgets. Its contents include the address, length, ID, name, and other key parameters.

Figure 3-6: Variable Tables

2. Modbus: Click to open Modbus command table, Refer to *ModBus*

3. Static Text: Click to configure Static Text widgets. Refer to *Setting Static_Text*

4. UI_Debugger: Click to open the debugging tool. Refer to *UI_Debugger-III*

5. Font tool: Click to open the font tool. Refer to *Font Tool*

6. Numbering: Click to open the file numbering tool. Refer to *Numbering Tool*

7. WavTool: Click to open wav tool. Refer to *WavTool*

8. Emulator: Click to open emulator tool. Refer to *UI_Emulator-III*

9. Variable Association List: Click to setup the Variable Association List. Refer to section 3.2.2.1 for further explanation.

3.2.2.1. Variable Association List

This function is used to assign values to multiple variable addresses. Developers may setup the values of multiple variable addresses through the Variable Association List, as shown in Figure 3-7:

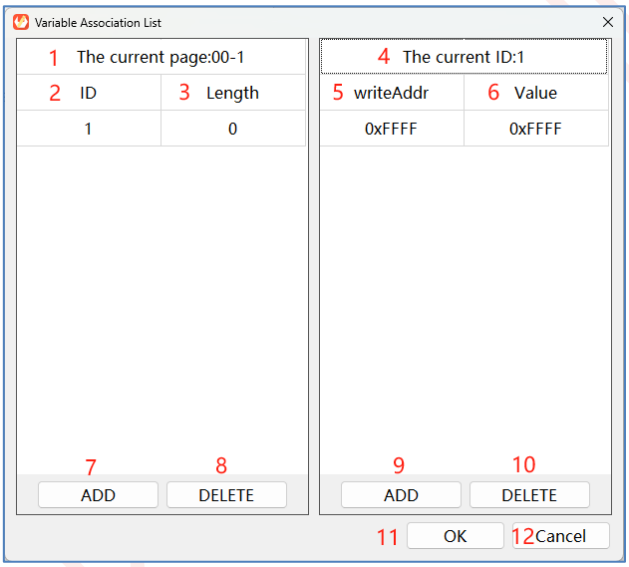


Figure 3-7: Variable Association List

After the Variable Association List is set properly, developers may then enable the settings by assigning the ID number to the specialized register, 0x7014.

1. [The current page]: this column shows the current page number. The settings of the Variable Association List are related to the current page. The settings will only be valid in the current page, and cannot be applied in other pages.
2. [ID]: this column shows the ID number of each variable association list. Each ID number can be linked to unlimited numbers of variables. The ID number is assigned by the software when the [ADD] button (7) is pressed. Note that the ID number can only be deleted from the last one in the list, and one at a time.
3. [Length]: this column shows the number of bytes for the Variable Association List of each ID number. Each variable is count as 4 bytes (2bytes of variable address + 2bytes of data length). Therefore, Length = the number of variables x 4
4. [The current ID]: this column shows the selected ID number. Click the ID number listed on the left table,

the linked variable addresses and their values will be listed on the right table.

5. [writeAddr]: this column shows the variable addresses linked to the selected ID number.
6. [Value]: this column shows the values to be assigned to the variables.
7. [ADD]: click this button to add an ID number.
8. [DELETE]: click this button to delete the last ID number in the list.
9. [ADD]: click this button to add a variable and its value.
10. [DELETE]: click this button to delete the last variable in the list.
11. [OK]: click this button to save the settings.
12. [Cancel]: click this button to cancel the modification.

Note:

The specialized register, 0x7014, is used to enable the settings in the Variable Association List. For example, when an ID number of 0x0003 is assigned to the specialized register, 0x7014, the variables linked to the ID number (0x0003) will be assigned the designated values.

Figure 3-8 shows a practical example. When the GIF widget is done playing, the ID number of 0x0001 will be assigned to the specialized register, 0x7014. Therefore, the variables linked to the ID number (0x0001) will be assigned the designated values.

Parameter	Data
name	gif_0
parameterA...	0xFFFF
writeAddr	0x3037
X	0
Y	0
W	1024
H	600
playOnceCode	11
startCode	10
stopCode	100
disappearCo...	200
playAtStart	Disable
interval(10ms)	12
gifName	0000.gif
dataFormat	
defaultStatus	Run
writeAddr	0x7014 ←
_value	0x0001 ←

Figure 3-8: GIF Widget Parameter List

3.2.3. Help

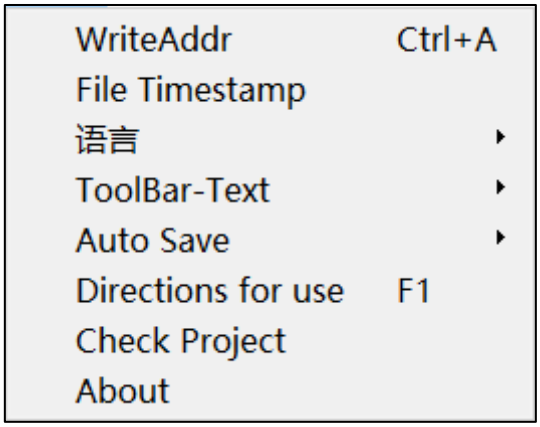


Figure 3-9：Help

1. WriteAddr:

(1) Paste Auto Address

Checked: the start value of writeAddr will be applied to the next pasted widget automatically.

Unchecked: the writeAddr parameter of the new copied widgets will be the same as the original one.

(2) New WriteAddr: The start value of writeAddr for the next added widget.

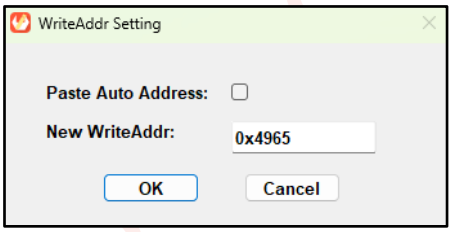


Figure 3-10：writeAddr

2. File Timestamp:

(1) If checked, the current time information will be saved in the UartTFT-II_Flash.bin when the project is compiled.

(2) If unchecked, the user-defined time information will be saved in the UartTFT-II_Flash.bin when the project is compiled.

Note: The displayed date/time in the pop-up window is the time of the last compilation.

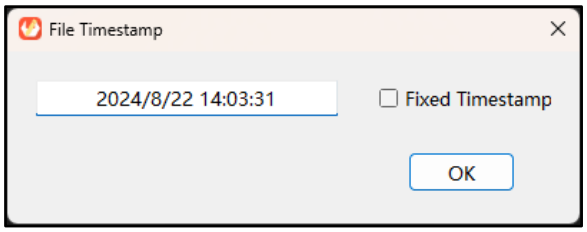


Figure 3-11：File Timestamp

3. **语言:** Options for choosing Chinese and English menu.

4. **ToolBar-Text:** Options for toolbar style (with / without Text)

5. **Auto Save:** When set, software will save the editing result every 5 seconds.

6. **Directions for use:** Open the user manual

7. **Check Project:** Check if the material's file naming follows the rule (refer to *Material Format*), and the used widgets are within the display range.

8. **About:** Software information

4. Create a New Project

4.1. Materials Preparation

4.1.1. About File Folders

After clicking on [New project] in the File menu, the default file folders will be automatically created as the file folders shown in Figure 4-1. The name of each file folder is specified by UI_Editor-III and should not be changed.

If developers would like to create a new project with existed material folders, refer to *Using an Existed Project to Create a New Project*

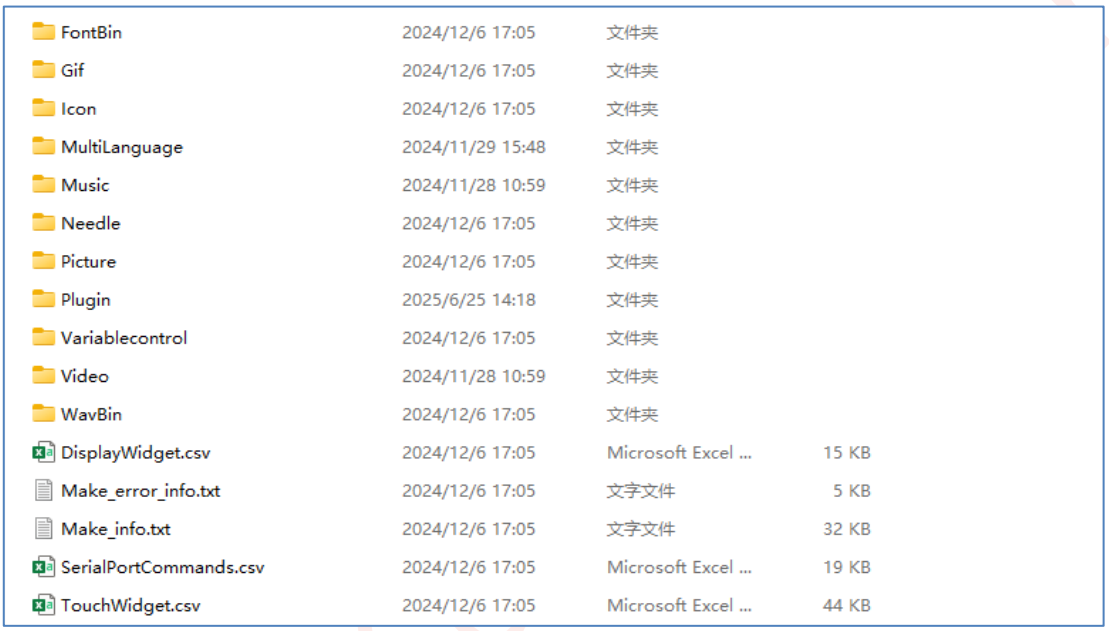


Figure 4-1: File Folders

4.1.2. Material Format

4.1.2.1. Picture

Contents: Page picture, Popup box pictures, Keyboard pictures

Format: BMP, JPG

Naming: Number the pictures by 0000 ~ 9999, and name them as “xxxx” or “xxxx_user defined”, as shown in the figure below:

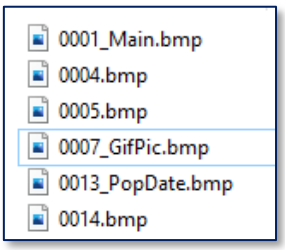


Figure 4-2: Name a Picture

Note:

1. Each number can only be used once.
2. If the maximum picture number is 0010, and there are only 6 pictures in the folder, then UI_Editor-III will still add blank pages to the project and make it total 11 pages (0000~0010), following the order of the numbers. Developers may manually add pictures to those pages afterwards.
3. The number of pages of a new created project is based on the maximum number of the picture name. Users may also add new pages by right-clicking on page column in UI_Editor-III, and click on [NewPage].
4. PNG pictures cannot be used as page pictures. Page pictures must be JPG or BMP format.
5. Developers may utilize a numbering tool, Numbering_tool_Vx.xx.exe, provided by Levetop to quickly number the pictures. Refer to *Numbering Tool*

4.1.2.2. Icon

Contents: Icons, Graphic Number Display, Slide Menu, Slider Bar, Progress Bar, and Analog Clock etc.

Format: BMP, JPG, PNG

Naming: Number the materials by 0000 ~ 9999, and name them as "**xxxx**" or "**xxxx_user defined**", as shown in the figure below:

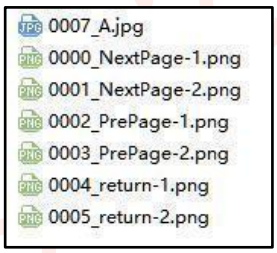


Figure 4-3: Name an Icon

Note: For setting the width and height of Icons in the same group, refer to *Icon Width & Height*

4.1.2.3. FontBin

Contents: Font bin

Format: bin

Naming: Number the FontBin by 00 ~ 35, and name them as "**xx_Font-user defined**", as shown in the figure below:

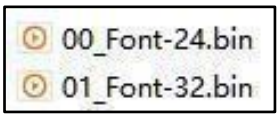


Figure 4-4: Name a Font

Note:

Developers may utilize a font tool, BWFont_Vx.xx.exe, provided by Levetop to make customized font libraries. Refer to *Font Tool*

4.1.2.4. Gif

The Gif widget supports two different materials, one is in GIF format, another is in JPG format. When selecting the material of a Gif widget, a popup box will be shown as Figure 4-5. Click [GIF] to select a Gif file; click [JPG] to select a folder with a group of JPG pictures.

Figure 4-5: Select GIF type

Contents: Gif or folders with JPG pictures

Format: Gif, JPG

Naming: Number the Gif files and the JPG picture folders by 0000 ~ 9999, and name them as "**xxxx**" or "**xxxx_user defined**", as shown in the figure below: A ".gif " suffix must be added to a Gif file; and a " .jpg " suffix must be added to a JPG picture. No capital letters are allowed for the suffix. No suffix is required for the name of the JPG picture folders.

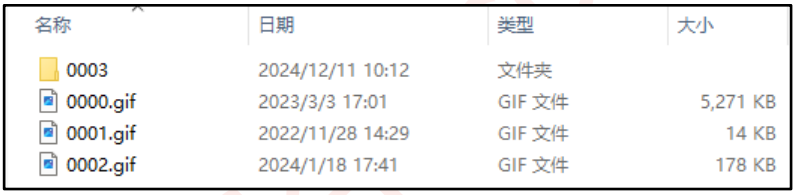


Figure 4-6: Name a Gif

The JPG pictures in the JPG picture folders should also be numbered with a 4-digit number from 0000 to 9999, as shown in the figure below:

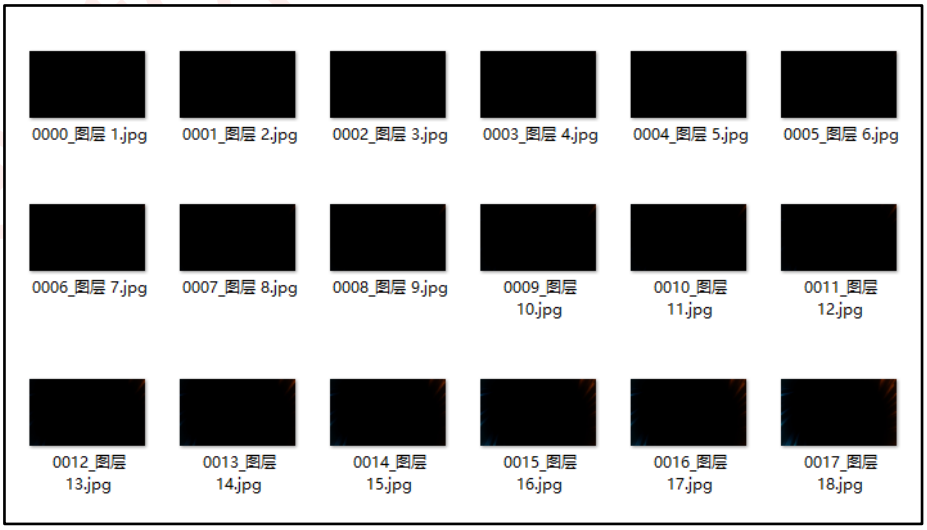


Figure 4-7: Name a JPG

4.1.2.5. WavBin

Contents: Wav bin files

Format: bin

Naming: Number the Wav files by 0000 ~ 0099, and name them as "**00xx_Wav**" or "**00xx_Wav_user defined**", as shown in the figure below:

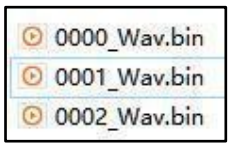


Figure 4-8: Name a Wav

4.1.2.6. Needle

Contents: Picture generated by needle widgets

Note: The contents are generated by UI_Editor automatically.

4.1.2.7. UI_projectname.h

This file will be generated by UI_Editor after the UI project is compiled, and it lists the macro definition and the address of each widget used in every page.

Note:

1. The file will be named after the project name.
2. The name of each macro definition is consisted of [widget type + Page ID + widget default name]. The name of each macro definition may also include the user-defined name, that is, [widget type + Page ID + widget default name + user-defined name]. To do so, the editorConfig.ini (refer to *UI_Editor-III Installation*) must be modified. As the figure shown below, the [Header_file_index] value has to be changed to 1.

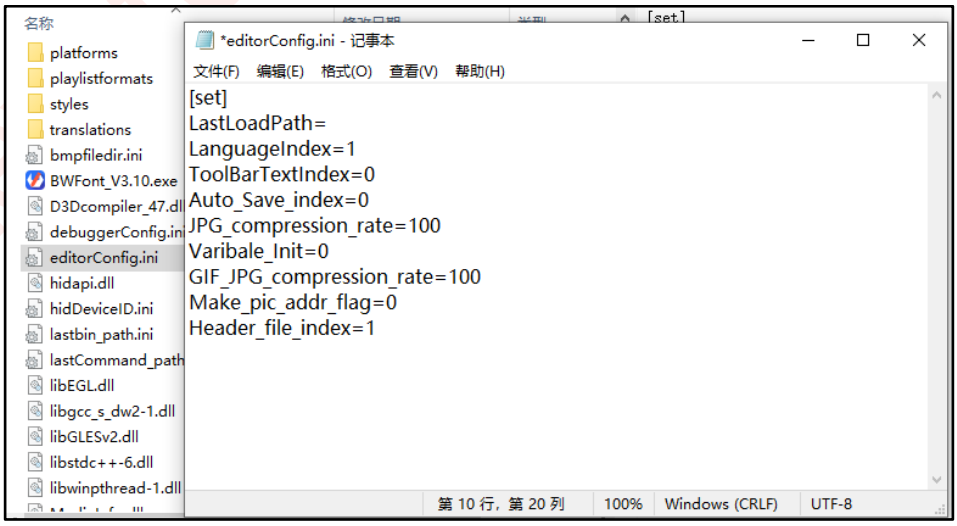


Figure 4-9: editorConfig.ini

3. UI_projectname.h can be applied to the source code of UartTFT controllers. Developers may include this file in

the [user] folder of the source code for further application.

4. If the [user-defined name] is to be used in the UI_projectname.h, then the user-defined name must be in English.

4.1.2.8. Others

1. Plugin folder is for storing files of compiled information.

2. In a folder, each material must be named differently. For example, 0000.png and 0000.bmp may be allowed in the computer's file management system, but it will confuse UI_Editor-III and result in failure phenomena.

3. All of the materials, widgets, pages, and projects cannot be named after any of the following symbols:

\ / : * ? “ < > | . ,

Also, only one period symbol is allowed, used in between the file name and the suffix.

4、Levetop offers a public material library, refer to *Material Library*

4.1.2.9. Material Location

Materials should be organized and stored in the folders as listed in below table. Only the used materials will be exported to the UartTFT-V3_Flash.bin. However, the wav bin files will always be exported to UartTFT-V3_Flash.bin, no matter they are used or not. Refer to *UartTFT-V3_Flash.bin* for more detail.

Table 4-1: Material Location

Material Folders & Contents						
FontBin	**Gif**	**Icon**	**Picture**	**Plugin**	**WavBin**	**Needle**
Font	Gif	Icon	Base map	Compiled files	Wav files	Auto-generated by UI_Editor
		Slide Menu	Keyboard			
		Slider Bar	Popup box			
		Circular Touch				
		Analog Clock				
		Graphics Number Display				

4.2. Project Setting

The parameters in [Project Setting] page, as shown below, need to be properly set before creating a new project:

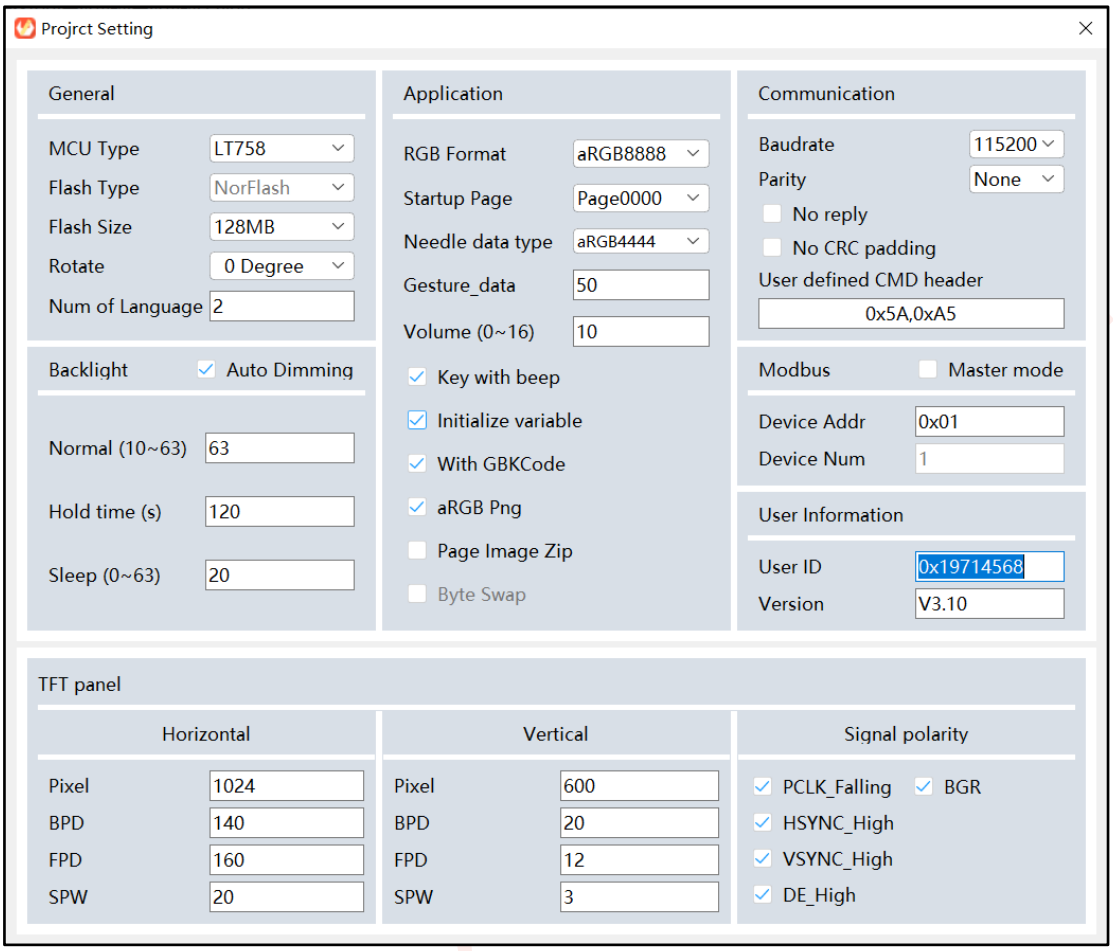


Figure 4-10：Project Setting

The definition of each parameter is described as below:

General	**MCU Type**	:	Select MCU models
	Flash Type	:	Select SPI Flash types
	Flash Size	:	SPI Flash Size. Set by actual flash size.
	Rotate	:	Rotation angle. Refer to *Screen Rotation*
	Num of Language	:	Set the number of languages used.

Backlight Control	**Auto Dimming**	:	Checked → Auto dimming control; Unchecked → Always on
	Normal (10~63)	:	Brightness Level, adjustable from 10 to 63
	Hold time (s)	:	Dim the backlight if no operation in set period. Range: 1 to 65535, unit: second
	Sleep (0~63)	:	Dimming level, Range: 0 to 63. Touching the panel again can turn on the backlight
Application	**RGB Format**	:	Picture data format
	Startup Page	:	Boot screen. The first picture/animation shown right after power-on
	Needle Data Type	:	Choose to compress Needle files or not. LT7589 does not support this function.
	Gesture Data	:	Minimum sliding effective distance in pixel. When sliding to switch the display page, if the sliding distance exceeds the set value, then the sliding action will be effective.
	Volume (0~16)	:	Initialize the volume. 16 means the maximum volume
	Initialize Variable	:	Enable the default value of the widgets if checked.
	With GBKCode	:	Add GBK code to UartTFT-V3_Flash.bin. Must be checked if using GBK font.
	Key with beep	:	Enable buzzer. If checked, the buzzer will beep when the panel is touched.
	aRGB Png	:	Checked → Hardware PNG (αRGB4444-16bits); Unchecked → Software PNG
Communication	**Baudrate**	:	Data transmitting speed, bit per second
	Parity	:	Three options, [None], [Odd], and [Even]
	No reply	:	No returned messages during communication if checked.
	No CRC padding	:	CRC will not be used if checked.
	User defined CMD header	:	Using user-defined start bytes as the header for Uart communication.
Modbus	**Master Mode**	:	Check the box to set the project as Modbus Master, uncheck the box to set the project as Modbus Slave. Either one requires customized MCU_Code.
	Device Addr	:	Set the slave address here, when UartTFT controller is used as Modbus Slave.
	Device Num	:	Reserved.
User Information	**User ID**	:	No modification required.
	Version	:	Software version.

For TFT panel:

TFT Horizontal	**X-Pixel**	:	TFT panel resolution, X direction
	HBPD	:	Based on TFT panel spec
	HFPD	:	Based on TFT panel spec
	HSPW	:	Based on TFT panel spec
TFT Vertical	**Y-Pixel**	:	TFT panel resolution, Y direction
	VBPD	:	Based on TFT panel spec
	VFPD	:	Based on TFT panel spec
	VSPW	:	Based on TFT panel spec
Signal Polarity	**PCLK_Rising**	:	Based on TFT panel spec
	HSYNC_Low	:	Based on TFT panel spec
	VSYNC_Low	:	Based on TFT panel spec
	DE_Low	:	Based on TFT panel spec
	BGR	:	Checked → BGR; Unchecked → RGB

4.3. Create a New Project – Procedure

Click on [File] menu, and then click on [New project] to create a new project, as shown below. It is suggested that developers create independent folders for new projects.

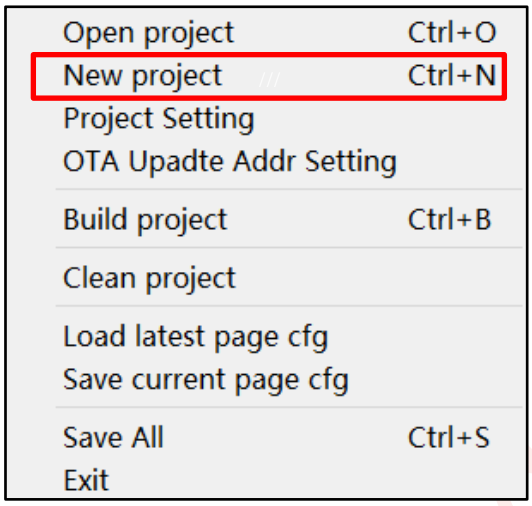


Figure 4-11: Create a New Project

Step 1: Create the file folders as described in *Create a New Project*, and store the needed materials to the designated folders.

Step 2: Activate UI_Editor-II, click on [Project Setting] and setup each parameters properly as described in *Project Setting*. Click on [New Project] when the settings are done.

Step 3: Locate the pre-created folder, enter the new project name in the pop-up window, and then click on [Save], as shown below:

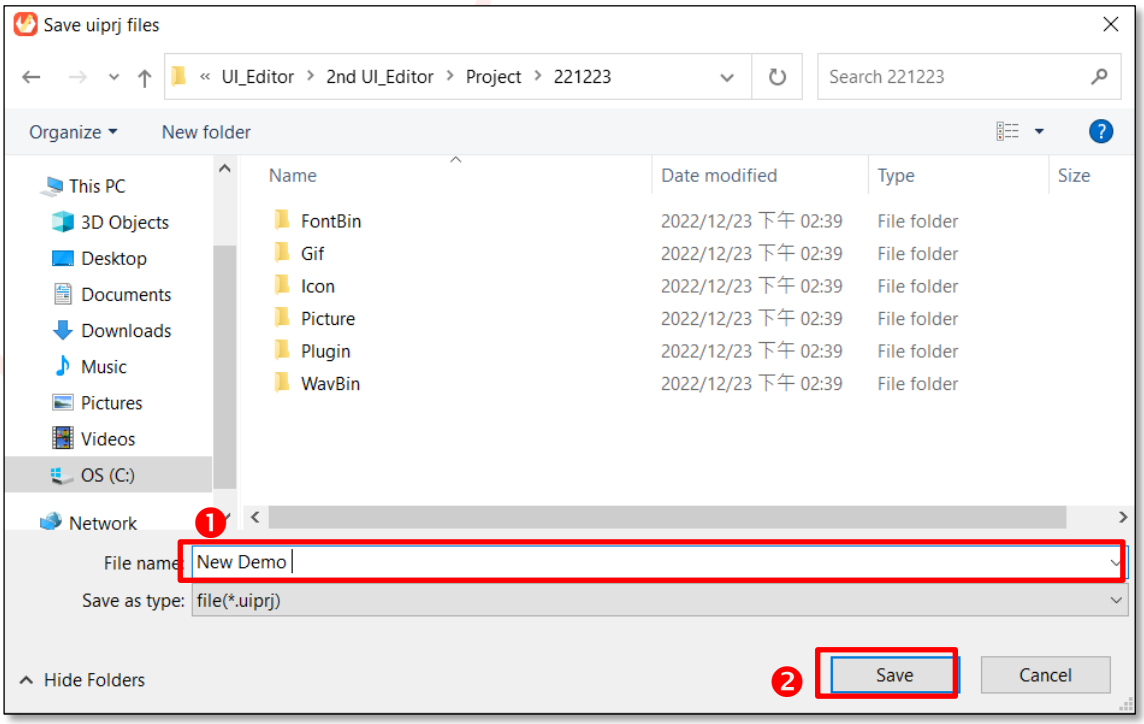


Figure 4-12: Enter the Project Name and Save it

Step 4: After clicking [Save] button, a new pop-up window will show up as below. Choose one of the pictures, and then click on [Open]. If there is no picture available at the time, simply click [Cancel].

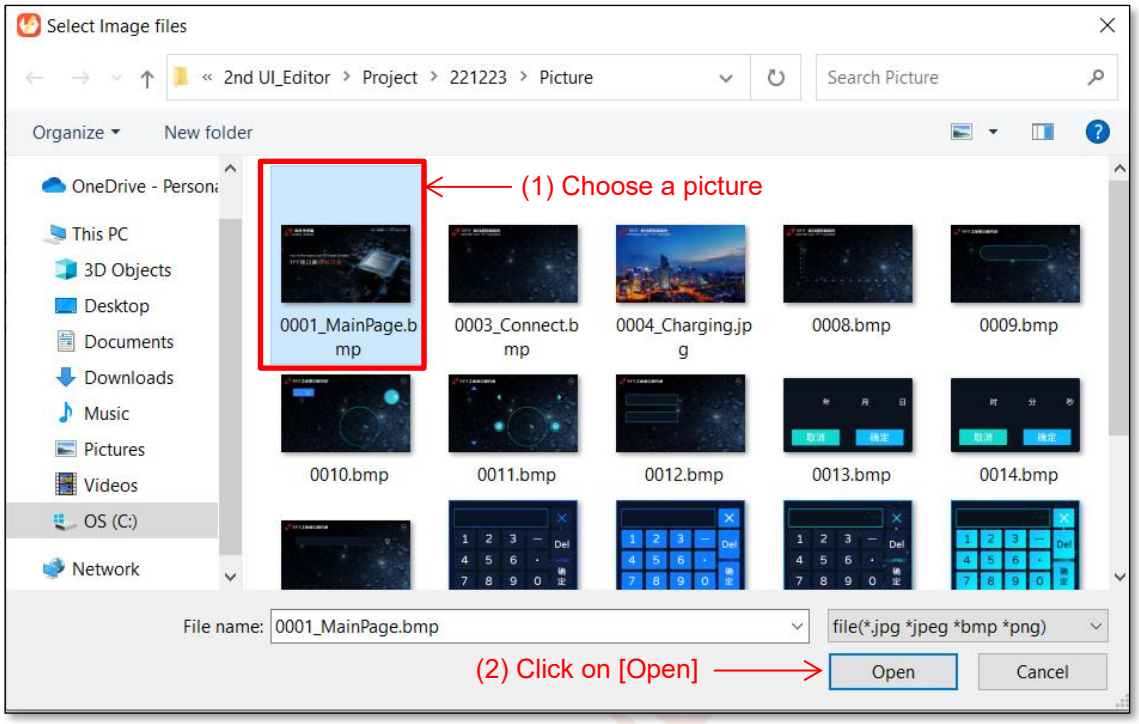


Figure 4-13: Choose a Picture

Step 5: Click on [UI Page] to view and edit the contents.

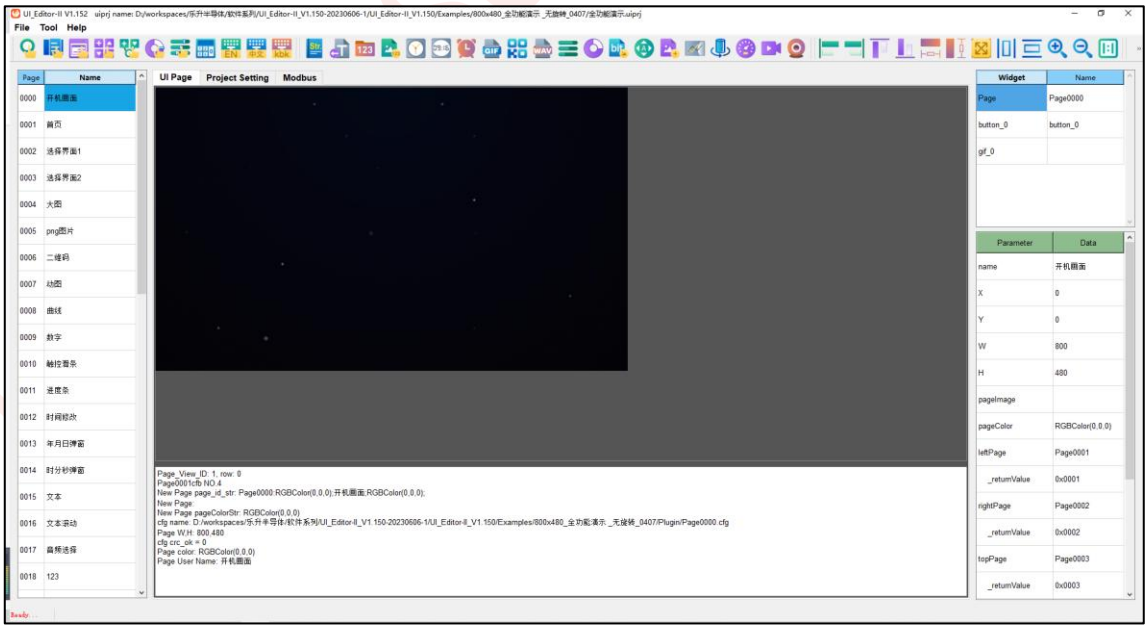


Figure 4-14: UI Page View

5. Page Operation

5.1. Page Operation and Parameters

As shown in Figure 5-1, ❷, this area lists all the page parameters. To review certain page's parameters, developers may (1) select from the page list, as shown in Figure 5-1, ❶; (2) click on the editing area (not on any widgets), as shown in Figure 5-1, ❸.

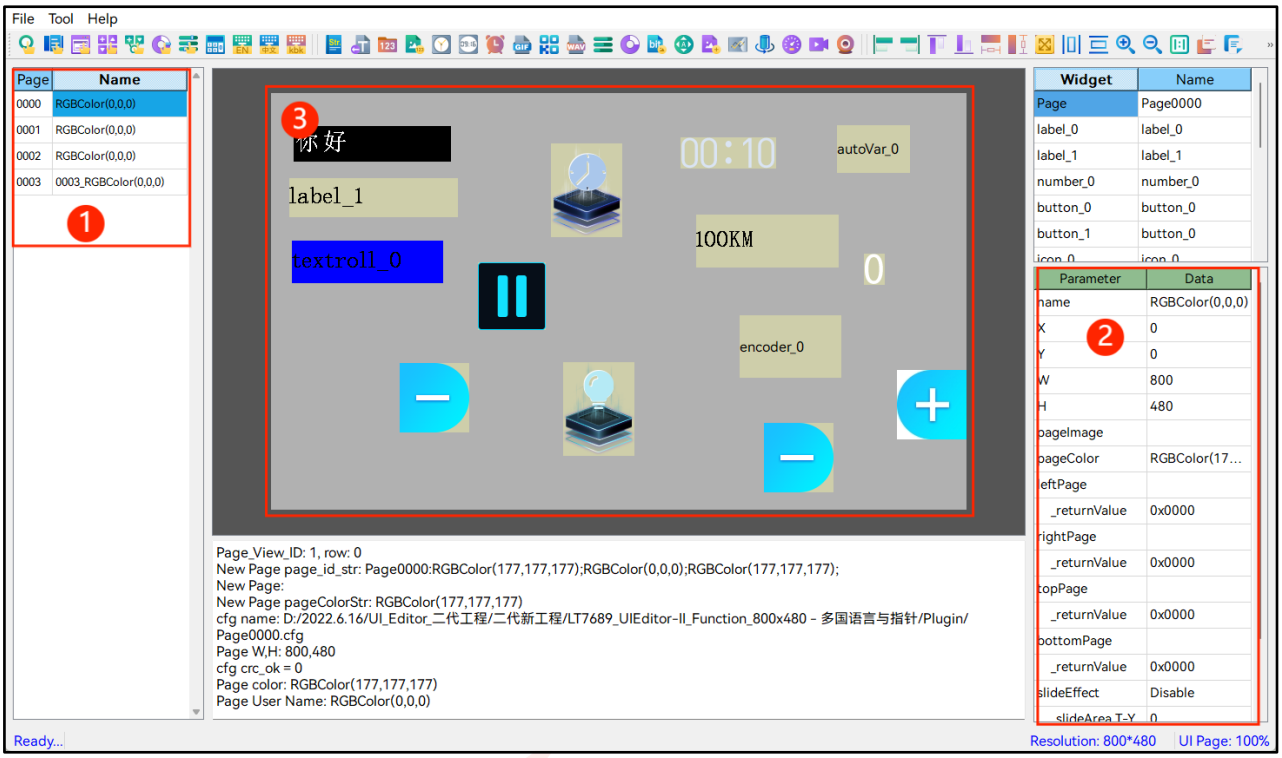


Figure 5-1: Check Page Parameters

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

Figure 5-2: Page Parameter List

name : Page name, user-definable. Default is the original file name when creating a new project.

X and Y : Default values are 0 for both parameters, no need to modify.

W and H : Width and Height of the page, no need to modify. If the background picture is changed, these two parameters will be auto adjusted.

pageImage : Double click to switch to other background pictures in the materials' folder.

pageColor : The color of the page when there is no background picture. When the PageImage is empty, this parameter will be effective. Double click it to select a color.

leftPage : The designated page to jump to, when a slide-to-left touch operation occurs.

_returnValue : The designated value to report to the host when a slide-to-left touch operation occurs.

rightPage : The designated page to jump to, when a slide-to-right touch operation occurs.

_returnValue : The designated value to report to the host when a slide-to-right touch operation occurs.

topPage : The designated page to jump to, when a slide-to-top touch operation is happened.

_returnValue : The designated value to report to the host when a slide-to-top touch operation occurs

bottomPage : The designated page to jump to, when a slide-to-bottom touch operation occurs.

_returnValue : The designated value to report to the host when a slide-to-bottom touch operation occurs.

slideEffect : Enable the slide operation effect, refer to *Slide to jump - with sliding effects*

_slideArea T-Y : The Y coordinate of the top edge of the sliding area. The reference point is the left-

top coordinate (0, 0)

_slideArea B-Y : The Y coordinate of the bottom edge of the sliding area. The reference point is the left-top coordinate (0, 0)

reportToHost : If set to Enable, the UartTFT controller will return a fixed address (0xFFFF) and the designated returnValue to the host when a sliding operation occurs. Refer to *Touch Returned Message*

Note: Only the pages whose parameters (leftPage, rightPage, topPage, and bottomPage) are properly set, can they support the sliding operations. In addition, the below conditions must be satisfied:

0 <= _slideArea T-Y < _slideArea B-Y <= Panel Y resolution

5.2. Slide to Jump

Developers may utilize below methods to implement page jumps:

Type I: Page jump by UI controls

1. Page jump by Button widgets, refer to *Button*
2. Page jump by Combo Button widgets, refer to *Combo Button*

Type II: Page jump by sliding gesture

1. Slide to jump, without sliding effects
2. Slide to jump, with sliding effects

Type III: Page jump by Uart command

1. Assign the destination page number to Register 0x7000, refer to *Page Register - 0x7000*

5.2.1. Slide to jump – without sliding effects

Setting [slideEffect] to “Disable” will skip sliding effects. The page will not move when the finger slides on the panel. When the sliding gesture triggers a page jump action, the new page will be shown at once.

As shown in Figure 5-3, when sliding to the left, page0001 will be shown up, and a value of 0x0001 will be reported to the host; when sliding to the right, page0002 will be shown up, and a value of 0x0002 will be reported to the host.

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

Figure 5-3: Slide to jump – without sliding effects

5.2.2. Slide to jump – with sliding effects

Developers may enable the sliding effects by setting [slideEffect] to “Enable”. Only two sliding gestures are supported – [sliding to the left] and [sliding to the right]. The designated area (set by _slideArea T-Y and _slideArea B-Y) will move as the finger on the panel moves. Once the finger touch is released, the page jump will be performed.

slideEffect	Enable
_slideArea T-Y	100
_slideArea B-Y	300

Figure 5-4: Slide to jump – with sliding effects

Based on the settings shown in Figure 5-5, _slideArea T-Y is 100, and _slideArea B-Y is 300. The sliding area is then depicted as the green area shown in Figure 5-5.

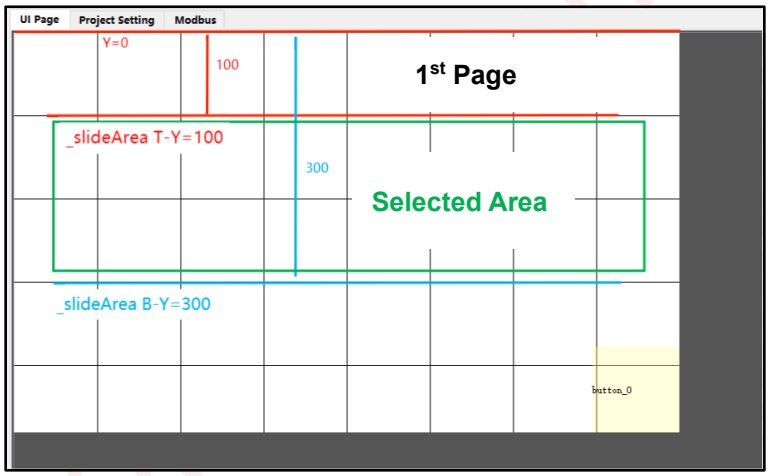


Figure 5-5: Sliding Area

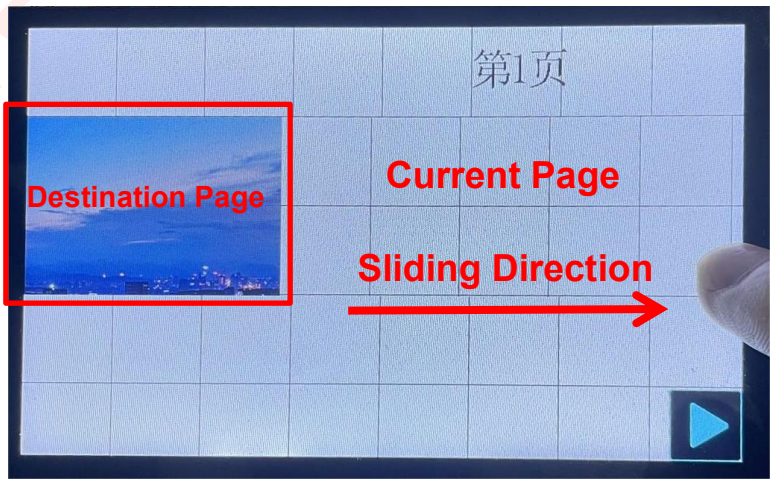


Figure 5-6: Demonstration on Sliding Area

If _slideArea T-Y is set to 0, and _slideArea B-Y is set to 480（Y resolution of the Panel）, then the sliding effects will be like sliding a whole page, as the figure shown below:

Figure 5-7：Sliding the whole page

Note:

1. Sliding area should not have dynamic widgets such as GIF, analogue clock, or digital clock, otherwise it may result in abnormal display.
2. **0 <= _slideArea T-Y < _slideArea B-Y <= Panel resolution of the Y direction**
3. If RGB format is set as aRGB4444, then

   LT7589 does not support the sliding effect on panels with resolution of 1024x600 or above.

   If RGB format is set as aRGB8888, then

   LT7589 does not support the sliding effect on panels with resolution of 800x480 or above.

5.3. New Page

Right click on page list, a pop-up window will be shown as Figure 5-8. Select [NewPage] to add a new page. Every new page will be added to the end of the page list by default. A new created page will have no contents.

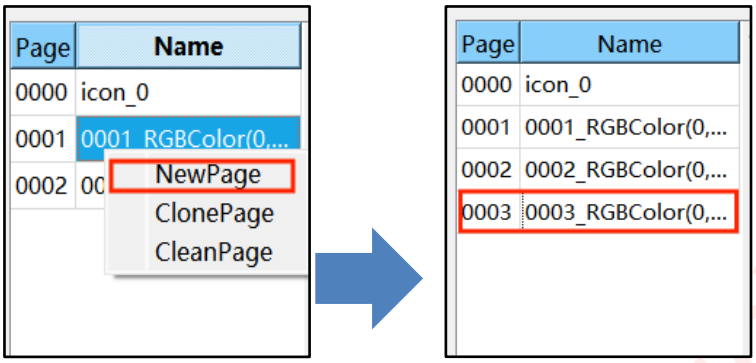


Figure 5-8: Add a New Page

5.4. Clone a Page

Right click on the page you wish to clone in the page list; a pop-up window will be shown as Figure 5-9. Select [ClonePage] to clone the page. A new page will be added to the end of the page list by default, and its contents will be the same as the original one. Developers must avoid address conflicts between widgets.

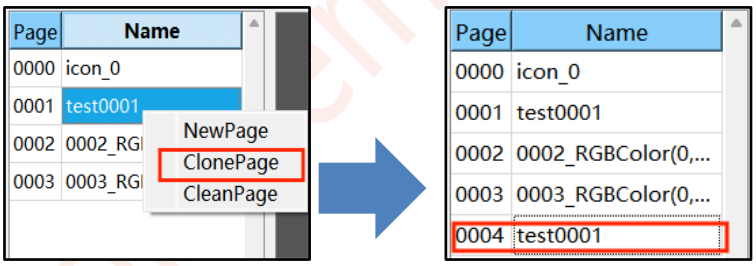


Figure 5-9: Clone a Page

5.5. Clean Page

Right click on the page you wish to clear in the page list; a pop-up window will be shown as Figure 5-10. Select [CleanPage] to clear the page.

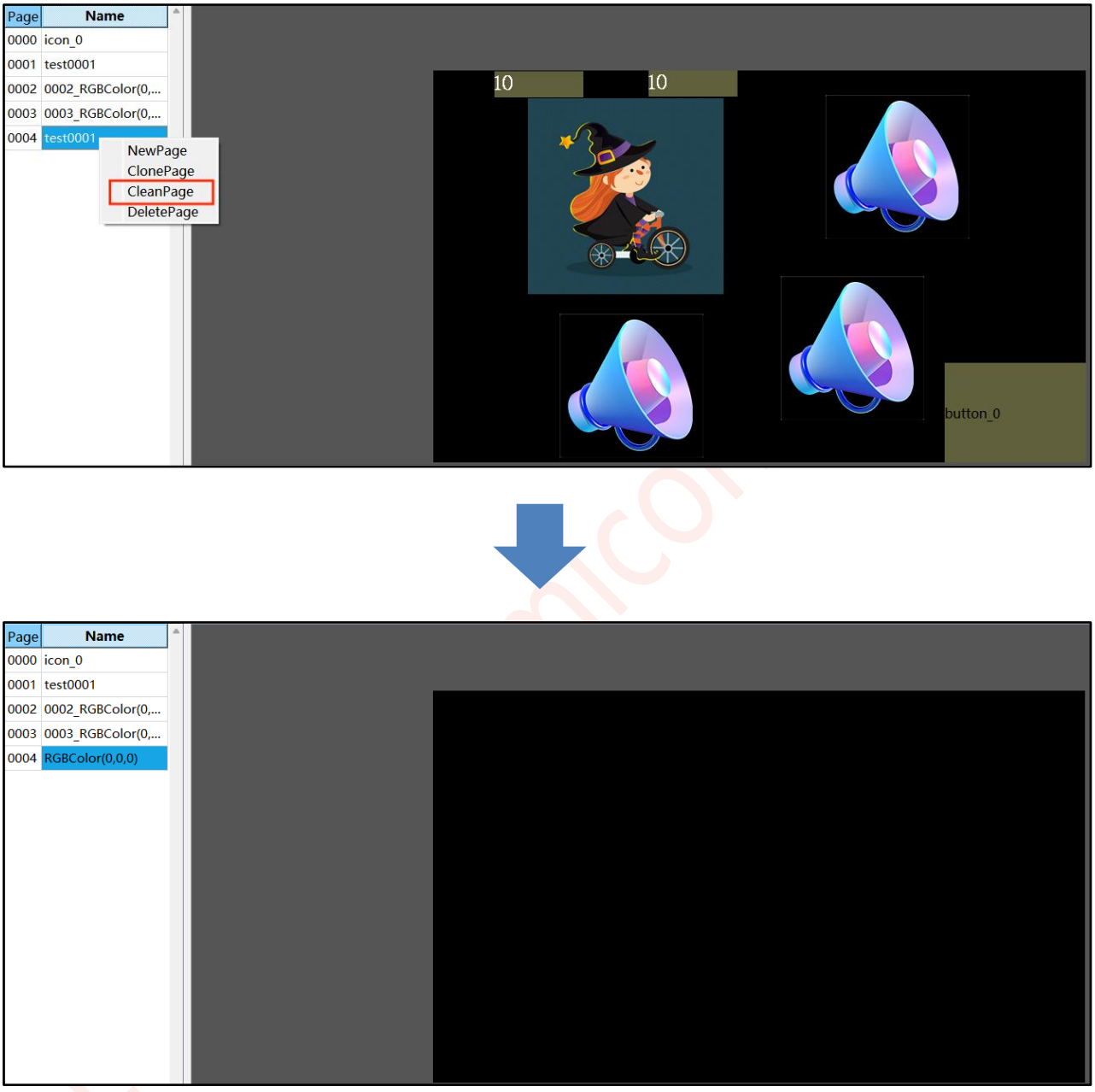


Figure 5-10: Clean Page

5.6. Delete the Last Page

Developers may delete the last page by clicking on [DeletePage] in the pop-up window, as shown in Figure 5-11. However the page is deleted, the cfg file still keeps its contents, therefore, if a new page with the same page ID is created afterwards, the deleted contents will be loaded to the new page.

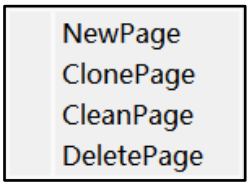


Figure 5-11: Delete the Last Page

5.7. Redundant Page

For the redundant pages (pages that are not used), developers may simply clear up their contents. Empty pages will not occupy Flash space.

5.8. Basic Operation

5.8.1. Add a Widget

Step I: Click on the target widget icon, the cursor will be switched to “Cross” style.

Figure 5-12: Add a Widget

Step II: Click within the editing area, and then drag to form the widget.

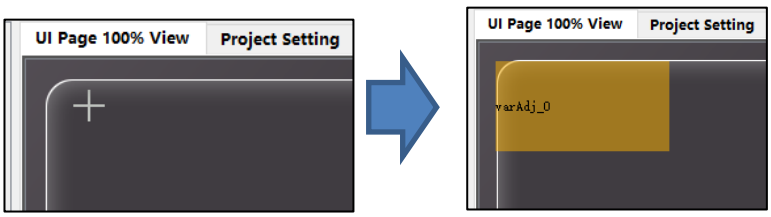


Figure 5-13: Generate a Widget

Step III: Right click on the editing area to exit the Widget Adding mode, the cursor will be switched back to “Arrow” style.

5.8.2. Select Existed Widgets

There are two ways to select existed widgets, (1) click on the target widget; (2) frame selection. When using the frame selection, the whole target widgets should be included. Once the frame selection is done, developers may move the selected widgets together, or copy them and then paste them to other pages.

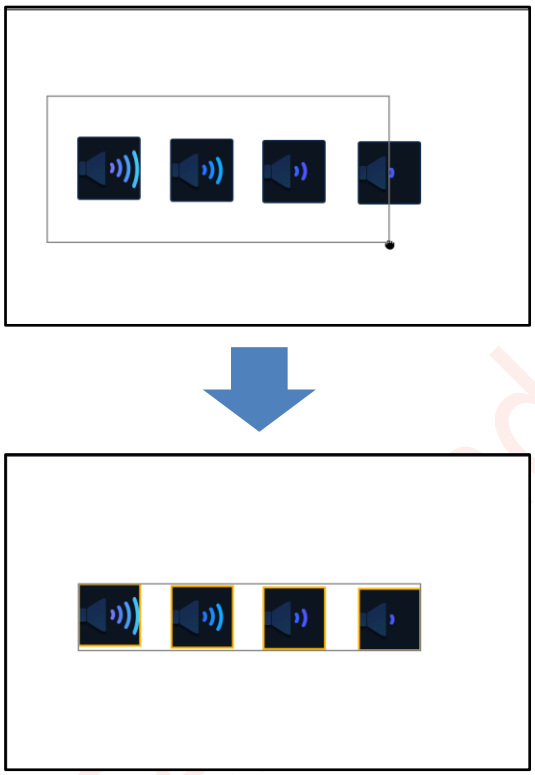


Figure 5-14: Frame Selection

5.8.3. Delete Widgets

ICON: 

Click on the target widget or select multiple widgets through frame selection. Next, click on the ICON shown above or the [Delete] key on the keyboard, then the selected widgets will be deleted.

5.8.4. Widget_Clone

ICON: 

Click on the target widget, and then click on the ICON shown above. A new widget will be generated on the side of the original one. All the parameters of the new widget will be the same as the original one, except for its coordinates.

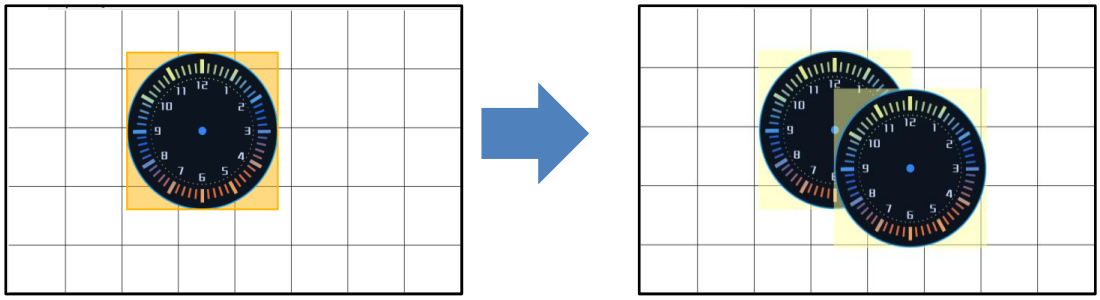


Figure 5-15：Clone a Widget

5.8.5. Widget Copy and Paste

ICON： 

This function is for copying multiple widgets, and the copied widgets can be pasted to different pages. All the parameters will be the same as the original ones except for the addresses. The function also supports short-key: Ctrl + C = copy to clipboard; Ctrl + V = paste to the current page.

Step I: Frame select the target widgets, and then use Ctrl + C or click on the ICON above to copy the contents to the clipboard.

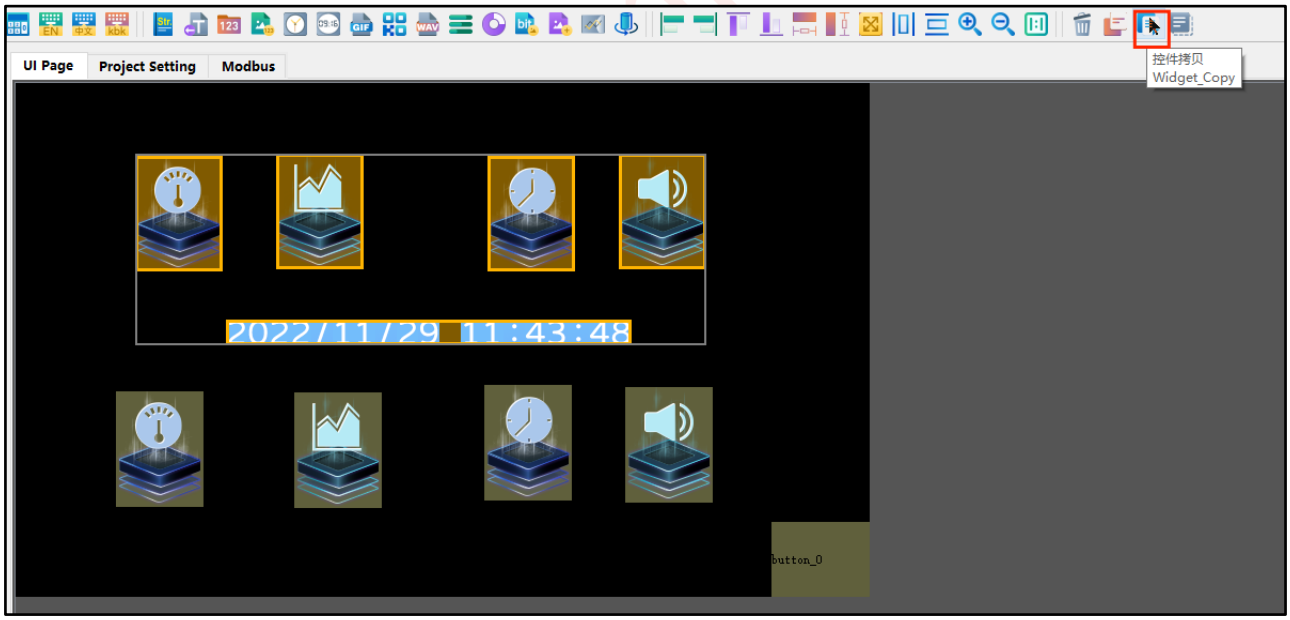


Figure 5-16：Copy Widgets

Step II: Go to the destination page, and then use the short-key, Ctrl + V, to paste the copied widgets. Finally, right click on the editing area to exit the selection mode.

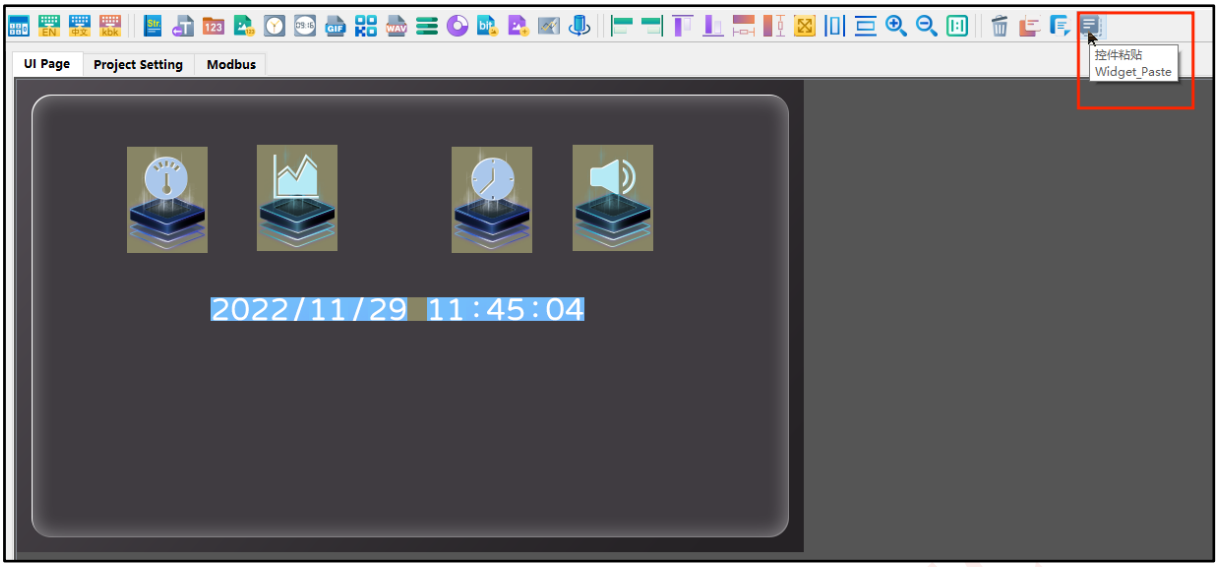


Figure 5-17: Paste the Widgets

5.8.6. Fine-tune Widget Location

Developers are allowed to adjust the location of widgets by directly entering coordinates or using mouse to drag the widgets to a designated location. To fine-tune the location of one or a set of widgets, developers may also utilize the 4 direction keys (← ↑ ↓ →) on the keyboard. Each click will move the selected widgets to the designated direction for 1 pixel. These direction keys also support long-press operation.

Note: When the base map is zoom-in and exceeds the editing window, developers can also use the direction keys to move the editing window. Under such situation, using direction keys to fine-tune widget location will remain no actions until the editing window is moved to the limit.

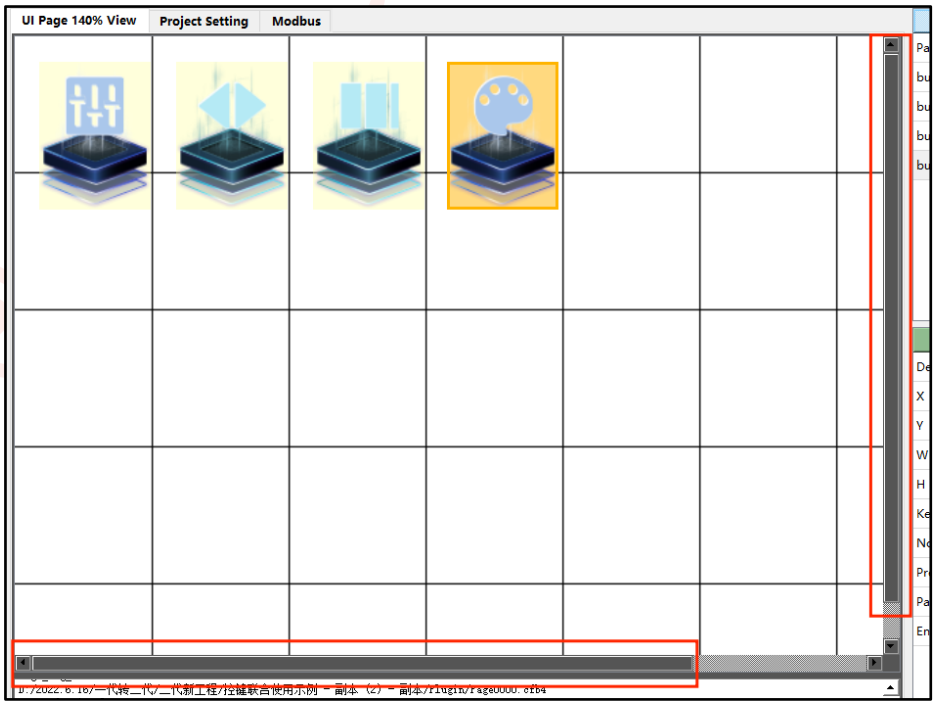
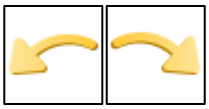


Figure 5-18: Fine-tune Widget Location

5.8.7. Load previous step & Load next step

ICON: 

Load previous step: Undo operation (short-key: Ctrl+Z).

Load next step: Redo operation (short-key: Ctrl+Y).

Note:

1. Undo / Redo operations are only valid for the current editing page. The most 50 operations can be undone / redo.
2. Following operations will be recorded: (1) Move widgets; (2) Add/Delete widgets.
3. If a widget parameter is modified, it will only be recorded when (1) the current page is switched; (2) the project is compiled and saved.

5.8.8. Lock & Unlock

ICON: 

Function: Used to Lock/Unlock the widget location.

5.9. Short Keys

Generate UartTFT-II_Flash.bin	:	Ctrl + B
Fine-tune Widget Location	:	4 direction keys (← ↑ ↓ →)
New Project	:	Ctrl + N
Open Project	:	Ctrl + O
COPY	:	Ctrl + C
PASTE	:	Ctrl + V
DELETE	:	Delete
Zoom-in	:	Ctrl + I
Zoom-out	:	Ctrl + U
Original Size	:	Ctrl + Q
Set writeAddr	:	Ctrl + A
Undo	:	Ctrl + Z
Redo	:	Ctrl + Y
Save All	:	Ctrl + S
Open the User Manual	:	F1

6. Widget

6.1. Button

ICON: 

Function : Jump to designated page

name : Widget name, User-definable

X & Y : Left-top coordinates of the Button

W & H : The width and height of the Button. Developers may set the width and height for virtual buttons. If an icon is added, then its width and height will be adapted automatically.

returnValue : Report value (through Uart), user-definable. Valid when [reportToHost] is set to Enable.

unpressedIcon : Icon for the button (unpressed state)

pressedIcon : Icon for the button (pressed state)

The width / height of unpressedIcon and pressedIcon must be the same.

pageGoto : Setup which page to jump to if the button is pressed.

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

Figure 6-1: Button

reportToHost : Set [Enable] to report the returnValue through Uart interface if the button is pressed, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

hostControl : Set [Enable] to allow host to trigger the button. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the widget.

6.2. SlideMenu

ICON: 

Function : Better visualize the slide menu.

name : Widget name, User-definable

writeAddr : Variable address, user-definable.

X & Y : Left-top coordinates of the SlideMenu

W & H : The width and height of the SlideMenu. When the sliding direction is horizontal, the height does not need to be modified, and the width should be set according to the material. When the sliding direction is vertical, the width does not need to be modified. As shown in Figure 6-3, H0 is the height of a single digit, unit: Pixel.

Parameter	Data
name	slmenu_0
writeAddr	0x0000
X	348
Y	72
W	197
H	124
L1	30
L2	30
direction	Vertical
foreground	
background	
minValue	0
maxValue	10
defaultValue	0
adjStep	1
reportToHost	Disable

Figure 6-2: Slide Menu

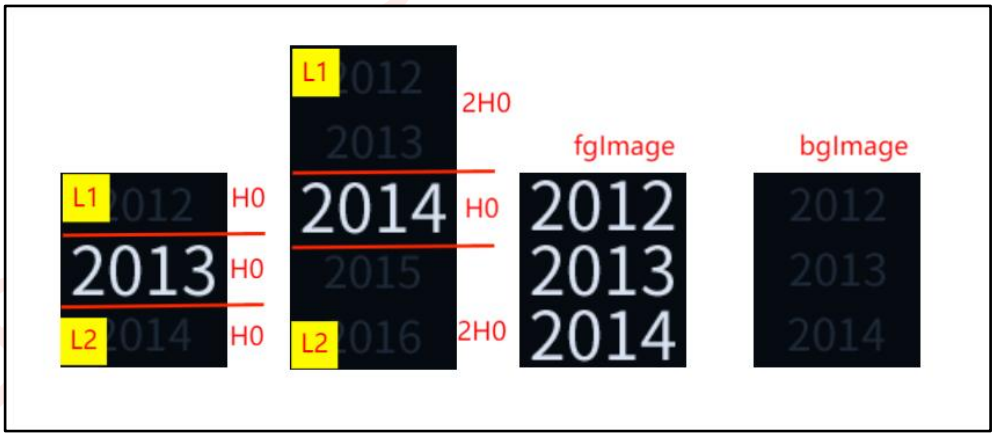


Figure 6-3: Example of SideMenu

L1 & L2 : As shown in Figure 6-3, the height of L1 is the same as that of L2. There are two examples: (1) To choose from a display of 3 options. As shown in Figure 6-3, the first picture on the left, the SlideMenu is separated into three parts, where L1 = L2 = H0, and the height of the selected area (in the middle) is also H0. Therefore, the total height (H) is 3H0; (2) To choose from a display of 5 options. As shown in Figure 6-3, the second picture on the left, the SlideMenu is also separated into three parts, whereas L1 = L2 = 2H0, and the height of the selected area (in the middle) is H0. Therefore, the total height (H) is 5H0.

Developers may follow below rules to setup L1, L2, and H:

Assume there are N (must be an odd number) options in a display, then

L1 = L2 = H0 * (N – 1) / 2, H = N * H0

Same rules apply to horizontal SlideMenu, simply change H to W

direction : Setup sliding direction
(4 options: Vertical / Horizontal / Vertical-Loop / Horizontal-Loop).

foreground : Foreground Image, as shown in Figure 6-3, the second picture on the right.

background Background Image, as shown in Figure 6-3, the first picture on the right.

minValue & maxValue : Setup the range of selection, based on the prepared material. If there are 10 options in the prepared material, for example, Year 2020 to Year 2029, then minValue can be set to 20, and maxValue can be set to 29.
Settable range is 0 ~ 65535.

defaultValue : Default value, must be within the minValue and maxValue.

adjStep : Movement of each slide operation. One step = H0, in pixel.

reportToHost : Set [Enable] to report the writeAddr and data through the Uart interface if the SlideMenu is operated, otherwise set [Disable]. Refer to *Touch Returned Message* for more detail.

Note :
1. SlideMenu cannot be used to adjust backlight;
2. The picture size should meet the following conditions:
   (1) H < 8192; (2) W < 8192; and (3) W*H < 800*480.
3. When using Uart commands to change the display of the SlideMenu, the display of the SlideMenu will not be updated until the page is refreshed.

6.3. PopupBox

ICON: 

Function : Add a popup box to better visualize the display

name : PopupBox name, user-definable

X & Y : Left-top coordinates of the PopupBox

W & H : The width and height of the PopupBox. When an icon is added, its width and height will be adapted automatically.

returnValue : Report value (through Uart), user-definable.

unpressedIcon : Icon for the PopupBox (unpressed state)

pressedIcon : Icon for the PopupBox (pressed state). The width / height of unpressedIcon and pressedIcon must be the same.

pageGoto : Setup which page to jump to if the PopupBox is pressed.

box_X & box_Y : The left-top coordinate of the PopupBox.

dimming : Background mode. There are two modes:

(1) Disable: The background brightness remains the same;

(2) Enable: The background brightness will be dimmed, yet the PopupBox brightness will be normal.

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

Figure 6-4: PopupBox

reportToHost : Set [Enable] to report returnValue through Uart interface to host if the Popupbox is triggered, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

clearLastPopupBox : Set [Enable] to clear PopupBox after exiting the PopupBox, set [Disable] otherwise.

hostControl : Set [Enable] to allow host to trigger the PopupBox. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the widget.

backgroundPage : Set a background page for the PopupBox.

6.4. Variable Button

ICON: 

Function : To increase/decrease the value of a designated variable when the button is pressed

name : Name of the Variable Button, user-definable

X & Y : Left-top coordinates of the Variable Button

W & H : The width and height of the Variable Button. When an icon is added, its width and height will be adapted automatically.

writeAddr : Start address of the variable value

adjStep : Increment / decrement value when the button is pressed.

maxValue & minValue : Setup for Maximum and Minimum value. These two values can be equal to each other. When these two values are set equal to each other, it means writing the value to the designated variable address when the button is pressed. The input value is in decimal form, ranging from -32768 ~ 32767.

dataType : There are 5 data types: uchar, char, ushort, short, and bitControl.

gradation : Set [+] to increase the value of the variable when the button is pressed; set [-] to decrease the value of the variable when the button is pressed.

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

Figure 6-5: Variable Button

cyclicalCounting : Set [Loop] to auto-adjust the value of the variable when it reaches min/max value. When it reaches the maximum value, then adjust the value to minimum when the button is pressed again. When it reaches the minimum value, then adjust the value to maximum when the button is pressed again.

longPress : [Once] : trigger the button one time when it is pressed and released.
[Repeat]: Long press is enabled. The button will be triggered continuously when it is pressed.

unpressedIcon : Icon for the button (unpressed state)

pressedIcon : Icon for the button (pressed state). The width / height of unpressedIcon and pressedIcon must be the same.

reportToHost : Set [Enable] to report writeAddr and data through Uart interface if the button is pressed, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

hostControl : Set [Enable] to allow host to trigger the button. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger:*

triggerValue for more detail.

_triggrValue : The data sent by the host to trigger the button.

Note:

1. When using a variable button with char or uchar data type to assign a value to certain address, the higher byte of such address will not be changed.
2. When using bitControl, maxValue and minValue can only be 1 or 0
3. When assigning values to a Text Number or Graphics Number widget, the data types should be set as the same.

6.5. Combo Button

ICON: 

Function : This button can recognize [Click], [Double Click], and [Long Press] operations, and execute the corresponding actions.

name : Name of the widget, user-definable

X & Y : Left-top coordinates of the widget.

W & H : The width and height of the widget. When an icon is added, its width and height will be adapted automatically.

unpressedIcon : Icon for the button (unpressed state)

pressedIcon : Icon for the button (pressed state). The width and height of unpressedIcon must be the same as that of pressedIcon.

click : Enable the [Click] function

double click : Enable the [Double Click] function

long press : Enable the [Long Press] function

Click_keyValue : The key value of the [Click] operation

Click_PageGoto : The page to switch to when the [Click] operation is detected.

writeAddr : The variable address to set when the [Click] operation is detected.

_value : The value of the designated variable to set (in hexadecimal), when the [Click] operation is detected.

Parameter	Data
name	comboButton_1
X	564
Y	59
W	119
H	105
unpressedIcon	
pressedIcon	
click	Enable
double click	Disable
long press	Disable
Click_keyValue	0x0000
Click_PageGoto	
writeAddr	0xFFFF
_value	0xFFFF
DoubleClick_ke...	0x0000
DoubleClick_de...	10
DoubleClick_Pa...	
writeAddr	0xFFFF
_value	0xFFFF
LongPress_key...	0x0000
LongPress_delay	10
LongPress_Pag...	
writeAddr	0xFFFF
_value	0xFFFF
reportToHost	Disable
hostControl	Disable
_triggerValue	0x0000

Figure 6-6: Combo Button

DoubleClick_keyValue : The key value representing the [Double Click] operation.

DoubleClick_delay : Set the effective time gap for double click. (50 * N ms, where 1<=N<=255)

DoubleClick_PageGoto : The page to switch to when the [Double Click] operation is detected.

writeAddr : The variable address to set when the [Double Click] operation is detected.

_value : The value of the designated variable to set (in hexadecimal), when the [Double Click] operation is detected.

LongPress_keyValue : The key value representing the [Long Press] operation.

LongPress_delay : Set the effective lasting time for the long-press operation, (50 * N ms, where 1<=N<=255)

LongPress_Page Goto	:	The page to switch to when the [Long Press] operation is detected.
writeAddr	:	The variable address to set when the [Long Press] operation is detected.
_value	:	The value of the designated variable to set (in hexadecimal), when the [Long Press] operation is detected.
reportToHost	:	Set [Enable] to report the Key value through the Uart interface if the Combo Button is operated, otherwise set [Disable]. Refer to *Touch Returned Message* for more detail.
hostControl	:	Set [Enable] to allow host to trigger the button. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.
_triggerValue	:	The data sent by the host to trigger the button.

Note:

1. Developers may set the [writeAddr] as the specialized register, 0x7014, to assign values to multiple variables. Refer to *Variable Association List* for more detail.
2. When [writeAddr] is set to 0x7014, the setting value of [_value] should be one of the ID numbers in the Variable Association List.

6.6. Circular Touch

ICON: 

Function : To control a variable by Circular Touch

name : Name of the Circular Touch, user-definable.

writeAddr : Address of the variable

X & Y : Left-top coordinates of the Circular Touch.

W & H : The width and height of the Circular Touch. When an icon is added, its width and height will be adapted automatically.

foreground : Foreground image, as the middle picture (blue circle) shown in Figure 6-7.

background : Background image, as the right picture (white circle) shown in Figure 6-7. The width/height of the foreground and background images must be the same.

slideButton : Slider Button, as ❶ shown in Figure 6-7. The distance from the center of the sliderButton to the edge of the foreground (or background) picture should be larger than the radius of the sliderButton, as the mark, “d”, shown in Figure 6-7.

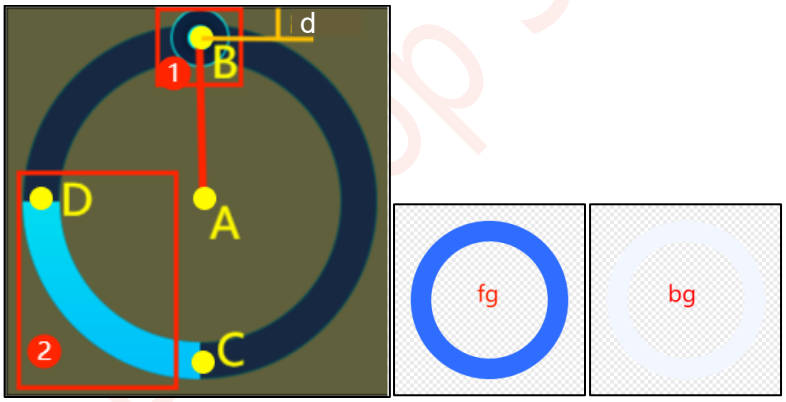


Figure 6-7: Circular Touch

Parameter	Data
name	rtouch_0
writeAddr	0x4AB1
X	154
Y	108
W	50
H	70
foreground	
background	
slideButton	
slide_R	50
touch_R	50
minValue	0
maxValue	100
defaultValue	0
startAngle	0
finalAngle	359
promptNum_x	25
promptNum_y	35
integerDigit	3
decimalDigit	0
alignment	Left
fontID	
fontColor	0x000000
firstIcon	
lastIcon	
digitDisplay...	NULL
reportToHost	Disable
Touch Ctrl	Enable

Figure 6-8: Circular Touch

slide_R : The distance from A to B, as shown in Figure 6-7. Move the slider button to the center of the circular rail as shown in Figure 6-9, UI_Eidtor-III will auto calculate the value.

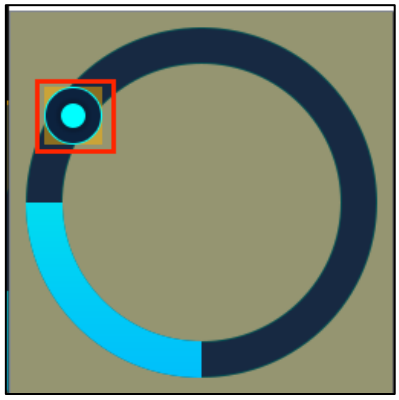


Figure 6-9: slide_R

touch_R : Radius of the touch area. As shown in Figure 6-7, based on the center of B.

minValue & maxValue : The range of the circular touch. -32768 ~ 32767.

defaultValue : Default value, must be between minValue and maxValue.

startAngle : Start Angle of the slider area.

finalAngle : Final Angle of the slider area.
As shown in Figure 6-7, C represents 0 degree, and the rotation degree will increase when rotating clockwise. In addition, startAngle cannot be larger than finalAngle (0°<= startAngle < finalAngle <=360°). This widget cannot be set to increase the degree counterclockwise.

promptNum_X & promptNum_Y : The coordinate of the prompt number. Reference point is the left-top coordinate of the widget. Also, the alignment mode should be set before setting the coordinate of the prompt number.

integerDigit : Number of integer digits of the prompt number.

decimalDigit : Number of decimal digits of the prompt number.

alignment : Alignment mode (only for the horizontal direction) for the prompt number.

Options include Left, Middle, and Right. The left-top X coordinate (promptNum_X) of the prompt number is used as the base line.

As shown in Figure 6-10:

[Left]: Display the prompt number as its left-top coordinate setting (promptNum_X, promptNum_Y)

[Middle]: Horizontally align the middle of the prompt number to the base line.

[Right]: Horizontally align the right of the prompt number to the base line.

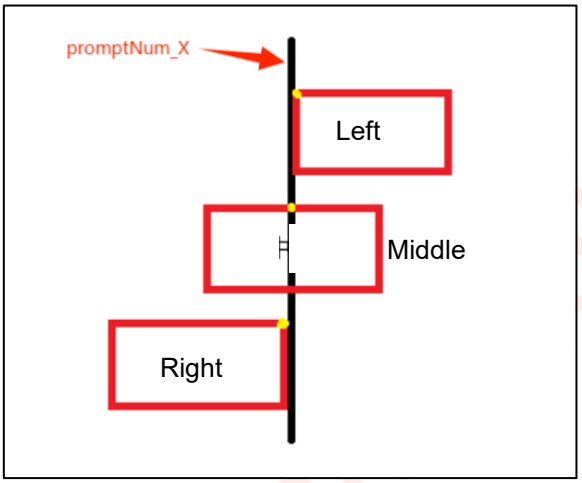


Figure 6-10: Prompt Number Alignment

fontID : Select from a Font list for the prompt numbers

fontColor : Set the font color for the prompt numbers

firstIcon : The first Png picture, which should be the number “0”

lastIcon : The last Png picture, which should be the number “9” or the decimal point “.”

digitDisplayMode : Select the form of the prompt numbers, including [Null], [FontNum], and [IconNum].

[NULL]: No prompt number used

[FontNum]: Using Font characters

[IconNum]: Using Png numbers

reportToHost : Set [Enable] to report the writeAddr and its value through Uart interface if the widget is operated, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

Touch Ctrl : Set [Enable] to allow touch operations, set [Disable] otherwise.

Note: foreground and **background** Images must be set and cannot be left empty.

6.7. Slider Bar

ICON: 

Function	:	To control a variable by Slider Bar
name	:	Name of the Slider Bar, user-definable.
writeAddr	:	Start address of the variable value
X & Y	:	Left-top coordinates of the Slider Bar
W & H	:	The width and height of the Slider Bar. When an icon is added, its width and height will be adapted automatically. If no background picture added, the width and height will need to be set manually.
touch_X & touch_Y	:	The left-top coordinate of the touch area. The reference point (0, 0) is the left-top coordinate of the Slider Bar.
touch_W & touch_H	:	The width and height of the touch area. The setting range must be within the background picture.
minValue & maxValue	:	The range of the Slider Bar. -32768 ~ 32767
defaultValue	:	Default location of the Slider Bar.

Parameter	Data
name	slider_0
writeAddr	0x4AB2
X	221
Y	371
W	114
H	104
touch_X	5
touch_Y	5
touch_W	104
touch_H	94
minValue	0
maxValue	100
defaultValue	0
bar_X	0
bar_Y	0
barIcon	
slideButton	
background	
direction	L_to_R
reportToHost	Disable
Touch Ctrl	Enable

Figure 6-11: Slider Bar

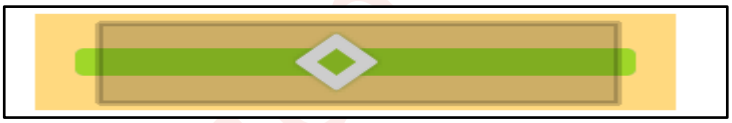


Figure 6-12: Slider Bar Example

bar_X & bar_Y	:	The left-top coordinate of the foreground picture. The reference point (0, 0) is the left-top coordinate of the Slider Bar.
barIcon	:	Foreground picture, as the green area shown in Figure 6-12
slideButton	:	Slider button, as the rhombus shape shown in Figure 6-12. The height (width) of the slider button must be larger than or equal to that of the foreground picture.
background	:	Background picture, as the yellow area shown in Figure 6-12. The background picture can be omitted.
direction	:	Sliding direction, from the small value to the larger value.
reportToHost	:	Set [Enable] to report the writeAddr and its value through Uart interface if the widget is operated, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.
Touch Ctrl	:	Set [Enable] to allow touch operations, set [Disable] otherwise.

6.8. Keyboard

6.8.1. Setup the keyboard widget

To use a keyboard widget, its materials must be set first. Refer to the steps below:

Step I: As the figure shown below, prepare two keyboard pictures, one represents the unpressed state, another represents the pressed state. The size the two pictures should be the same.

Figure 6-13: Keyboard Pictures

Step II: Add the two pictures to the Page list of UI_Editor-II, as shown below:

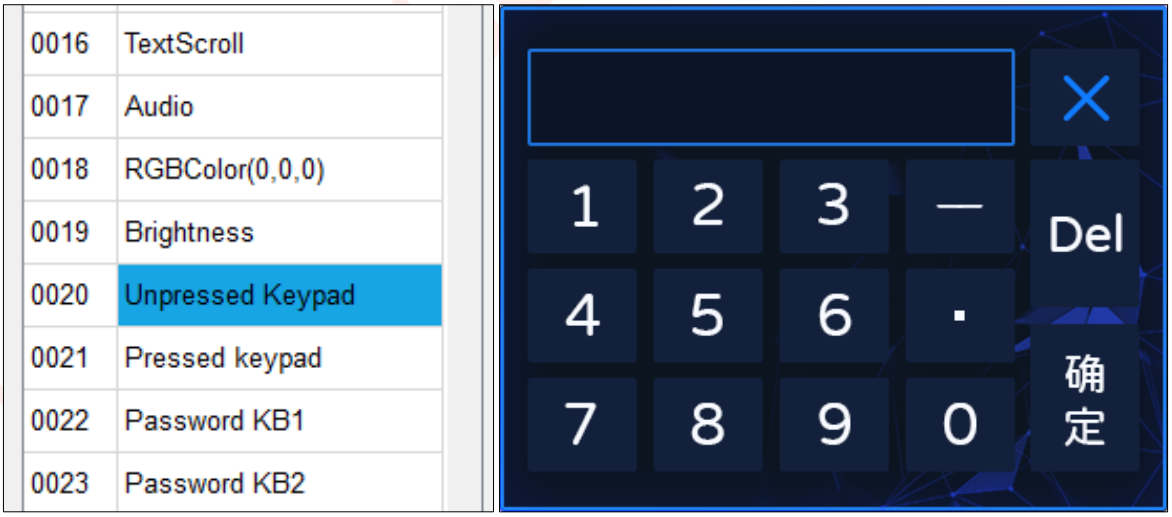


Figure 6-14: Add Keyboard Picture (unpressed state)

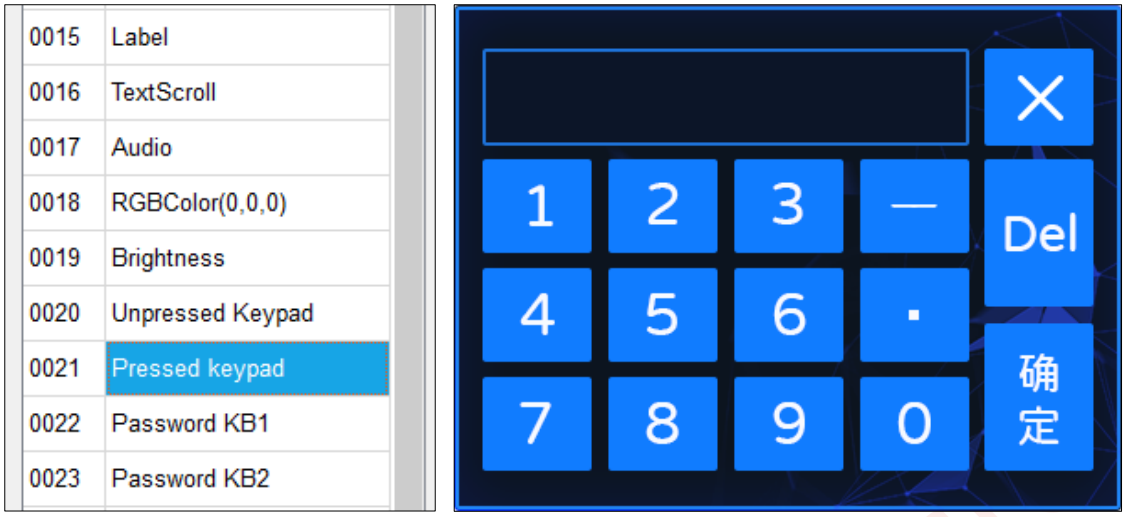


Figure 6-15: Add Keyboard Picture (pressed state)

Step III: Add SingleKey widgets to the page of Keyboard picture (unpressed state), as shown in Figure 6-16.

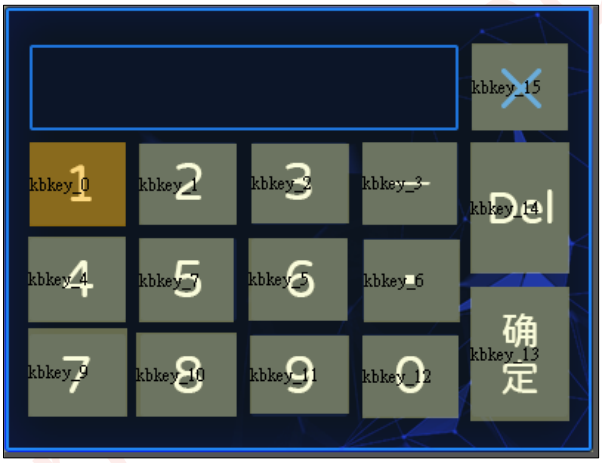


Figure 6-16: Add SingleKey Widgets

Note that only SingleKey widgets are allowed to be added to the page of keyboard picture. In the parameter table of the SingleKey widget (at "1" location), as shown in Figure 6-17, the keyCode parameter, is set to 0x0031 which is the ASCII code of number "1". (Refer to *Setup Keyboard Key-code* for the key-code list.) In addition, the SingleKey parameter, pressPage, should be pointed to the location of the keyboard picture with pressed state, which is Page0021 in the example here.

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

Figure 6-17: Setup keycode and pressPage

After the above materials are set, the parameter of Keyboard widget, pageID, should be set to the location of the keyboard picture with unpressed state, which is Page0020 in the example here, as shown in Figure 6-18.

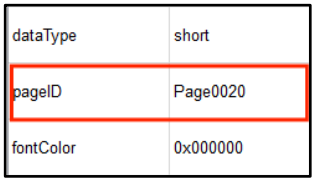


Figure 6-18: Setup Keypad Widget

Note:

1. Only SingleKey widgets are allowed to be added to the page of the keyboard picture of unpressed state.
2. No widgets are allowed to be added to the page of the keyboard picture of pressed state.

6.8.2. SingleKey

ICON: 

Function	:	Assign a key-code to each key
name	:	Name of the key, user-definable.
X & Y	:	Left-top coordinates of the key
W & H	:	The width and height of the key, roughly based on the key size on the keyboard picture.
keyCode	:	Key code
pressPage	:	When the key is pressed, the corresponding location of the designated page, as Figure 6-15, will be shown.

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

Figure 6-19: SingleKey

Note: Refer to *Setup Keyboard Key-code* for the key code list.

6.8.3. Numeric Keypad

ICON: 

Function : Write a number to the designated address. The entering number is by ASCII coding.

name : Name of the Numeric Keypad, user-definable.

writeAddr : Starting address of the entered number.

byteLength : Data Length, auto-adjusted by the datatype. No need to modify

X & Y : Left-top coordinates of the triggered area.

W & H : The width and height of the triggered area.

kpad_X & kpad_Y : Left-top coordinate of the pop-up keypad. The reference point (0, 0) is the left-top coordinate of the current page.

input_X & input_Y: : Left-top coordinate of the number input area. The reference point (0, 0) is the left-top coordinate of the Numeric Keypad page.

input_Max & input_Min : Set the max and min values of the input number. Theses settings are only valid when **inputLimit** is enabled. In addition, the input range is limited by the setting value of **integerDigit** and **decimalDigit**. Maximum input range is [-2147483647，2147483647]

integerDigit : The number of integer digits allowed.

decimalDigit : The number of decimal digits allowed.

fontWidth : The width of the number – auto adapted by the selected font, no need to modify.

inputLimit : Set [Enable] to limit the entered digits to be within the value set by **input_Max** and **input_Min**.

Parameter	Data
name	keypad_0
writeAddr	0x0902
byteLength	8
X	43
Y	276
W	429
H	96
kpad_X	504
kpad_Y	244
input_X	33
input_Y	38
input_Max	999999999
input_Min	0
integerDigit	8
decimalDigit	2
fontWidth	30
inputLimit	false
inputMode	New
alignment	Left
cursorColor	0xFFFFFF
cursorEnable	Enable
fontID	07_Font-GB...
dataType	longlong
pageID	Page0022
fontColor	0xFFFFFF
reportToHost	Disable
background...	
hostControl	Disable
_triggerVa...	0x0000

Figure 6-20: Numeric KeyPad

inputMode : Input Mode. There are two modes,

3. Modify: A default value will be shown in the entry box of the Numeric Keypad.
4. New: No value will be shown in the entry box of the Numeric Keypad.

(1) Modify:

(2) New:

alignment : Alignment mode. The display output is as shown below:

(1) Left:

(2) Middle:

(3) Right:

cursorColor : Set the cursor color

cursorEnable : Set [Enable] to show the cursor; set [Disable] otherwise.

fontID : Select a font

dataType : Select a data type

pageID : The page ID of the Numeric Keypad. The designated page must have the picture of the Numeric Keypad (unpressed state). This parameter must be set and cannot be left empty.

fontColor : Set the color of the entered number

reportToHost : Set [Enable] to report the input number and writeAddr through Uart port after the [Enter] key is pressed, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

backgroundPage : Set a background page for the Numeric Keypad.

hostControl : Set [Enable] to allow host to trigger the widget. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the Numeric Keypad.

Note1 : When assigning values to [Text Number Display] or [Graphics Number Display] through a Numeric Keypad, their parameters such as writeAddr, dataType, integerDigit, and decimalDigit must be the same, otherwise, the assigned value will be incorrect.

Note2 : input_Max, input_Min, integerDigit, and inputLimit are used to specify the input range more clearly. As shown in Table 6-1, if inputLimit is enabled (set to [Enable]), whereas the setting values of input_Min/Max conflict with that of integerDigit, then the input number will be limited by the setting value of IntegerDigit.

Table 6-1: Exzample of Input Range Limitation

integerDigit	inputLimit	input_Max	input_Min	Input Range
2	TRUE	60	30	30 — 60
		200	-200	-99 — 99
	FALSE	60	30	00 — 99
		200	-200	00 — 99

The input ranges of different data types are listed in Table 6-2.

Table 6-2: Input Ranges of Different Data Types

Data Type	Input Range	Maximum digit number	Recommended digit number
char	-128—127	2	2
short	-32768—32767	4	4
int	-2^31—2^31-1	9	9
longlong	-2^63—2^63-1	18	18

Note: digit number = integerDigit + decimalDigit

6.8.4. EN_KeyBoard

ICON: 

Parameter	Data
name	enkeyB_0
writeAddr	0xFFFF
wordLength	16
X	407
Y	55
W	69
H	63
fontID	
fontWidth	16
fontHeight	16
cursorColor	0x000000
cursorEnable	Enable
fontColor	0x000000
entryBox_X	0
entryBox_Y	0
pageID	
keyboard_X	0
keyboard_Y	0
inputMode	New
displayFormat	normal
reportToHost	Disable
background...	
hostControl	Disable
_triggerVa...	0x0000

Figure 6-21: EN_KeyBoard

Function : Input English letters.

name : Name of the EN_KeyBoard, user-definable.

writeAddr : Starting address of the input data

wordLength : Data length. Unit: Word. These addresses cannot be used by other widgets thereafter.

X & Y : Left-top coordinates of the triggered area.

W & H : The width and height of the triggered area.

fontID : Select a font

fontWidth : The width of the letter – auto adapted by the selected font, no need to modify.

fontHeight : The height of the letter – auto adapted by the selected font, no need to modify.

cursorColor : Set the cursor color

cursorEnable : Set [Enable] to show the cursor; set [Disable] otherwise.

fontColor : Set the color of the letters

entryBox_X & entryBox_Y : Left-top coordinate of the letter entry box. The reference point (0, 0) is the left-top coordinate of the EN_Keyboard page.

pageID : The page ID of the EN_Keyboard. The designated page must have the picture of the English Keyboard (unpressed state). This parameter must be set and cannot be left empty.

keyboard_X & keyboard_Y : Left-top coordinate of the pop-up English Keyboard. The reference point (0, 0) is the left-top coordinate of the current page.

inputMode : Set [New] to start a new input; set [Modify] to read the existed value of the designated address and display the data in the entry box. Please note that English Keyboard does not support reading Chinese characters.

displayFormat : Set [Star] to display the entered letter as the symbol, ' * '.

reportToHost : Set [Enable] to report the input data and writeAddr through Uart port after the [Enter] key is pressed, Set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

backgroundPage : Set a background page for the EN_KeyBoard widget

hostControl : Set [Enable] to allow host to trigger the EN_Keyboard. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the EN_Keyboard.

Note: English Keyboard can only be used to assign values to String_Label and Text Scroll widgets.

6.8.5. CN_KeyBoard

ICON: 

Function	:	Input Chinese characters. The coding table is as shown in Table 6-5.
name	:	Name of the CN_KeyBoard, user-definable.
writeAddr	:	Starting address of the input data
wordLength	:	Data length. This parameter is set to limit the length of the input data. An ending code, 0x0000, will be added to the end of the input string. Therefore, the default data length will be wordLength+1, and these addresses cannot be used by other widgets thereafter.
X & Y	:	Left-top coordinates of the triggered area.
W & H	:	The width and height of the triggered area.
显示文本字库	:	Select a Chinese Font, must be GBK font.
文字宽度	:	The width of the character – auto adapted by the selected font, no need to modify.
文字高度	:	The height of the character – auto adapted by the selected font, no need to modify.
拼音文本字库	:	Select a PinYin font, must be GBK font. Developers may set the same font for both 拼音文本字庫 and 显示文本字库.
拼音字母宽度	:	The width of the PinYin font – auto adapted by the selected font, no need to modify.
拼音字母高度	:	The height of the PinYin font – auto adapted by the selected font, no need to modify.
光标颜色	:	Set the cursor color
光标使能	:	Set [Enable] to show the cursor; set [Disable] otherwise.
输入文字颜色	:	Set the color of the Chinese Font.
输入文字坐标 X 和 Y	:	Set the left-top coordinate of the first entered character, as ❶ shown in Figure 6-23.The reference point (0, 0) is the left-top coordinate of the keyboard page.
提示文字颜色	:	Set the color of the prompt characters, as ❷ and ❸ shown in Figure 6-23.
拼音提示坐标 X 和 Y		Set the left-top coordinate of the PinYin prompt character, as ❷ shown in Figure 6-23. The reference point (0, 0) is the left-top coordinate of the keyboard page.

Parameter	Data
name	cnkeyB_0
writeAddr	0xFFFF
wordLength	16
X	417
Y	81
W	27
H	81
显示文本字库	
文字宽度	16
文字高度	16
拼音文本字库	
拼音字母宽度	16
拼音字母高度	16
光标颜色	0x000000
光标使能	Enable
输入文字颜色	0x000000
输入文字坐标X	0
输入文字坐标Y	0
提示文字颜色	0x000000
拼音提示坐标X	10
拼音提示坐标Y	24
汉字提示坐标X	10
汉字提示坐标Y	48
pageID	
键盘坐标X	0
键盘坐标Y	0
显示模式	New
文字间距pixel	3
reportToHost	Disable
background...	
hostControl	Disable
_triggerVa...	0x0000

Figure 6-22: CN_KeyBoard

汉字提示坐标 X 和 Y : Set the left-top coordinate of the Chinese prompt characters, as ❸ shown in Figure 6-23. The reference point (0,0) is the left-top coordinate of the keyboard page.

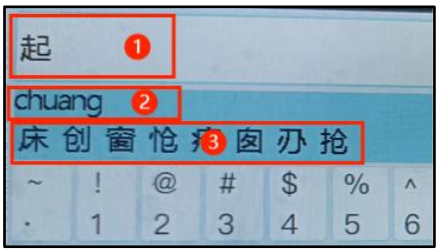


Figure 6-23: Example of PinYin Characters

pageID : The page ID of the keyboard. This parameter cannot be left empty.

键盘坐标 X 和 Y : The left-top coordinate of the pop-up keyboard. The reference point (0, 0) is the left-top coordinate of the current page.

显示模式 : Set [Modify] to import the data of designated address and display it onto the entry box of the keyboard, set [New] otherwise.

文字间距 pixel : Set the gap between characters.

reportToHost : Set [Enable] to report the input data and writeAddr through Uart port after the [Enter] key is pressed, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

backgroundPage : Set a background page for the CN_KeyBoard widget

hostControl : Set [Enable] to allow host to trigger the CN_Keyboard. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the CN_Keyboard.

Note: Chinese character can only be entered one by one. Also, the encoding of the related String_Label and Text Scroll widgets should be set the same as the font of CN_KeyBoard (GBK font).

6.8.6. Setup Keyboard Key-code

(1) Numeric Keypad

ASCII Function Code:

0x00F0：Cancel

0x00F1：Enter

0x00F2：Backspace

Table 6-3：Numeric Keypad Coding

Key-code List of Numeric Keypad			
Value	Key-code	Value	Key-code
0	0x0030	7	0x0037
1	0x0031	8	0x0038
2	0x0032	9	0x0039
3	0x0033	-	0x002D
4	0x0034	.	0x002E
5	0x0035	Cancel	0x00F0
6	0x0036	Enter	0x00F1
		Backspace	0x00F2

(2) EN_KeyBoard

ASCII Function Code:

0x00F0: Cancel

0x00F1: Enter

0x00F2: Backspace

0x00F3: Caps Lock

Table 6-4: English Keyboard Coding

Key-code List of EN_KeyBoard											
1st Row			2nd Row			3rd Row			4th Row		
Capital	Lowercase	Key-code	Capital	Lowercase	Key-code	Capital	Lowercase	Key-code	Capital	Lowercase	Key-code
~	`	0x7E60	Q	q	0x5171	A	a	0x4161	Z	z	0x5A7A
!	1	0x2131	W	w	0x5777	S	s	0x5373	X	x	0x5878
@	2	0x4032	E	e	0x4565	D	d	0x4464	C	c	0x4363
#	3	0x2333	R	r	0x5272	F	f	0x4666	V	v	0x5676
$	4	0x2434	T	t	0x5474	G	g	0x4767	B	b	0x4262
%	5	0x2535	Y	y	0x5979	H	h	0x4868	N	n	0x4E6E
^	6	0x5E36	U	u	0x5575	J	j	0x4A6A	M	m	0x4D6D
&	7	0x2637	I	i	0x4969	K	k	0x4B6B	<	,	0x3C2C
*	8	0x2A38	O	o	0x4F6F	L	l	0x4C6C	>	.	0x3E2E
(	9	0x2839	P	p	0x5070	:	;	0x3A3B	?	/	0x3F2F
)	0	0x2930	{	[	0x7B5B	"	'	0x2227	SP	SP	0x2020
_	-	0x5F2D	{	]	0x7D5D						
+	=	0x2B3D	|	\	0x7C5C						

(3) CN_Keyboard

GBK Function Code:

0x00F0: Cancel

0x00F1: Enter

0x00F2: Backspace

0x00F3: Caps Lock

0x00F4: PinYin / English input

0x00F5: Clear all input contents

0x00F7: Display the last Chinese character list (for Pinyin Chinese Characters)

0x00F8: Display the next Chinese character list (for Pinyin Chinese Characters)

Table 6-5: Chinese Keyboard Coding

Key-code List of CN_KeyBoard											
1st Row			2nd Row			3rd Row			4th Row		
Capital	Lowercase	Key-code	Capital	Lowercase	Key-code	Capital	Lowercase	Key-code	Capital	Lowercase	Key-code
~	`	0x7E60	Q	q	0x5171	A	a	0x4161	Z	z	0x5A7A
!	1	0x2131	W	w	0x5777	S	s	0x5373	X	x	0x5878
@	2	0x4032	E	e	0x4565	D	d	0x4464	C	c	0x4363
#	3	0x2333	R	r	0x5272	F	f	0x4666	V	v	0x5676
$	4	0x2434	T	t	0x5474	G	g	0x4767	B	b	0x4262
%	5	0x2535	Y	y	0x5979	H	h	0x4868	N	n	0x4E6E
^	6	0x5E36	U	u	0x5575	J	j	0x4A6A	M	m	0x4D6D
&	7	0x2637	I	i	0x4969	K	k	0x4B6B	<	,	0x3C2C
*	8	0x2A38	O	o	0x4F6F	L	l	0x4C6C	>	.	0x3E2E
(	9	0x2839	P	p	0x5070	:	;	0x3A3B	?	/	0x3F2F
)	0	0x2930	{	[	0x7B5B	"	'	0x2227	SP	SP	0x2020
_	-	0x5F2D	{	]	0x7D5D						
+	=	0x2B3D	|	\	0x7C5C						

6.9. State Button

ICON: 

Function : The state of this button can be represented by the pictures choosing from different image groups. By changing the content value of the associated variable address, a specific image group can be chosen for representing the button's unpressed and pressed states.

name : Name of the State Button, user-definable.

X & Y : Left-top coordinates of the button.

W & H : The width and height of the widget. When an icon is added, its width and height will be adapted automatically.

startDisplayID : The numbering of the first image group.

stopDisplayID : The numbering of the last image group.

Icon : Choose the first icon

Parameter	Data
name	stateButton_0
X	418
Y	442
W	76
H	49
startDisplayID	0
stopDisplayID	0
Icon	
writeAddr	0xFFFF
_value	0xFFFF
VarCtrlID	0
hostControl	Disable
_triggerVa...	0x0000

Figure 6-24: State Button

writeAddr : The associated variable address.

_value : The content value of the associated variable address.

VarCtrlID : The ID number in the Variable Association List.

hostControl : Set [Enable] to allow host to trigger the CN_Keyboard. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: _triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the State Button

Note:

5. The state buttons are controlled on a group-by-group basis, with each group corresponding to two images. Starting from the initial mapping value, each mapping value is equivalent to a group, and UI Editor compiles the two images. When the state button is not pressed, the first image of the group is displayed, and when the status button is pressed, the second image of the group is displayed.

1. Assign an initial value to the associated variable address (writeAddr), which corresponds to the mapping value of the state button. After powering on, the state button will display the mapping value corresponding to the initial value and the corresponding image. When the associated variable address is set to 0xFFFF, the state button will not be displayed.

2. For example, if the start mapping value is 0, the end mapping value is 2, and the first image is numbered 0050, then a total of 6 images from 0050 to 0055 will be compiled by UI-Editor-III, and the 6 images will be divided into 3 groups. 0050 and 0051 are the first group, and the mapping value is 0. 0052 and 0053 are the second group, with a mapping value of 1. 0054 and 0055 are the third group with a mapping value of 2. If the

associated variable address is 0x0010 and the initial value is 0x0000, after power on, the first set of image materials for the state button will be displayed, namely 0050 and 0051 images.

3. The state button can be used in conjunction with the variable batch processing tool (Variable Association List) to operate on multiple variables. Which ID number in the Variable Association List will be executed is determined by three parameters: (1) the current value of the associated variable (writeAddr); (2) the initial ID number in the Variable Association List; (3) the starting mapping value

   Calculation formula: The current value of the associated variable - the starting mapping value + the initial ID number in the Variable Association List = the ID number to be executed (valid number>0).

   When the value of the associated variable address is changed, the designated ID number of the Variable Association List will also change accordingly. For example, if the associated variable address (writeAddr) is 0x0010 and its initial value is 0000, and the initial ID number in the Variable Association List is 1, pressing the state button will batch process the variable with the ID number 1. When the value of the associated variable address 0x0010 is changed to 0001, the ID number will become 2, that is, pressing the state button will batch process the variable with the ID number 2, and so on.

6.10. Extend Button

ICON: 

Function : Make the values of two variable addresses equal, and after pressing, assign the value of the source variable address to the target variable address.

name : Name of the Extend Button, user-definable.

X & Y : Left-top coordinates of the button.

W & H : The width and height of the widget. When an icon is added, its width and height will be adapted automatically.

returnValue : Report value (through Uart), user-definable.

unpressedIcon : Icon for the button (unpressed state)

pressedIcon : Icon for the button (pressed state)

pageGoto : Setup which page to jump to if the button is pressed.

Parameter	Data
name	extendButto...
X	201
Y	475
W	91
H	57
returnValue	0x0020
unpressedIcon	
pressedIcon	
pageGoto	
purposeAddr	0xFFFF
scoreAddr	0xFFFF
CtrlLen	1
reportToHost	Disable
hostControl	Disable
_triggerVa...	0x0000

Figure 6-25: Extend Button

purposeAddr : Target variable address

scoreAddr : Source variable address

CtrlLen : The length of the data to be assigned to the target variable address.

reportToHost : Set [Enable] to report the returnValue through Uart interface if the button is pressed, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

hostControl : Set [Enable] to allow host to trigger the button. Note that the touch control function will be invalid when hostControl is enabled. Refer to *Widget Trigger: triggerValue* for more detail.

_triggerValue : The data sent by the host to trigger the widget.

Note: By setting the value n to CtrlLen, one can choose to copy the content of n bytes starting from the source variable address to the designated area that starts from the target variable address.

6.11. String_Label

ICON: 

Parameter	Data
name	label_0
parameterA...	0xFFFF
writeAddr	0x4AB6
wordLength	20
X	616
Y	18
W	60
H	38
fontWidth	40888
fontHeight	1791
fontID	
encoding	
alignment	Left
Horizontal	0
Vertical	0
background...	Disable
_color	0xD3D3D3
fontColor	0x000000
defaultText	label_0
passwordMo...	Disable
multiLangua...	Disable

Figure 6-26: String_Label

Function : Display Chinese, English, numbers, and special characters.

name : Name of the String_Label, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *String_Label: parameterAddr* for more details.

writeAddr : Starting address of the input string

wordLength : Default data length of the string. Unit: Word. The assigned storage space cannot be used by other unrelated widgets.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget. The height must be larger than or equal to the height of the font.

fontWidth & fontHeight : For prompting the font width and height only. No need to set them.

fontID : Select a Font

encoding : For prompting the selected font encoding types. No need to set it.

alignment : Alignment mode.
There are total 9 modes, combined by Horizontal and Vertical options.
Horizontal: Left / Middle / Right
Vertical: Top / Middle / Bottom

Horizontal : Horizontal gap

Vertical : Vertical gap

backgroundColor : Enable background color

_color : Set the background color

fontColor : Set the font color

defaultText : Set a string to be displayed when power-on

passwordMode : Set [Enable] to display the contents as the symbol, ' * ' (The contents will not be changed.)

multiLanguage : Set [Enable] to activate multi-language function. Refer to *Implement Multi_Language Display by Switching Text Code* for more detail.

Note:

1. Users may change the string contents either by sending character codes (Refer to *Write Commands to Control Widgets*) or by a keyboard widget.

2. The height of the widget must be set large enough for displaying multiple rows of contents.
3. When a String_Lable is updated by a keyboard widget, their font encoding must be set as the same.
4. If Unicode is used, then Win10 or above OS is suggested.

6.12. Static_Text

ICON: 

Function	:	To display fixed string contents of Chinese, English, numbers, and special characters without occupying the variable addresses.
name	:	Name of the widget, user-definable.
parameterAddr	:	The parameter contents of this widget are fixed. No need to set or change.
writeAddr	:	The variable address is not required by this widget. No need to set or change.
wordLength		The data length of the string. Unit: Word. The value will be auto calculated, according to the string content. No need to set or change.
X & Y	:	Left-top coordinates of the widget.
W & H	:	The width and height of the widget. The widget height must be larger or equal to the height of the font.
fontWidth & fontHeight	:	For prompting the font width and height. No need to set them.
fontID	:	Font selection. This parameter is auto-adapted, no need to set it.
encoding	:	For prompting the selected encoding types. No need to set it.

Parameter	Data
name	sText_0
parameterA...	0xFFFF
writeAddr	0x0000
wordLength	20
X	726
Y	26
W	55
H	30
fontWidth	4
fontHeight	0
fontID	
encoding	
alignment	Left
Horizontal	0
Vertical	0
background...	Disable
_color	0xD3D3D3
fontColor	0x000000
defaultText	Statictext_1
passwordMo...	Disable
multiLangua...	Disable
Constant_Nu...	0

Figure 6-27: Static_Text

alignment	:	Alignment mode. There are 9 different modes, combined by Horizontal and Vertical options. Horizontal: Left / Middle / Right Vertical: Top / Middle / Bottom
Horizontal	:	Horizontal gap
Vertical	:	Vertical gap
backgroundColor	:	Enable the background color
_color	:	Set the background color
fontColor	:	Auto-addapted by the numbering setting. No need to set it.
defaultText	:	Auto-addapted by the numbering setting. No need to set it.
passwordMode	:	Set [Enable] to display the contents as the symbol, ‘ * ‘ (The contents will not be changed.)
multiLanguage	:	Set [Enable] to activate multi-language function. Refer to *Setting Static_Text* for more detail.

Constant_Num : Select from the configured Static_Text numbers

Note:

1. The display content of the Static_Text widget is fixed, and cannot be changed by Uart commands.
2. If Unicode is used, then Win10 or above OS is suggested.

6.12.1. Setting Static_Text

A Static_Text widget can be setup by below steps:

Step 1: Click on the [Tool] Menu, and then click on [Static Text]

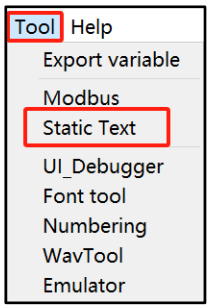


Figure 6-28: Activate Static_Text

Step 2: As shown in Figure 6-28, fill in the text, color, and font to the columns noted as ❸,❹, and ❺ respectively. See below description for other setting detail.

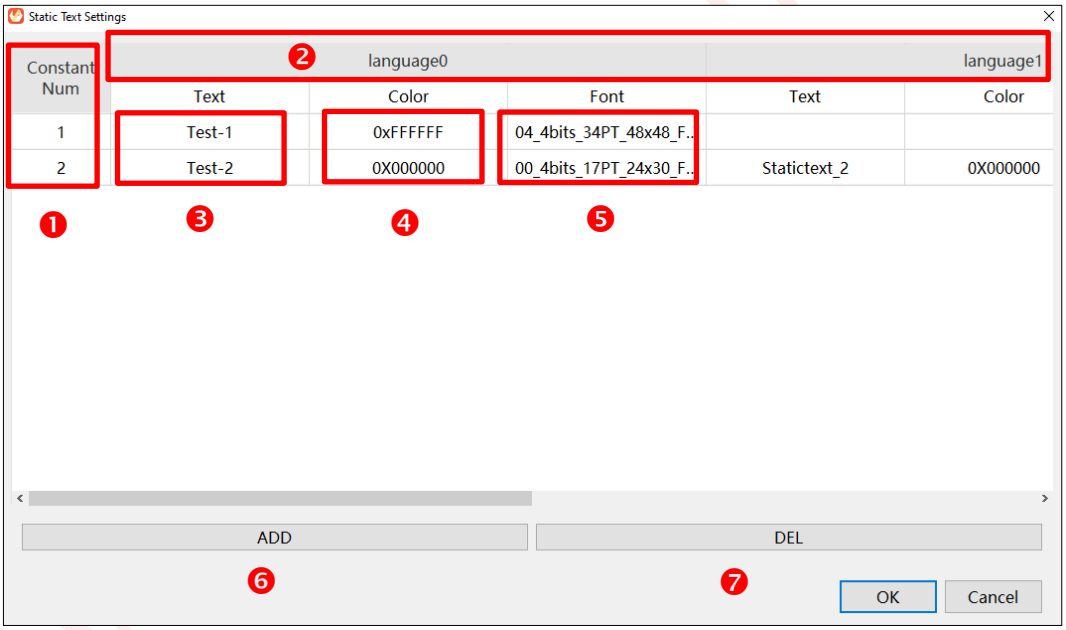


Figure 6-29: Setting Menu of Static_Text

❶ Number : This is the index number of the string list. The display content of a Static_Widget can be easily switched by assigning different index number to the parameter, Constant_Num.

❷ language0, language1 : When the Number of Language in the [Project Setting] is set > 1, the language options will be increased accordingly. Users may then fill in the string of each different language version for each index number.

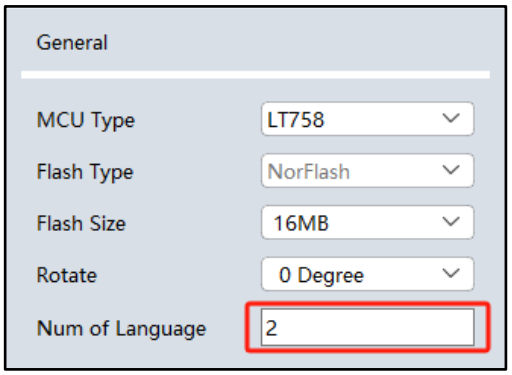


Figure 6-30: Set the number of Multi-Language

❸ Text : Fill in the strings for each index number
❹ Color : Assign the string colors for each index number
❺ Font : Select the Fonts
❻ ADD : Click to add one more index
❼ DEL : Click to delete one index

Step 3: Add a Static_Text widget, and assign a proper index number to the parameter, Constant_Num.

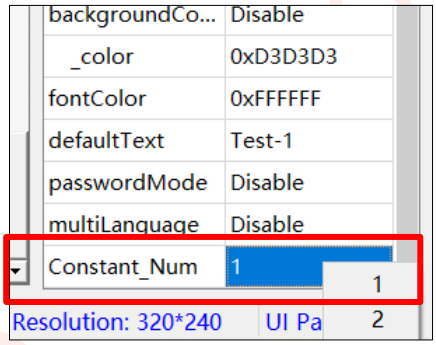


Figure 6-31: Set the Constant Number

Note: If multi-languages are configured, the display content of a Static_Text widget can be switched to different language versions by Uart commands, through the Register 0x703F.

For example, to switch the display content to [language1], the Uart command is as below:

5A A5 07 10 70 3F 00 01 0E CF

6.13. Text Scroll

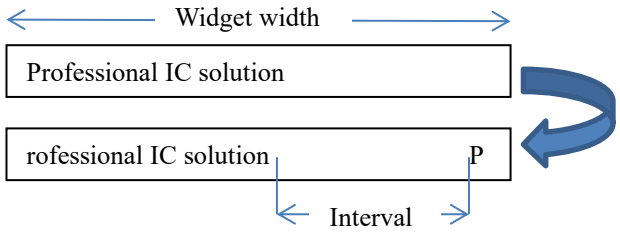
ICON: 

Function	:	Scroll text from right to left
name	:	Name of the widget, user-definable.
parameterAddr	:	Used to update widget parameters through Uart interface. Refer to *String: parameterAddr* for more details.
writeAddr	:	Starting address of the input text
X & Y	:	Left-top coordinates of the widget.
W & H	:	The width and height of the widget. The widget height must be larger or equal to the height of the font.
wordLength	:	Default data length of the string. Unit: Word. The assigned storage space cannot be used by other unrelated widgets.
fontWidth & fontHeight	:	For prompting the font width and height. No need to set them.
fontID	:	Select a font
encoding	:	For prompting the selected encoding types. No need to set it.
fontColor	:	Set the font color
backgroundColor	:	Set background color

Parameter	Data
name	textroll_0
parameterAddr	0xFFFF
writeAddr	0x5E1D
X	158
Y	119
W	176
H	56
wordLength	32
fontWidth	24
fontHeight	24
fontID	00_Font_1bit-...
encoding	GB2312
fontColor	0x000000
backgroundColor	0x0000FF
trailingSpace	64
interval(10ms)	50
alignment	Left
scrollMode	Enable
defaultText	textroll_0
transparency	Disable
multiLanguage	Disable

Figure 6-32: Text Scroll

trailingSpace: Refer to the below illustration. If the length of the text is longer than the widget width, the actual trailingSpace is as the set value. If the length of the text is shorter than the widget width **W**, the actual interval will be, **W – the length of the text**, no matter what the value is set. Unit: Pixel.



interval (10ms): The scrolling speed. Set 1 to move a pixel every 10ms; set 10 to move a pixel every 100ms. Setting range: 1 to 255

alignment	:	Alignment mode. This parameter is only effective when the text is not scrolling.
scrollMode	:	Set [Enable] to scroll the text, set [Disable] otherwise. Note: If the length of the text is longer than the widget width, then scroll mode will be enabled automatically, no matter what the scrollMode is set**.**
defaultText	:	Set a string to be displayed when power-on.
transparency	:	Set [Enable] to skip the background color, and display the text only, set [Disable] otherwise
multiLanguage	:	Set [Enable] to activate multi-language function. Refer to *Implement Multi-Language Display by Switching Text Code* for more detail.

Note:

1. The total text width (pixel) must be < X resolution of the panel * 2, where text width = fontWidth * number of characters.
2. Text contents can be changed either by sending character codes (Refer to *Write Commands to Control Widgets*) or by a keyboard widget.
3. There can be only one scrolling row per widget.
4. When using a keyboard widget to change the text, the font encoding must be the same for both [Text Scroll] and [Keyboard] widgets.

6.14. Text Number Display

ICON: 

Function : To display a number

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Text Number Display: parameterAddr* for more details.

writeAddr : Starting address of the number

byteLength : The length of the space for storing the number. This parameter is auto adapted according to the data type of the number.

X & Y : Left-top coordinates of the widget.

W & H : The width and height of the widget. The widget height must be larger than or equal to the height of the font.

fontWidth : Used to show the font width, no need to setup

fontID : Select a font

encoding : Used to show the coding type, no need to setup

alignment : Alignment mode. Options includes Left, Right, and Middle

integerDigit : Set the digit number of the integer.

decimalDigit : Set the digit number of the decimal. See *Digit Number of Integer & Decimal* for more details.

Parameter	Data
name	number_0
parameterA...	0xFFFF
writeAddr	0x4ACA
byteLength	2
X	97
Y	125
W	77
H	47
fontWidth	17328
fontID	
encoding	
alignment	Left
integerDigit	4
decimalDigit	0
dataType	short
unitSymbol	
_length	0
fontColor	0x000000
defaultNum...	0
leadingZero	Disable
addr	0xFFFF
_value	0xFFFF

Figure 6-33: Text Number

dataType : Data types include char, uchar, char_H8, uchar_H8, Short, ushort, int, uint, and long long. See *Data Type* for more details.

uniSymbol : Support symbols based on ASCII

_length : Number of bytes of the uniSymbol. (One byte for each ASCII character)

fontColor : Font color

defaultNumber : Default text to display after power on.

leadingZero : Set [Enable] to add leading zeros, set [Disable] otherwise.

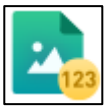
addr : Variable address for controlling the display of the Text Number.

_value : Set 0 to show the Text Number, set other numbers to hide the Text Number.

Note:

1. Only one decimal point is allowed. Redundant decimal points and the numbers behind them will be eliminated.
2. Refer to *Write Commands to Control Widgets* for the example of updating numbers by Uart port.

6.15. Graphics Number Display

ICON: 

Function : Display numbers by assembled png pictures including 0 ~ 9 numbers, a decimal point, and a minus sign.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Graphics Number Display: parameterAddr* for more details.

writeAddr : Starting address of the number

byteLength : The length of the space for storing the number. This parameter is auto adapted according to the data type of the number.

X & Y : Left-top coordinates of the widget.

W & H : The width and height of the widget. The height will be auto adapted, according to the imported png pictures.

integerDigit : Set the digit number of the integer.

decimalDigit : Set the digit number of the decimal. See *Digit Number of Integer & Decimal* for more details.

dataType : Data types include char, uchar, char_H8, uchar_H8, Short, ushort, int, uint, and long long. See *Data Type* for more details.

Parameter	Data
name	pngNumber_0
parameterA...	0xFFFF
writeAddr	0x4ACE
byteLength	2
X	395
Y	194
W	36
H	49
integerDigit	4
decimalDigit	0
dataType	short
alignment	Left
firstIcon	
lastIcon	
defaultNum...	100
leadingZero	Disable
addr	0xFFFF
_value	0xFFFF

Figure 6-34: Graphics Number

alignment : Alignment mode. Options include Left, Right, and Middle

firstIcon : Select the icon of “0”

lastIcon : Select the last icon based on display needs. Developers may assign either the icon of decimal point, minus sign, or the number “9” to this parameter.

defaultNumber : Default number to be displayed after power on.

leadingZero : Set [Enable] to add leading zeros, set [Disable] otherwise.

addr : Variable address for controlling the display of the Text Number.

_value : Set 0 to show the Graphic Number, set other numbers to hide the Graphic Number.

Note:

1. The order of the pictures is 0 ~ 9, decimal point, and then the minus sign.
2. Only one decimal point is allowed. Redundant decimal points and the numbers behind them will be eliminated.
3. Refer to *Write Commands to Control Widgets* for the example of updating numbers by Uart port.

6.16. Real Time Clock

6.16.1. Analog Clock

ICON: 

Function : Display an analog clock

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Analog Clock: parameterAddr* for more details.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget. These parameters will be auto adapted, according to the imported pictures.

hourHand_L : The length of the hour hand on the longer side. See example depicted in Figure 6-35.

hourHand_S : The length of the hour hand on the shorter side. See example depicted in Figure 6-35.

hourHand_W : The width of the hour hand.

hourHandColor : The color of the hour hand.

background : The background image.

centerIcon : The center image of the analog clock. (e.g. the dot image shown in Figure 6-34)

Parameter	Data
name	Clock_0
parameterAddr	0xFFFF
X	236
Y	70
W	209
H	200
hourHand_L	40
hourHand_S	15
hourHand_W	10
hourHandColor	0xB49600
minuteHand_L	50
minuteHand_S	15
minuteHand_W	6
minuteHandColor	0xB4FF00
secondHand_L	65
secondHand_S	20
secondHand_W	3
secondHandColor	0x00B400
background	
centerIcon	

Figure 6-35: Analog Clock

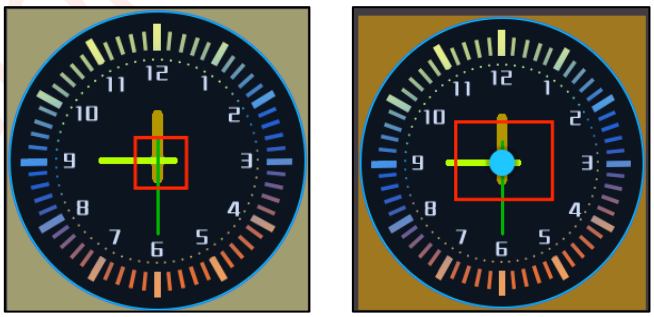


Figure 6-36: Example of Analog Clock

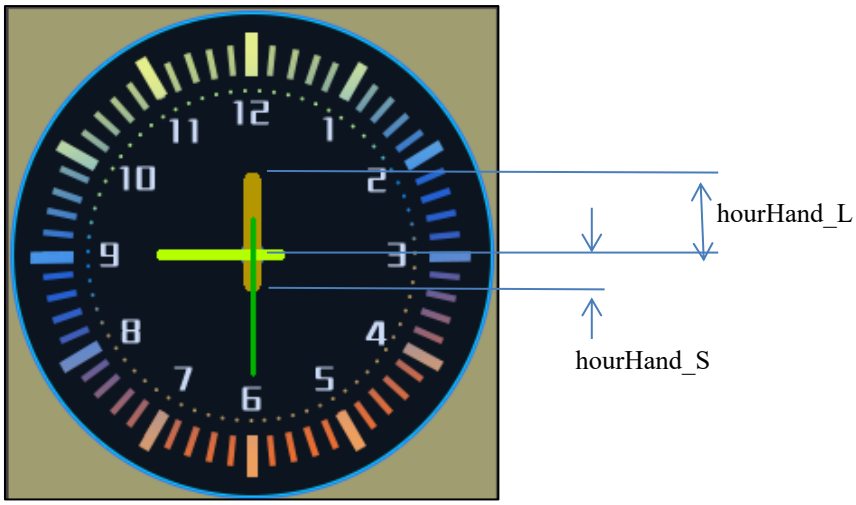


Figure 6-37: hourHand_L and hourHand_S

Note:

1. To set the parameters of minute hand and second hand, please refer to the description of hour hand.
2. This widget only works correctly when the RTC circuit is available.

6.16.2. Digital Clock

ICON: 

Function	:	To display a digital clock
name	:	Name of the widget, user-definable.
parameterAddr	:	Used to update widget parameters through Uart interface. Refer to *Digital Clock: parameterAddr* for more details.
X & Y	:	Left-top coordinate of the widget
W & H	:	The width and height of the widget
firstIcon	:	Select the picture of "0"
lastIcon	:	Select the picture of "Saturday" or "/(Day)"

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

Figure 6-38: Digital Clock

displayFormat : Display options, as shown in Figure 6-39.

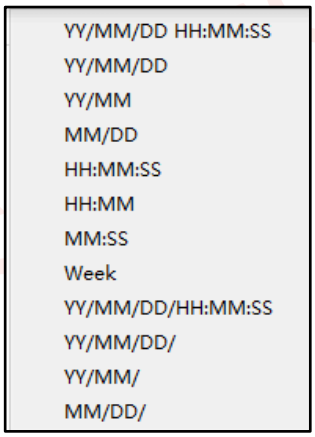


Figure 6-39: Display Options of Digital Clock

Note:

1. The order of the PNG pictures is 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, :, / (Year), / (Month), / (Day), Sun, Mon, Tues, Wed, Thu, Fri, Sat.
2. If week information is not needed, then only the below PNG pictures are required: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, :, / (Year), / (Month), / (Day)
3. The file number of '/ (Day)' cannot be used by other materials even if '/ (Day)' is not used.
4. Refer to *Icon Width & Height* for the setting rules about the Icon width/height.
5. This widget only works correctly when the RTC circuit is available.

6.16.3. How to update Date and Time

There are two steps for updating the date and time:

Step 1: Write data to the corresponding registers

Related registers: Year 0x7002, Month 0x7003, Day 0x7004, Hour 0x7005, Minute 0x7006, Second 0x7007

Step 2: Confirm the modification

Assign one of the values listed below to 0x7008:

0：Year, Month, Day, Hour, Minute, Second

1：Year, Month, Day

2：Year, Month

3：Month, Day

4：Hour, Minute, Second

5：Hour, Minute

6：Minute, Second

Note:

1. When updating Date and Time through Uart interface, simply write data to the registers of 0x7002 ~ 0x7007, no need to send confirmation value to 0x7008.
2. Refer to *Time Register - 0x7002 ~ 0x7007* for the example of updating Date/Time by Uart port.

6.17. Timer

ICON: 

Function : Set the timer and the operations to execute after the countdown is done.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Timer: parameterAddr* for more details.

X & Y : Left-top coordinate of the widget

W & H : The width and height of the widget. These parameters will be auto adapted, according to the imported picture.

presetAddr : The address of the target time.

_value : Set the target time in decimal, ranging from 1~65535, in seconds.

countAddr : The address of the counting time.

_value : Set the start counting time in decimal, ranging from 1~65535, in seconds.

controlAddr : The address of the control register of the timer.

_value Set timer operations:

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

firstIcon : Select the picture of “0”

lastIcon : Select the picture of “/(Day)”

displayFormat : Timer styles. Set NULL to hide the timer.

Parameter	Data
name	uitimer_0
parameterA...	0xFFFF
X	428
Y	306
W	59
H	86
presetAddr	0x4AD2
_value	120
countAddr	0x4AD3
_value	0
controlAddr	0x4AD4
_value	1:Start the ...
firstIcon	
lastIcon	
displayFormat	MM:SS
countMode	+
globalCount...	Disable
reportToHost	Disable
writeAddr	0xFFFF
_value	0xFFFF

Figure 6-40: Timer

countMode : Set the counting mode. “+” : incremental; “-” : decrement

globalCounting : Set [Enable] to keep the timer counting even if the display is switched to other pages. Set [Disable] otherwise.

reportToHost : Set [Enable] to report writeAddr0~7 and their values through Uart port after the counting is done, set [Disable] otherwise.

writeAddr : The address of the operation that will be executed after the counting is done. Developers may set the [writeAddr] as the specialized register, 0x7014, to assign values to multiple variables. Refer to *Variable Association List* for more detail.

_value : The value to be assigned to the address of the operation after the counting is done.

Note:

1. For incremental counting, the counting time = Preset_value – count_value, which means Preset_value must be greater than count_value, and the timer will count from the value of count_value to that of Preset_value. For example, if count_value = 60 and Preset_value = 180, then the timer will count 2 minutes (180 – 60 = 120 seconds) . The display will be initially 01:00, and then count to 03:00.
2. For decremented counting, the counting time = count_value no matter what the value of Preset_value is set. For example, if count_value = 60, then the timer will start at 01:00 and then countdown to 00:00.
3. When displayFormat is set to “SS”, the number will be displayed in seconds. The digit number will be based on the setting of Preset_value. For example, if Preset_value = 4, then the digit number of the timer is 4.
4. This widget only works correctly when the RTC circuit is available.

6.18. Gif

ICON: 

Function : To display a Gif picture.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *GIF: parameterAddr* for more details.

writeAddr : Address of the value for controlling Gif widget.

X & Y : Left-top coordinate of the Gif.

W & H : The width and height of the Gif. These parameters will be auto adapted, according to the imported picture.

playOnceCode : Set a value to represent the action of [play once]. When this value is assigned to the writeAddr, Gif will be played once.

startCode : Set a value to represent the action of [start playing]. When this value is assigned to the writeAddr, the related Gif will be played.

stopCode : Set a value to represent the action of [stop playing]. When this value is assigned to the writeAddr, the playing Gif will be stopped.

playAtStart : Set [Enable] to play Gif from the first frame. Set [Disable] to play Gif from where it is stopped.

Parameter	Data
name	gif_0
parameterA...	0xFFFF
writeAddr	0x4AD5
X	64
Y	238
W	70
H	65
playOnceCode	11
startCode	10
stopCode	100
disappearCo...	200
playAtStart	Disable
interval(10ms)	5
gifName	
dataFormat	
defaultStatus	Run
writeAddr	0xFFFF
_value	0xFFFF

Figure 6-41: Gif

Interval(10ms) : The time gap between frames. 10ms per unit, if the set value is 2, then the time gap is 20ms. Maximum setting value: 255.

gifName : Double-click to add a Gif. Select a file format from the pop-up window.

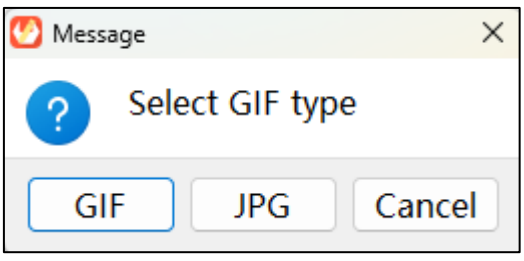


Figure 6-42: Select a Gif type

Click the [GIF] button to choose a gif file from the GIF file folder.
Click the [JPG] button to choose a sub-folder from the GIF file folder

The below example shows the contents of a GIF file folder:

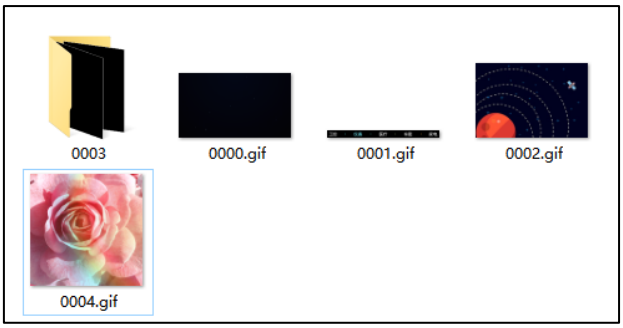


Figure 6-43: Example of a Gif file folder

As the example shown in the above figure, when the [GIF] button is clicked, developers may choose one of the gif files (e.g. 0000.gif, 0001.gif, 0002.gif, 0004.gif) from the folder.

When the [JPG] button is clicked, developers may choose one of the sub-folders listed in the GIF file folder. In the example of the above figure, 0003 will be the only sub-folder available.

In the sub-folder, there should be a group of JPG pictures to form an animation, as the example shown below:

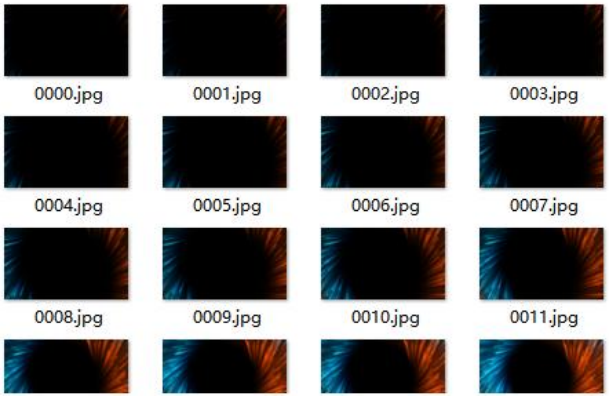


Figure 6-44: JPG files in the sub-folder

dataFormat : Set data format for the gif frames. See *dataFormat* for more details.

defaultStatus : Set the default status, including:

Run: Play the Gif in loop

Stop: Stop at the first frame

Disappear: No show

RunOnce: Play the Gif once and then stop

writeAddr : The address of the operation that will be executed after the GIF is done playing. Developers may set the [writeAddr] as the specialized register, 0x7014, to assign values to multiple variables. Refer to *Variable Association List* for more detail.

_value : The value to be assigned to the address of the operation.

Note:

1. Assigning 0xFFFF to the variable address of Gif can make the Gif disappeared.
2. When using Variable Button to control a Gif widget, its data type must be set as ushort.
3. Assign 0x7000 to writeAddrN to implement “Play once then jump to designated page” action. The designated page number (hexadecimal value) should be assigned to _valueN.
4. Gif can only be overlapped with graphic number, icon (αPNG), and text widgets. Each overlapped widget should be fully covered by the Gif widget to avoid abnormal display.
5. The refresh rate is related the size of each frame, and the interval setting.
6. Refer to *UartTFT-V3_Flash.bin* for the explanation about the size of the bin file converted from Gif.
7. In the GIF file folder, no files or folders should be named the same. In addition, other unrelated folders should not be listed under the GIF file folder, as this will result in compilation error.

6.19. QRCode

ICON: 

Function : To show a QR Code.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *QRCode: parameterAddr* for more details.

writeAddr : Start address of the QR code information.

byteLength : The length of the variable. Set by required data amount. The assigned storage space cannot be used by other unrelated widgets.

X, Y, W, H : The coordinate, width, and height of the QR code. The width and height will be auto adapted by the assigned value of size.

Parameter	Data
name	qrcode_0
parameterAddr	0xFFFF
writeAddr	0x0044
byteLength	200
X	362
Y	128
W	100
H	100
size(50pixels)	2
content	https://...

Figure 6-45: QR Code

size : The display magnification of the QR code. Available settings range from 1 to 6. The default size of the QR code is 50 pixels. For example, set 2 to enlarge the QR code to 2x50 = 100 pixels. The width and height will be changed accordingly.

content : Setting QR code information. Developers may update the information through Uart port when needed.

Note:

1. Refer to *Write Commands to Control Widgets* for the example of updating data by Uart port.

6.20. Audio Play

ICON: 

Function	:	To play audios. Support maximum 99 audio files.
name	:	Name of the widget, user-definable.
X, Y, W, H	:	The coordinate, width, and height of the widget.
wav ID	:	Click to assign an audio file.
repeat	:	Set [Enable] to play the assigned audio file in loop. Set [Disable] to play it only once.

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

Figure 6-46: Audio Play

Note: Only one Audio Play widget is allowed in a page.

How to Switch Audios:

Developers may assign the designated value to the Wave Control Register to play the desired audio. The address of the Wav Control Register is 0x700A.

Available operations (by assigning the below values to Wav Control Register):

0x0000: Stop playing

0x0001: Play the 1st audio file (The index number of the audio file is 0000) in the WavBin folder.

(Assign 0x0002 to play the 2nd audio file)

0x8001: Play the 1st audio file in loop

(Assign 0x8002 to play the 2nd audio file in loop)

6.21. Bit Status

ICON: 

Function : Display the designated picture based on the bit status of the data assigned to the variable address.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Bit Status: parameterAddr* for more details.

writeAddr : Variable address of the widget.

bitIndex : Set a designated bit, ranging from 0 to 15.
1. If this designated bit is 0, then the picture assigned to offStateIcon will be displayed.
2. If this designated bit is 1, then the picture assigned to onStateIcon will be displayed.
3. Initial value of the variable is 0x0000

Parameter	Data
name	bitIcon_0
parameterA...	0xFFFF
writeAddr	0x0005
bitIndex	bit0
X	743
Y	110
W	85
H	86
offStateIcon	
onStateIcon	
displayMode	Disable

Figure 6-47: Bit Status

X & Y : Left-top coordinate of the widget. The reference point (0, 0) is the left-top coordinate of the current page.

W & H : The width and height of the widget. These parameters will be auto adapted, according to the imported picture.

offStateIcon : Select a picture to be shown when the designated bit is 0.

onStateIcon : Select a picture to be shown when the designated bit is 1.

displayMode : [Disable]: To display the picture directly onto the base map regardless of the other existed widget images at the same location.
[Enable]: To display the picture by overlapping with the existed widget images at the same location.

Note:

1. Refer to *Write Commands to Control Widgets* for the example of updating data by Uart port.

6.22. Icon

ICON: 

Function : To display one or a set of icons.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Icon: parameterAddr* for more details.

writeAddr : Variable address of the widget.

byteLength : Variable data length

X & Y : Left-top coordinate of the widget. The reference point (0, 0) is the left-top coordinate of the current page.

W & H : The width and height of the widget. These parameters will be auto adapted, according to the imported picture.

firstIcon : Select the start picture

lastIcon : Select the last picture. To display a set of icons, these icons must be numbered in consecutive order, and their width/height must be the same.

Parameter	Data
name	icon_0
parameterA...	0xFFFF
writeAddr	0x0004
byteLength	2
X	352
Y	119
W	339
H	232
firstIcon	
lastIcon	
dataFormat	
defaultDispl...	
minDisplayID	0
maxDisplayID	0
displayMode	Disable

Figure 6-48: Icon

dataFormat : Set the icon data format, refer to *dataFormat*

defaultDisplayID : Set the default icon to be displayed once power-on. If this parameter is left empty, then the value will be 0.

minDisplayID & maxDisplayID : These two parameters must meet the condition of **max – min + 1 = the amount of the icons**. For example, if there are 10 icons named by consecutive numbers, 0100 ~ 0109, then minDisplayID and maxDisplayID can be set to [0, 9] or [10, 19]. The acceptable setting range is 0 ~ 65535.

displayMode : Set [Disable] to display the icon directly onto the base map regardless of the other existed widget images at the same location.

Set [Enable] to display the icon by overlapping with the existed widget images at the same location.

Note:

1. If an Icon widget controls only one icon, then only when minDisplayID = maxDisplayID = the value assigned to writeAddr, can the icon be displayed.
2. If an Icon widget controls a set of icons, then only when minDisplayID <= value assigned to writeAddr <= maxDisplayID, can the designated icon be displayed.

6.23. Trend Graph

ICON: 

Function : To display one trend graph based on the data transmitted by the MCU.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Trend Graph: parameterAddr* for more details.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the display area.

y_RefereceLine : The distance between the top of the widget display area and the baseline. Refer to Figure 6-47, unit: pixel.

_referenceValue : The value represented by the baseline. Refer to Figure 6-47, the baseline value is 2000, unit: pixel. When host sends a value of 2500, it will be displayed above the baseline. When host sends a value of 1500, it will be displayed below the baseline. Refer to *Example: Trend Graph* for more details.

lineColor : Set the line color

Parameter	Data
name	curve_0
parameterAddr	0xFFFF
X	443
Y	83
W	142
H	167
y_ReferenceLine	83
_referenceValue	83
lineColor	0x00B400
channel	0
x_Spacing(Pixels)	1
lineWidth	1
direction	R-L
maxData	256
minData	0

Figure 6-49: Trend Graph

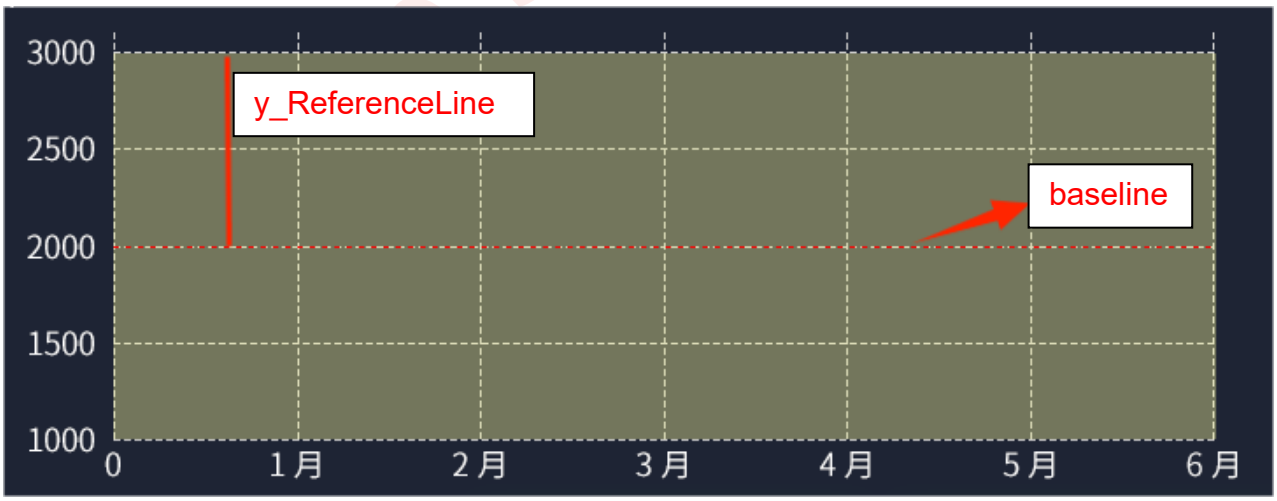


Figure 6-50: y_ReferenceLine and baseline

channel : Select the channel of the trend graph, ranging from 0 ~ 7.

x_Spacing(Pixels) : Set the horizontal gap between data points. Unit: pixel.

lineWidth : Set the line width of the trend graph. Unit: pixel.

direction : Trend Graph moving direction. R-L: from right to left; L-R: from left to right. As shown in Figure 6-48, where host sends two data (0x00C8, 0x0064) to channels with different direction settings.

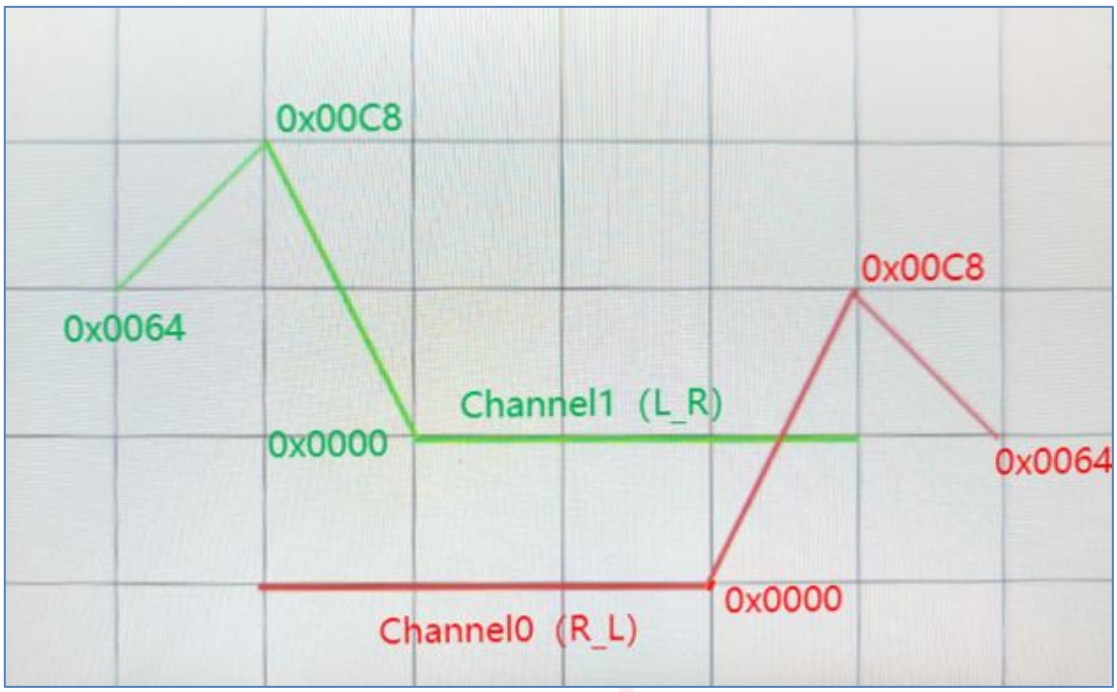


Figure 6-51: Example of Direction Settings

maxData : The maximum value that the widget area represents.

minData : The minimum value that the widget area represents.

As shown in the below figure, based on the widget area, maxData is set to 3000, and minData is set to 1000.

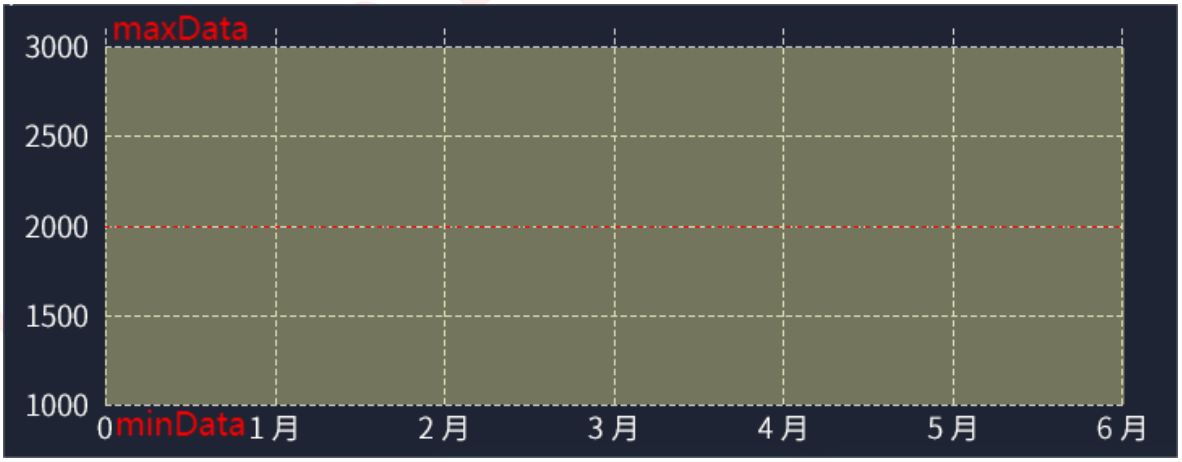


Figure 6-52: Example of maxData and minData

Note: To place multiple Trend Graph widgets in one page, the below rules must be followed.

1. The Trend Graph widgets should not be overlapped with each other.
2. If an overlapped display of different Trend Graph widgets is required, then their X, Y, W, and H parameters must be set to the same.

6.24. Encoder

ICON: 

Function: To operate the display by an Encoder, instead of a touch panel. A knob part will be needed for implementation. There are three operation modes available. When adding a new Encode widget, a pop-up window will be shown as the below figure. Users may then choose one mode from the available options. Please note that each page can have only one Encoder widget.

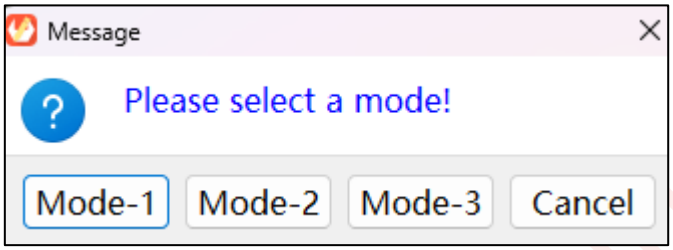


Figure 6-53: Encoder Modes

6.24.1. Encoder Mode-1

name	:	Name of the widget, user-definable.
writeAddr	:	Encoder Address. The icon to be controlled by the encoder should be assigned to the same address, ranging from 0x0000 to 0x7FFF. Note that 0x7000~0x71FF are reserved and cannot be used here.
X & Y	:	Left-top coordinate of the widget.
W & H	:	The width and height of the widget.
item0 ~ 15	:	These parameters are used to define different operations from 3 options. Each page is allowed to have only one encoder widget, which means each page may have the most 16 operations through the encoder widget.

Parameter	Data
name	encoder_0
writeAddr	0x2801
X	594
Y	239
W	125
H	58
item0	NULL
item1	NULL
item2	NULL
item3	NULL
item4	NULL
item5	NULL
item6	NULL
item7	NULL
item8	NULL
item9	NULL
item10	NULL
item11	NULL
item12	NULL
item13	NULL
item14	NULL
item15	NULL

Figure 6-54: Encoder Mode-1

An Encoder is usually operated with a set of icons or number widgets. For example, when the knob of an encoder is turned, a preset icon will be shown up for further operations. To implement this function, the encoder and the icon/number widgets have to share the same variable address. When the knob of an encoder is turned, the variable value of the encoder will be changed. Developers may then make the icon to be displayed or disappeared based on the updated value.

There are two approaches to apply Mode-1. See the below sections for more detail.

6.24.1.1. Mode-1: To Control Icons

Approach 1: Using an Encoder to control multiple icon widgets.

- The Encoder and the icon widgets must share the same address
- The minDisplayID and maxDisplayID of an Icon widget must be set to the same.
- Each Icon has different min and max value (must be incremental from 0)
- The number of Icon widgets should be the same as the Item setting of the Encoder widget
- [item] must be set in consecutive order. For example, if item 1 is set, then item 0 must be set too.

Example of Approach1:

Figure 6-55 and Figure 6-56 illustrate the settings for both icon widget and encoder widget:

: Assign the same variable address for both icon and encoder widgets

: minDisplayID and maxDisplayID must be set as the same value for each icon widget.

: Three items for defining the operations represented by three Icon widgets

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x5101
byteLength	2
X	40
Y	37
W	20
H	20
firstIcon	0020_0.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	0
maxDisplayID	0
overlap	Disable

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x5101
byteLength	2
X	37
Y	93
W	20
H	20
firstIcon	0021_1.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	1
maxDisplayID	1
overlap	Disable

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x5101
byteLength	2
X	35
Y	162
W	20
H	20
firstIcon	0022_2.png
lastIcon	
dataFormat	
defaultDisplayID	
minDisplayID	2
maxDisplayID	2
overlap	Disable

Figure 6-55: Icon Settings for Encoder Mode-1 (Approach 1)

Parameter	Data
name	encoder_0
writeAddr	0x5101
X	338
Y	20
W	168
H	62
item0	1,Page0061,0xF...
item1	1,Page0062,0xF...
item2	1,Page0043,0xF...
item3	NULL
item4	NULL
item5	NULL
item6	NULL
item7	NULL

Figure 6-56：Encoder Settings (Approach 1)

Approach 2: An Encoder is used to control one Icon widget

- The Encoder and the icon widget must share the same address
- The icon widget should consist of a number (N) of pictures in a consecutive order (N <= 15)
- The minDisplayID and maxDisplayID should be set to [0 ~ N]
- The item parameter of the Encoder should be set the same as the icon picture amount (N)
- [item] must be set in consecutive order. For example, if item 1 is set, then item 0 must be set too.

Example of Approach 2:

Figure 6-57 illustrates the settings for both icon widget and encoder widget:

: Assign the same variable address for both icon and encoder widgets.

: Three pictures are used in the example. Set firstIcon, lastIcon, minDisplayID and maxDisplayID accordingly.

: Three items for defining the operations represented by the three pictures

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x5501
byteLength	2
X	8
Y	10
W	82
H	108
firstIcon	0000.png
lastIcon	0002.png
dataFormat	
defaultDisplayID	
minDisplayID	0
maxDisplayID	2
overlap	Disable

Parameter	Data
name	encoder_0
writeAddr	0x5501
X	434
Y	12
W	145
H	50
item0	1,Page0042,0xF..
item1	1,Page0061,0xF..
item2	1,,...
item3	NULL
item4	NULL
item5	NULL
item6	NULL
item7	NULL

Figure 6-57: Icon Settings (Left) and Encoder Settings (Right)

6.24.1.2. Mode-1: Sub Functions

Before selecting an item to execute in Mode 1, the encoder rotation can control the increase or decrease of the value in its variable address (writeAddr). At this time, the variables correspond to the item parameter. That is, when the knob is rotated to change the value of the variable to 2, it indicates that the current encoder has reached item 2. At this time, single clicking the encoder indicates selecting and executing the current item 2.

The encoder has three sub functions for each item, namely Click, Rotation, and Combo. Click's function is to change the value of a variable by clicking, Rotation's function is to change the value of a variable by rotating, and Combo's function can be set to single click, double click, and long press. The three sub functions will be introduced later.

Noted that the item address of the encoder cannot be the same as the encoder's writeAddr. Click on the encoder widget, select Mode-1 in the popup window, as shown in Figure 6-54, double-click on item0 of the parameter table (note that the item setting cannot be left blank, for example, if item1 is set, item0 must be set too), and a popup window will be shown as below:

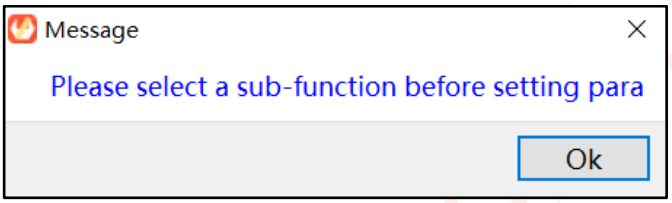


Figure 6-58: Double-click [item0~15]

Clicking OK will bring up the function menu shown in the figure below.

1. Click ❶ to select one of the sub functions (Click, Rotation, Combo).
2. Click ❷ [New] to create a new sub function.
3. Click ❸ to confirm the settings
4. Click ❹ to exit the setting page with saving any changes
5. Click ❺ to delete the selected sub function and settings.

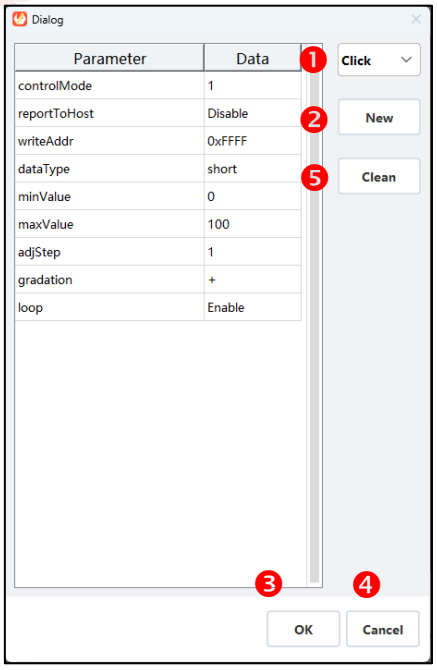


Figure 6-59: [Click] Function Menu

6.24.1.3. Click

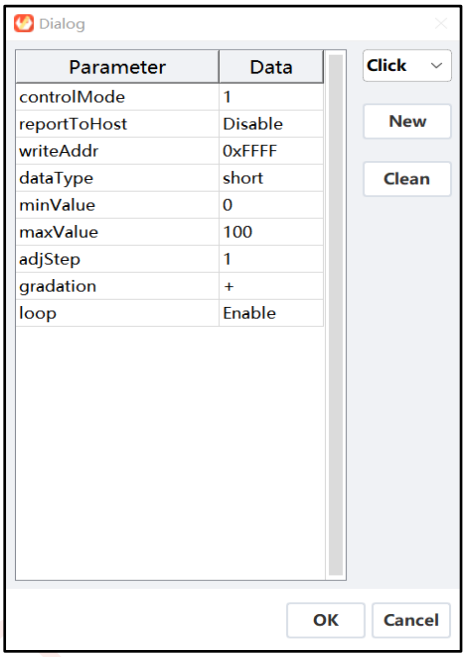


Figure 6-60: Click Mode

Function : Assign variable values to the designated variable address when the knob of the encoder is clicked.

controlMode : Function mode. Click mode = 1.

reportToHost : Set [Enable] to report writeAddr and its value through Uart port, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

writeAddr : Assign the variable address.

dataType : Data type

minValue & maxvalue : Adjustable range, from -32768 ~ +32767

adjStep : The incremental / decrement value of each clicking

gradation : Select incremental (+) or decrement (-) mode

loop : Set [Enable] to reset the variable value when the value reaches the Min/Max value. Set [Disable] otherwise.

6.24.1.4. Rotation

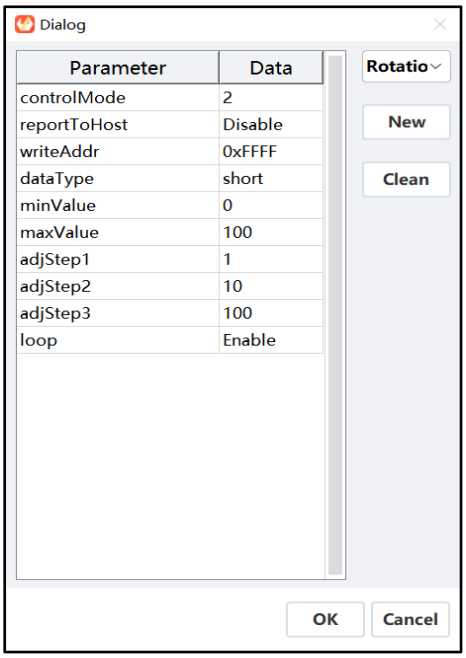


Figure 6-61: Rotation Mode

Function : Assign variable values to designated address by turning the knob of the encoder.

controlMode : Function mode. Rotation mode = 2.

reportToHost : Set [Enable] to report writeAddr and its value through Uart port, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

writeAddr : Assign the variable address.

dataType : Data type

minValue & maxvalue : Adjustable range, from -32768 ~ +32767

adjStep1 ~ 3 : The incremental / decrement values by different rotation speeds

Note: When an item is set to the Rotation sub-function, the encoder's primary function is to alter the value of the designated variable address by rotating the encoder. In addition, there are three step values that will be automatically selected by different rotation speeds. When the encoder rotates slowly, the program chooses adjStep1 as the step adjustment value. When the rotation speed increases, the program automatically switches to adjStep2. If the rotation speed continues to accelerate, the program will automatically select adjStep3 as the

step adjustment value.

6.24.1.5. Combo

Function : Execute the designated operations. The function can be triggered by [Click], [Double Click] and [Long-Press]

controlMode : Function mode. Combo mode = 3.

click : Enable/Disable the click function

double click : Enable/Disable the double click function

long press : Enable/Disable the long press function

singleClickPageGoto : Set the target page to jump to, when [Click].

singleClickReport : Enable/Disable the report function. Refer to *Touch Returned Message* for more detail.

singleClick_Addr : Assign the variable address.

_value : Assign the variable value.

doubleClickPageGoto : Set the target page to jump to, when [Double Click].

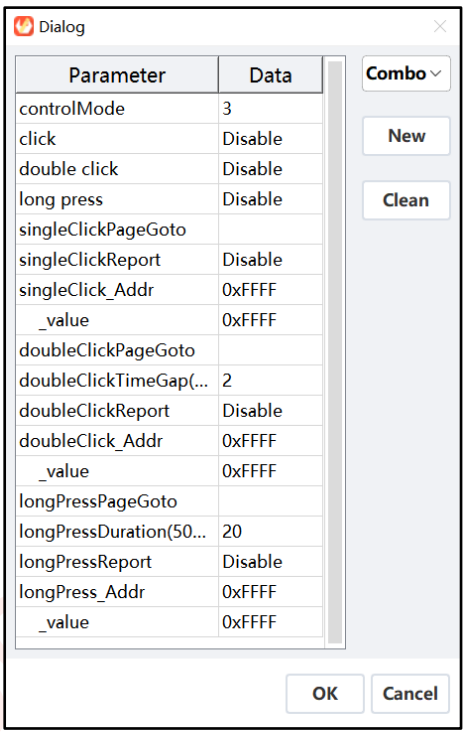


Figure 6-62: Combo Mode

doubleClickTimeGap (50ms) : Set the effective time gap for double click. (50 * N ms, where 1<=N<=255)

doubleClickReport : Enable/Disable the report function. Refer to *Touch Returned Message* for more detail.

doubleClick_Addr : Assign the variable address.

_value : Assign the variable value.

longPressPageGoto : Set the target page to jump to, when [Long-Press]

longPressDuration (50ms) : Set the effective lasting time for long-press operation, (50 * N ms, where 1<=N<=255)

longPressReport : Enable/Disable the report function. Refer to *Touch Returned Message* for more detail.

longPress-Addr : Assign the variable address

_value : Assign the variable value

Note: When an item is set to Combo sub-function, the encoder's primary function is to switch pages and to assign values to the designated variable address by click, double click, and long press operations. Developers may set the [writeAddr] as the specialized register, 0x7014, to assign values to multiple variables. Refer to *Variable Association List* for more detail.

6.24.2. Encoder Mode-2

name : Name of the widget, user-definable.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget.

click : Enable/Disable the click function

double click : Enable/Disable the double click function

long press : Enable/Disable the long press function

Rotation : Enable/Disable the rotation function

ClickPageGoto : Set the target page to jump to, when [Click].

writeAddr : The variable address of the [Click] operation.

_value : The value to be assigned to the writeAddr when the encoder is clicked.

Click_reportToHost : Set [Enable] to report the writeAddr and its value through the Uart port if the encoder is clicked. Refer to *Touch Returned Message* for more detail.

DoubleClick_delay (10ms) : Set the effective time gap for double click. (10ms)

DoubleClick_PageGoto : Set the target page to jump to, when [Double Click]

writeAddr : The variable address of the [Double Click] operation.

_value : The value to be assigned to the writeAddr when the encoder is double-clicked.

DoubleClick_reportTo Host : Set [Enable] to report the writeAddr and its value through the Uart port if the encoder is double-clicked. Refer to *Touch Returned Message* for more detail.

Parameter	Data
name	I-encoder_0
X	761
Y	154
W	61
H	77
click	Disable
double click	Disable
long press	Disable
Rotation	Disable
Click_PageGoto	
writeAddr	0xFFFF
_value	0xFFFF
Click_reportTo...	Disable
DoubleClick_de...	10
DoubleClick_Pa...	
writeAddr	0xFFFF
_value	0xFFFF
DoubleClick_re...	Disable
LongPress_dela...	10
LongPress_Pag...	
writeAddr	0xFFFF
_value	0xFFFF
LongPress_rep...	Disable
writeAddr	0xFFFF
dataType	short
minValue	0
maxValue	100
adjStep	1
adjStep	10
adjStep	100
loop	Disable
Rotation_repor...	Disable

Figure 6-63: Encoder Mode-2

LongPress_delay (10ms) : Set the effective lasting time for long-press operation (10ms).

LongPress_PageGoto : Set the target page to jump to, when [Long Press]

writeAddr : The variable address of the [Long Press] operation.

_value : The value to be assigned to the writeAddr when the encoder is long-pressed.

LongPress_reportTo Host : Set [Enable] to report the writeAdddr and its value through the Uart port if the encoder is long-pressed. Refer to *Touch Returned Message* for more detail.

minValue : The minimum value.

maxValue : The maximum value.

adjStep1~3 : The incremental / decrement values by different rotation speeds

loop : Set [Enable] to reset the variable value when the value reaches the Min/Max value. Set [Disable] otherwise.

Rotation_reportToHost : Set [Enable] to report the writeAdddr and its value through the Uart port if the encoder is rotated. Refer to *Touch Returned Message* for more detail.

Note:

1. The four functions, [Click], [Double Click], [Long Press], and [Rotation], must be enabled befofe use.
2. Developers may set the [writeAddr] as the specialized register, 0x7014, to assign values to multiple variables. Refer to *Variable Association List* for more detail.
3. When the Rotation function is enabled, if the encoder rotates slowly, the program chooses adjStep1 as the step adjustment value. When the rotation speed increases, the program automatically switches to adjStep2. If the rotation speed continues to accelerate, the program will automatically select adjStep3 as the step adjustment value.

6.24.3. Encoder Mode-3

name : Name of the widget, user-definable.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget.

Rotate left to jump page : Set the target page to jump to, when the encoder is rotated to the left.

Rotate right to jump page : Set the target page to jump to, when the encoder is rotated to the right.

reportToHost : Set [Enable] to report the target page number through the Uart port if the encoder is rotated. Refer to *Touch Returned Message* for more detail.

Parameter	Data
name	II-encoder_0
X	622
Y	329
W	79
H	62
Rotate left t...	
Rotate right ...	
reportToHost	Disable

Figure 6-64: Encoder Mode-3

Note:

In the UI project, after selecting Encoder Mode-3 on a certain page, simply fill in the page numbers to jump by left and right rotation in the parameters "Rotate left to jump page" and "Rotate right to jump page," and you can navigate between pages by rotating the encoder left or right.

6.25. Automatic variable

ICON: 

Function : To increase / decrease the data of a designated address.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Automatic Variable: parameterAddr* for more details.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget.

presetAddr : The control address of the widget.

_value : The initial data value of the control address.

loopCode : Set a value to represent [execute in loop] operation. When presetAddr is assigned this value, the widget will be executed in loop.

onceCode : Set a value to represent [execute once] operation. When presetAddr is assigned this value, the widget will be executed once.

stopCode : Set a value to represent [stop] operation. When presetAddr is assigned this value, the widget will stop execution.

stepValue : Set the value of each increment/decrement.

interval (10ms) : The time gap between increment/decrement operations. 10ms per unit. Setting range: 1 ~ 65535

targetAddr : The target variable address. (e.g. the writeAddr of another widget.)

Parameter	Data
name	autoVar_0
parameterAddr	0xFFFF
X	477
Y	39
W	31
H	36
presetAddr	0x6015
_value	0
loopCode	0
onceCode	1
stopCode	2
stepValue	1
interval(10ms)	1
targetAddr	0xFFFF
dataType	short
minValue	0
maxValue	100
gradation	+
reportToHost	Disable
writeAddr	0xFFFF
_value	0xFFFF

Figure 6-65: Automatic Variable

dataType : The data type of the target variable address.

minValue & maxValue : Set the increment/decrement range. Limited by dataType.

gradation : Set [+] to increase the value of the variable when the widget is executed; set [-] to decrease the value of the variable when the widget is executed.

reportToHost : Set [Enable] to report targetAddr and its data value through Uart port after the counting is done, set [Disable] otherwise. Refer to *Touch Returned Message* for more detail.

writeAddr : The address of the operation that will be executed after each counting is done. Developers may set the [writeAddr] as the specialized register, 0x7014, to assign values to multiple variables. Refer to *Variable Association List* for more detail.

_value : The value to be assigned to the address of the operation.

6.26. Needle

ICON: 

Function : For implementing meter/dashboard display.

name : Name of the widget, user-definable.

parameterAddr : Used to update widget parameters through Uart interface. Refer to *Needle: parameterAddr* for more details.

writeAddr : Needle address

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget.

background : Background of the meter/dashboard.

pivot_X : X coordinate of the meter center. The reference point (0, 0) is the left-top coordinate of the widget.

pivot_Y : Y coordinate of the meter center. The reference point (0, 0) is the left-top coordinate of the widget.

startAngle : Start angle. “0” represents the needle points to the 6 o’clock position.

finalAngle : Final angle.

step : Set the distance of each movement of the needle. Only valid when **needleType** is set to Animation or when **swing** is enabled. See *Parameter: step* for more details.

defaultValue : Default angle. The value should be set in the range between startAngle and finalAngle

swing : Swing effect. Refer to *Parameter: swing* for more details.

pivotIcon : Add an Icon to the center of the meter.

needleType : Set needle type. Refer to *Parameter: needleType* for more details.

needle_W : Needle width

needle_L1 : The needle length of the longer side. Refer to *Parameter: needle_L1 & needle_L2* for more details.

needle_C1 : The needle color of the right hand side. Refer to *Parameter: needle_C1 & needle_C2* for more details.

Parameter	Data
name	needle_0
parameterAddr	0xFFFF
writeAddr	0x00B0
X	590
Y	71
W	118
H	202
background	
pivot_X	59
pivot_Y	151
startAngle	0
finalAngle	180
step	5
defaultValue	90
swing	Disable
pivotIcon	
needleType	2D
needle_W	11
needle_L1	120
needle_C1	0x969696
needle_L2	30
needle_C2	0xB4B4B4
needleIcon	
showNumber	Disable
_numberAddr	0xFFFF
_defaultNumber	0
_dataType	short
_promptNum_X	0
_promptNum_Y	0
_firstIcon	
_lastIcon	
_alignment	Left
_leadingZero	Disable
_integerDigit	3
_decimalDigit	0

Figure 6-66: Needle

needle_L2	:	The needle length of the shorter side. Refer to *Parameter: needle_L1 & needle_L2* for more details.
needle_C2	:	The needle color of the left hand side. Refer to *Parameter: needle_C1 & needle_C2* for more details.
needleIcon	:	Add a needle icon. Only required when **needleType** is set to Animation
showNumber	:	Set [Enable] to display Graphics Number. The below parameters are only valid when **showNumber** is enabled.
_numberAddr	:	The address of the Graphics Number
_defaultNumber	:	Default number to be shown.
_dataType	:	Set data type
_promptNum_X	:	Left-top X coordinate of the Graphics Number.
_promptNum_Y	:	Left-top Y coordinate of the Graphics Number.
_firstIcon	:	Select the first icon of the Graphics Number.
_lastIcon	:	Select the last icon of the Graphics Number.
_alignment	:	Set the alignment mode for the Graphics Number.
_leadingZero	:	Set [Enable] to add leading zeros, set [Disable] otherwise.
_integerDigit	:	Set the number of integer digits for the prompt number.
_decimalDigit	:	Set the number of decimal digits for the prompt number.

Note: When a Needle widget is added, a new folder named “Needle” will be added to the project path once the project is compiled. If the **needleType** is set to Animation, then a set of icons (based on the designated **needleIcon** picture) will be generated and saved in the Needle folder. If the needleType is not set to Animation, then the Needle folder will be empty.

6.26.1. Parameter: step

When the **needleType** is set to Animation, UI_Editor-III will generate icons with different angles based on the value of **step**. The number of the generated icons = (finalAngle – startAngle)/step + 1. For example, if startAngle = 0, finalAngle=360, and step= 5, then there will be 73 icons generated, as shown below:

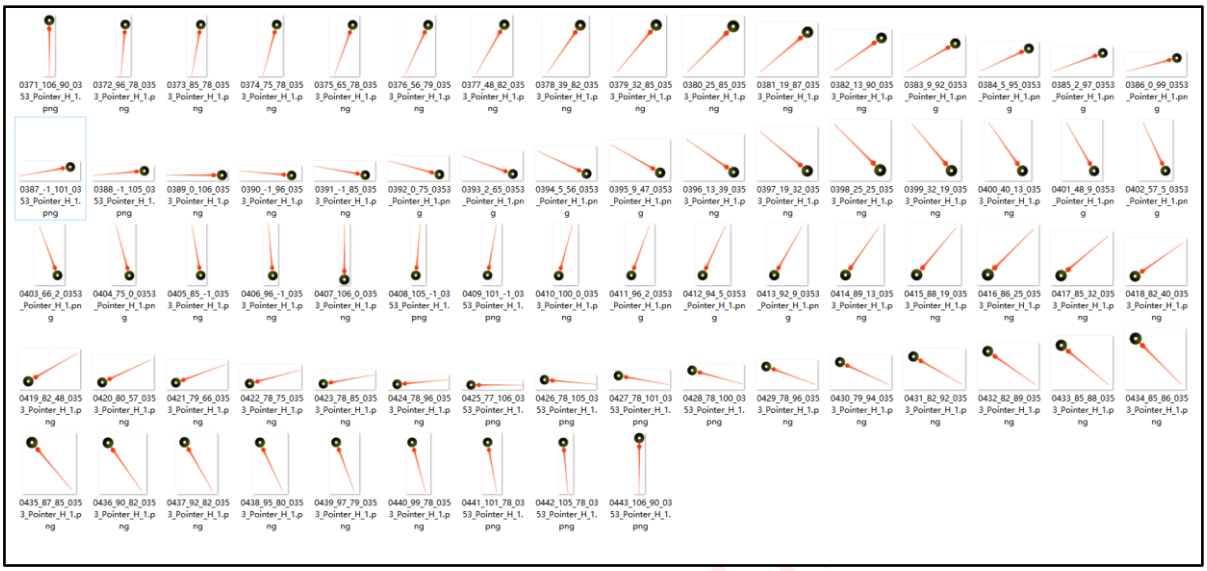


Figure 6-67: Needle icons with different angles

6.26.2. Parameter: swing

When a needle is to be rotated to a designated angle from current position, if the parameter **swing** is disabled, then the needle will directly rotate to the destination without passing by other positions. If the parameter **swing** is enabled, then the needle will pass by the positions on the path till reaching the destination.

For example, startAngle=0°, finalAngle=360°, step=90, and current position is 0°, if a value, 270 is assigned to writeAddr, then

when **swing** is set to [Disable], the needle will rotate from 0° to 270° directly.
when **swing** is set to [Enable], the needle will rotate from 0° to 90° first, then 180°, and finally 270°

6.26.3. Parameter: needleType

There are 4 kinds of needle types, including 2Ddrawing, 2Dsmooth, Animation, and Line. To apply these needle types, the related parameters must be properly set, as explained below:

2Ddrawing & 2Dsmooth: These two needle types are implemented by the drawing engine of UartTFT controllers. **2Ddrawing** does not apply anti-aliasing algorithms, therefore, its display speed is faster than **2Dsmooth**. Although **2Dsmooth** display speed is slower, its needle looks smoother because it applies the internal anti-aliasing algorithm. When **2Ddrawing** or **2Dsmooth** is set, both **needleIcon** and **step** will be invalid; however, **needle_W**, **needle_L1**, **needle_L2**, **needle_C1**, and **needle_C2** should be properly set.

Animation: When **Animation** is set, UI_Editor-II will generate corresponding icons based on the designated icon (**needleIcon**), and **step** setting value. When **Animation** is set, **needle_W**, **needle_C1**, **needle_C2**, and **needle_L2** are invalid; however, **needleIcon** and **needle_L1** should be properly

set. Note that the width (pixel) of the needle icon must be odd.

Line: When **Line** is set, the needle style is a line with round ends. Both **needleIcon** and **needle_C2** are invalid when Line is set; however, **needle_W**, **needle_L1**, **needle_L2**, and **needle_C1** should be properly set.

Icon: When **Icon** is set, the dial plate can only be implemented on the page picture. UI_Editor-II will generate needle icons with different angles based on the needleIcon and step value set by users. The UartTFT controller will display corresponding icons based on the value set in the variable address of the needle widget. Compared to other needle types, using **Icon** type will raise the display speed since there is no need to overlap the dial plate every time. In addition, outside of the needle display area, users may add other display widgets.

As shown in the below figure, the needle is designed to be displayed in the area between the two red circles. Other widgets must NOT be placed in this area. The rest of the area can be used to place other display widgets.

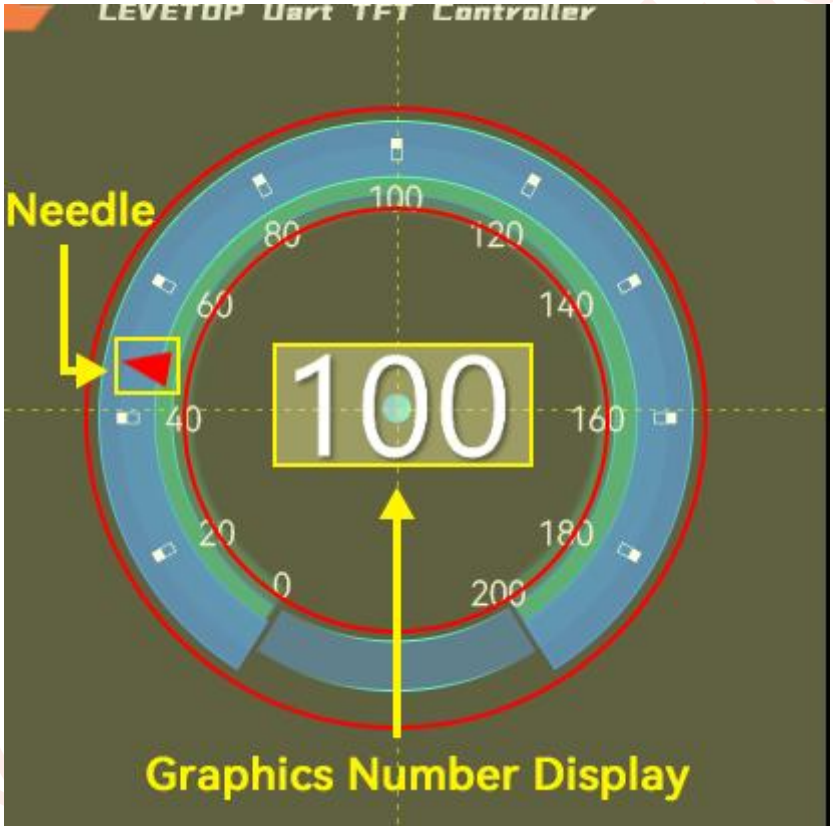


Figure 6-68: Needle – Icon type

Multiple needles can also be implemented, as shown in the below figure. Again, both of the needles' display area (moving path) should not be used to place other widgets.

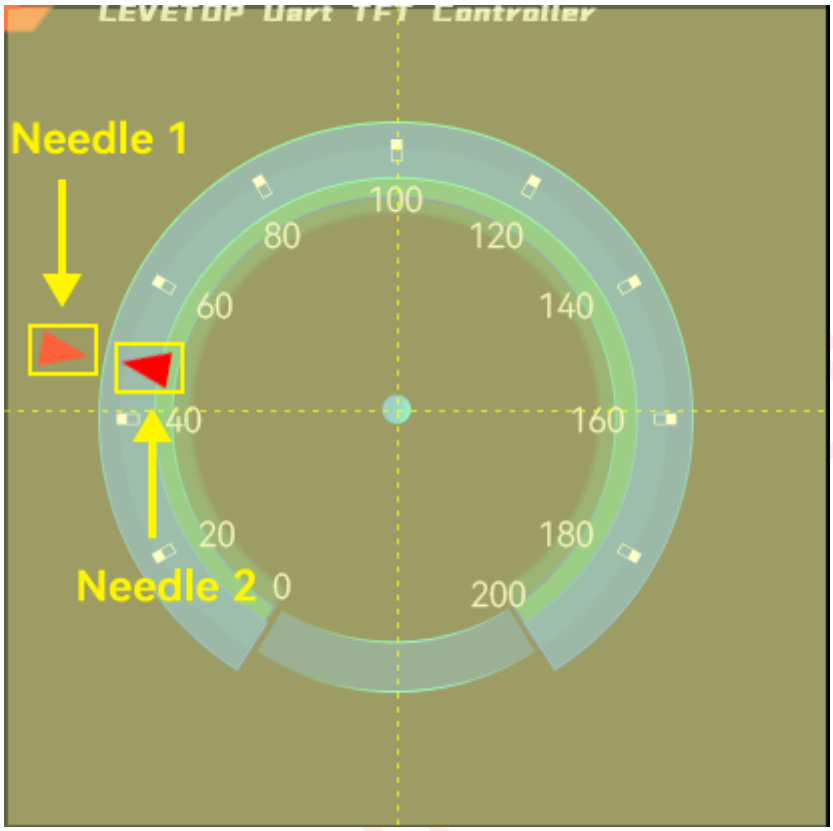


Figure 6-69: Multiple Needles – Icon type

6.26.4. Parameter: needle_C1 & needle_C2

When **2Ddrawing** or **2Dsmooth** is set, the needle color can be set through **needle_C1** and **needle_C2**. As the left picture shown below, when **startAngle**=0, **needle_C1** represents the color on the right, and **needle_C2** represent the color on the left. On the other hand, as the right picture shown below, when **startAngle**=180, **needle_C1** represents the color on the left, and **needle_C2** represents the color on the right.

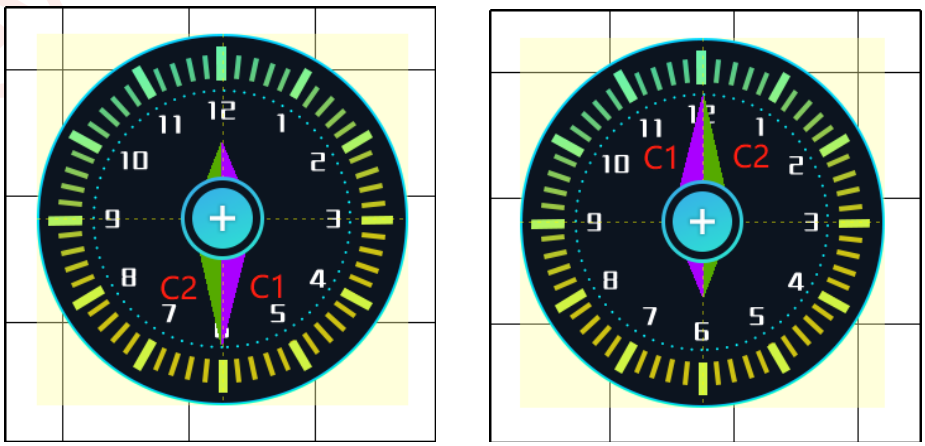


Figure 6-70: Needle Colors

6.26.5. Parameter: needle_L1 & needle_L2

As the figure shown below, **needle_L1** represents the needle length of the longer side, and **needle_L2** represents the needle length of the shorter side.

Figure 6-71: Needle Length

6.27. Cross-Page Menu

ICON: 

Function : The function of the Cross-Page Menu is similar to the SliderMenu widget. However, developers may place various widgets within the menu area.

name : Name of the widget, user-definable.

X & Y : Left-top coordinate of the widget.

W & H : The width and height of the widget. When selecting vertical scrolling, the width of the widget must match the width of the page set by the pageGoto parameter, and the width must be a multiple of 4. When selecting horizontal scrolling, the height of the widget must match the height of the page set by the pageGoto parameter, and the height must be a multiple of 4.

ID : Reserved.

Parameter	Data
name	slidemenuPro_0
X	49
Y	139
W	700
H	286
ID	1
pageGoto	Page0078
direction	Vertical
staticIcon	0764_滑...
slideIcon	0764_滑...
Interval_V	12
Interval_H	5

Figure 6-72: Cross-Page Menu

pageGoto : Set to the menu page that contains the background picture and widgets.

direction : Set the sliding direction.

staticIcon : Set the icon of the slider on the right side of the widget (released state)

slideIcon : Set the icon of the slider on the right side of the widget (sliding state)

Interval_V : The distance of the slider relative to the Y coordinate of the widget.

Interval_H : The distance of the slider relative to the X coordinate on the right side of the widget.

Note:

1. The page set by the pageGoto parameter does not support SlideEffect. This page does not support Circular Touch, SlideMenu, or Trend Graph widgets either.
2. The background picture of the page set by the pageGoto parameter must be BMP/JPG format.
3. The slider on the right side of the widget will only display when the direction parameter is not selected for the loop mode
4. The width and height of the slider pictures (released state and sliding state) need to be consistent.

6.27.1. Example of Cross-Page Menu

Step 1: Add a Cross-Page Menu widget to the page where the sliding menu is needed and set the sliding direction. Set the corresponding width and height, as well as the corresponding menu page, as shown in the following figure

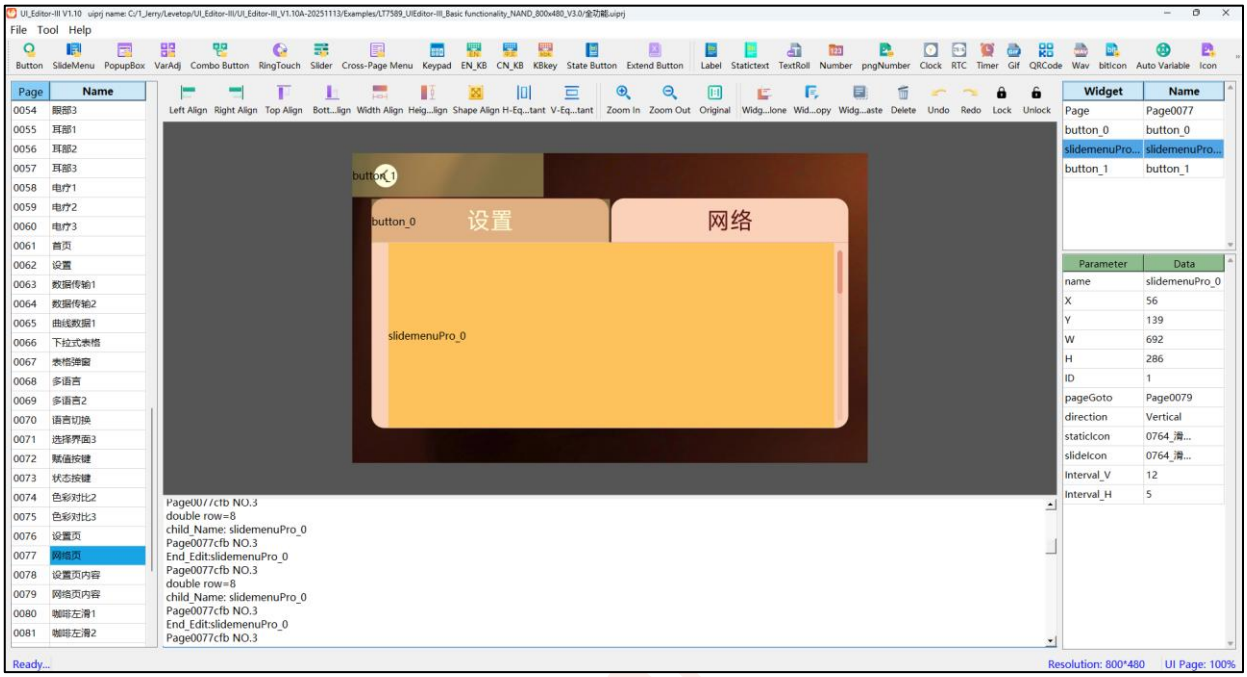


Figure 6-73: Add a Cross-Page Menu widget

Step 2: Place the background picture of the sliding menu and other widgets on the corresponding menu page, as shown in the following figure

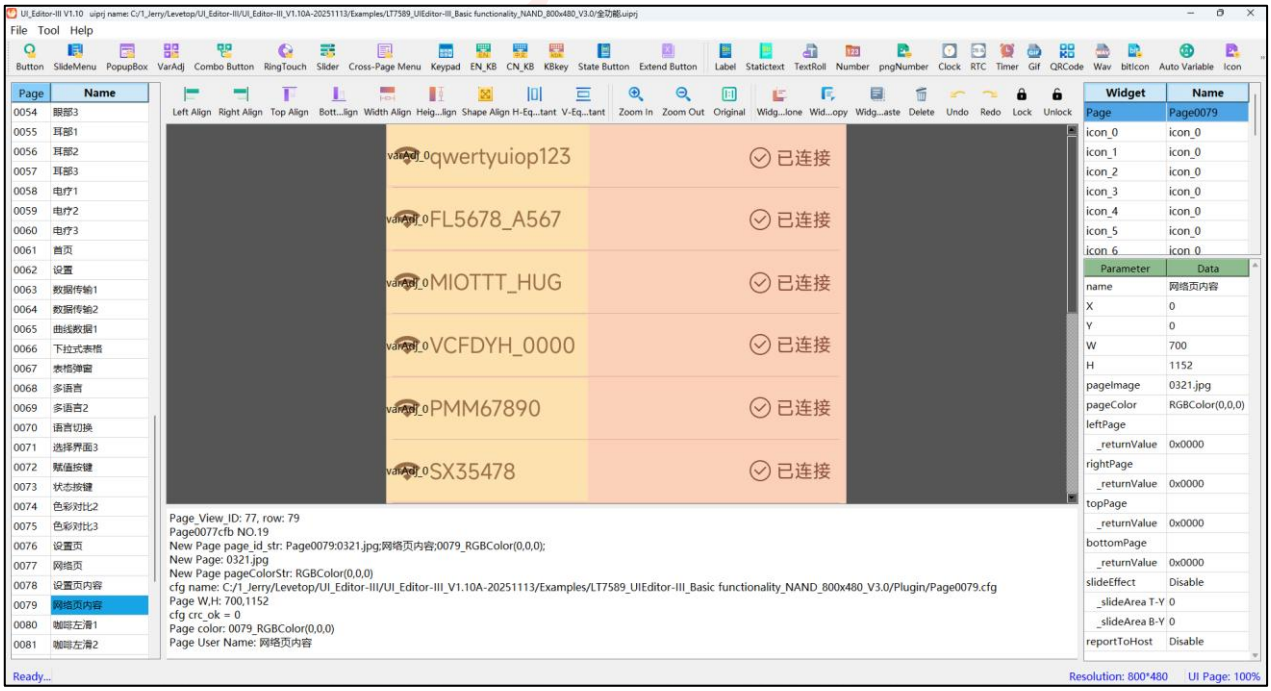


Figure 6-74: Setup the content of the sliding menu

6.28. Layout Widget

Figure 6-75: Layout Widgets

To implement the alignment operations, including Left_Align, Right_Align, Top_Align, Bottom_Align, Width_Align, Height_Align, and Shape consistent, please refer to the below steps:

Step 1: Select an existed widget as a reference widget

Step 2: Click on the desired layout widget, the cursor will be changed to 

Step 3: Press the left button of the mouse on the editing area, and drag to cover the desired widgets

Step 4: Release the left button to execute the alignment operations.

Step 5: Click on the right button of the mouse to exit the operation.

Note:

1. Horizontal and Vertical Equidistance do not need a reference widget.
2. In Step 3, a widget will only be selected when its left-top corner is included. Also, the reference widget is not necessary to be included.
3. Width_Align, Height_Align, and Shape consistent do not apply to those widgets with assigned pictures.

6.28.1. Left_Align

ICON: 

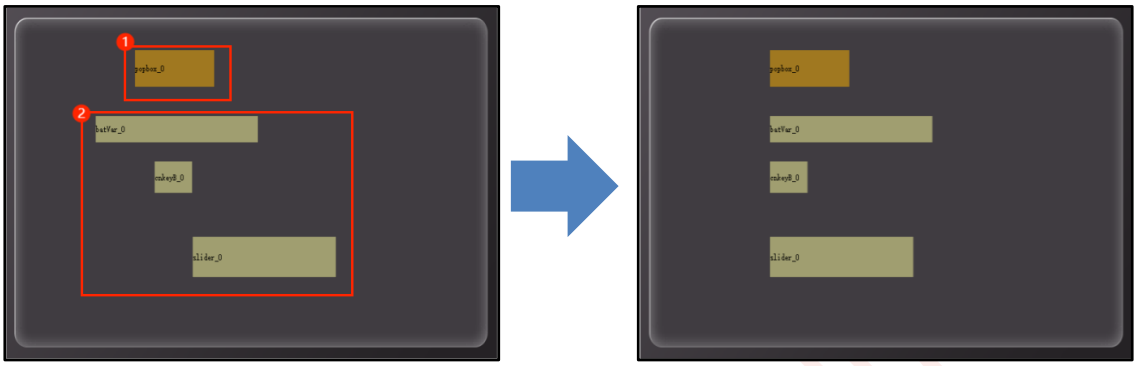


Figure 6-76: Example of Left_Align

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widgets will be left aligned.

6.28.2. Right_Align

ICON: 

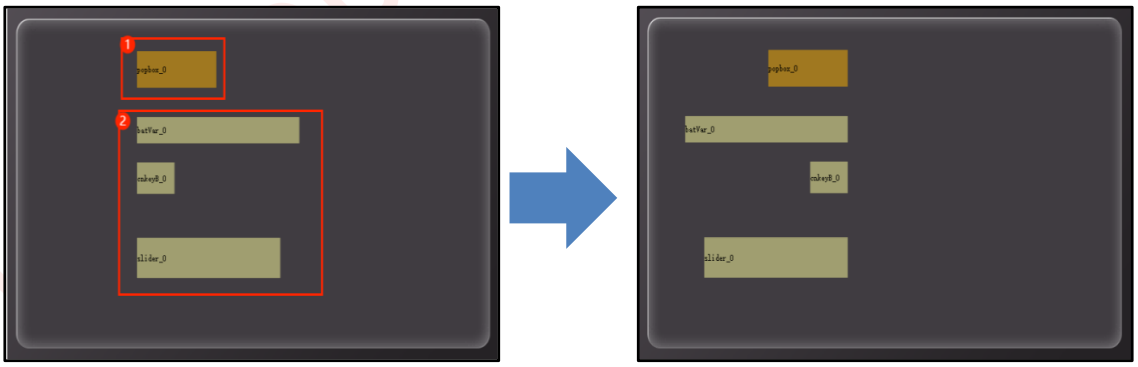


Figure 6-77: Example of Right_Align

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widgets will be right aligned.

6.28.3. Top_Align

ICON: 

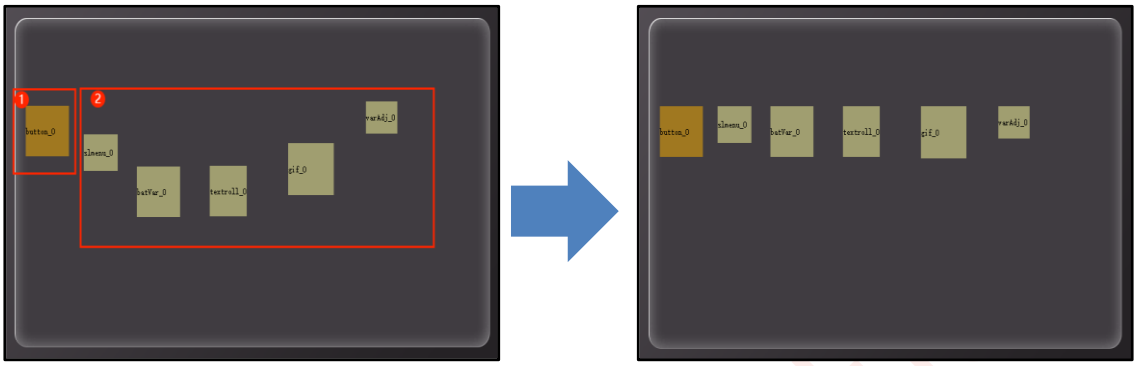


Figure 6-78: Example of Top_Align

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widgets will be top aligned.

6.28.4. Bottom_Align

ICON: 

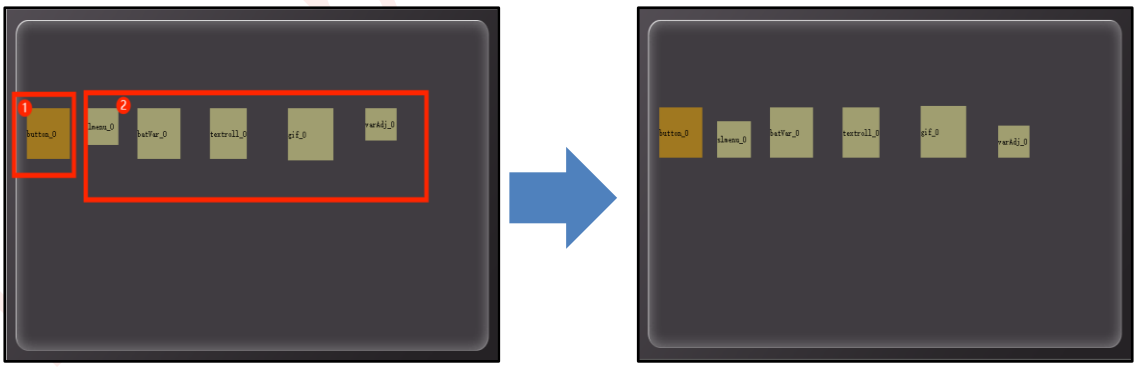


Figure 6-79: Example of Bottom_Align

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widget will be bottom aligned.

6.28.5. Width_Align

ICON: 

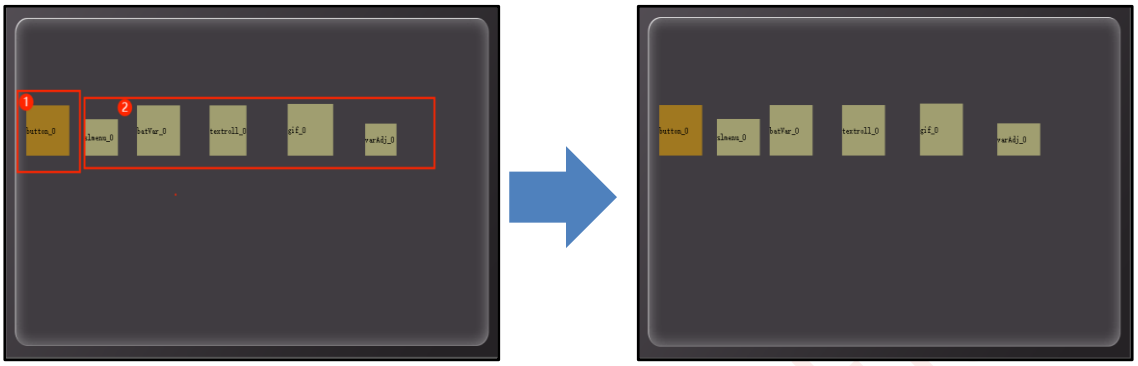


Figure 6-80: Example of Width_Align

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widget will be width aligned.

6.28.6. Height_Align

ICON: 

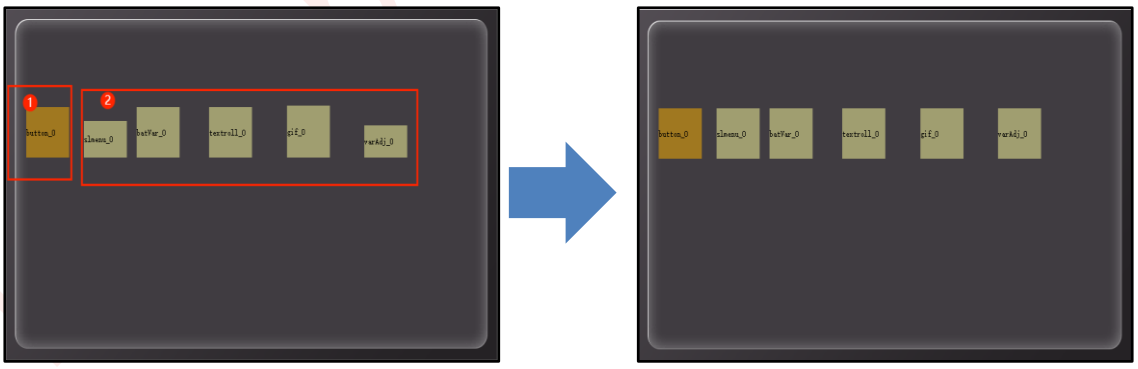


Figure 6-81: Example of Height_Align

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widget will be height aligned.

6.28.7. Shape Consistent

ICON: 

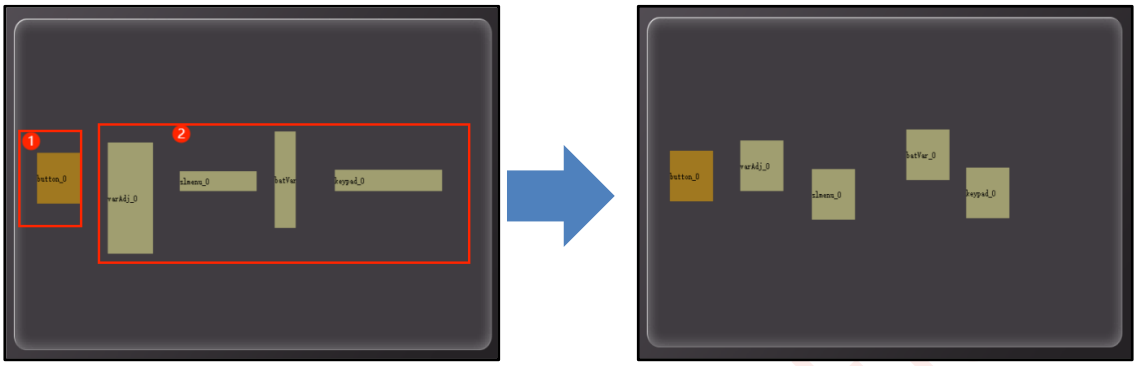


Figure 6-82: Example of Shape Consistent

❶ Select a widget as the reference, and then click on 

❷ Select the widgets that need to be aligned.

When the mouse button is released, the selected widget will be shape aligned.

6.28.8. Horizontal Equidistance

ICON: 

Function: To reallocate the selected widgets in horizontal equidistance.

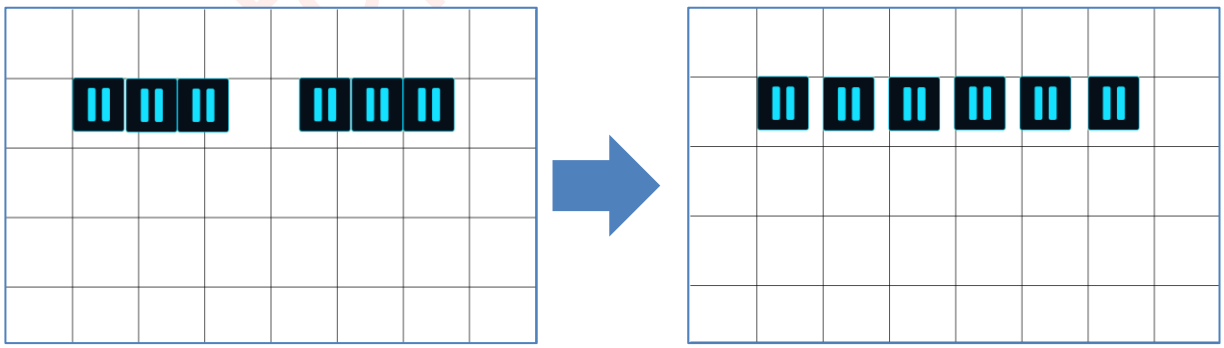


Figure 6-83: Example of Horizontal Equidistance

6.28.9. Vertical Equidistance

ICON: 

Function: To reallocate the selected widgets in vertical equidistance.

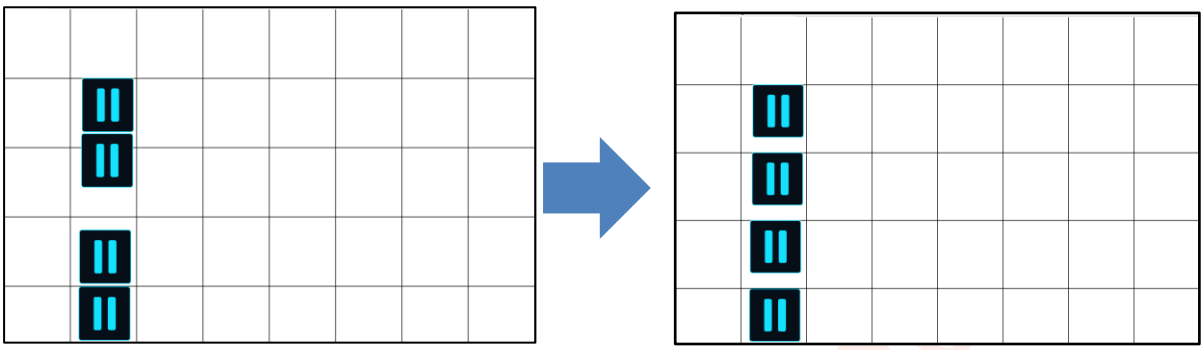


Figure 6-84: Example of Vertical Equidistance

6.28.10. Zoom in & Zoom out

ICON: 

Function: There are 3 widgets, including Zoom in, Zoom out, and Original size (100%). The editing area can be zoom out to 40% of the original size, and zoom in to 300% of the original size. Each click will increase or decrease 20% of the original size. All existed widgets will be adjusted accordingly during the zooming operation. The scaling ratio will be shown on the left-top of the editing area.

Short keys: Ctrl + I → Zoom in; Ctrl + U → Zoom out, Ctrl + Q → 100% size (Original)

Operation examples are as shown in below figures:

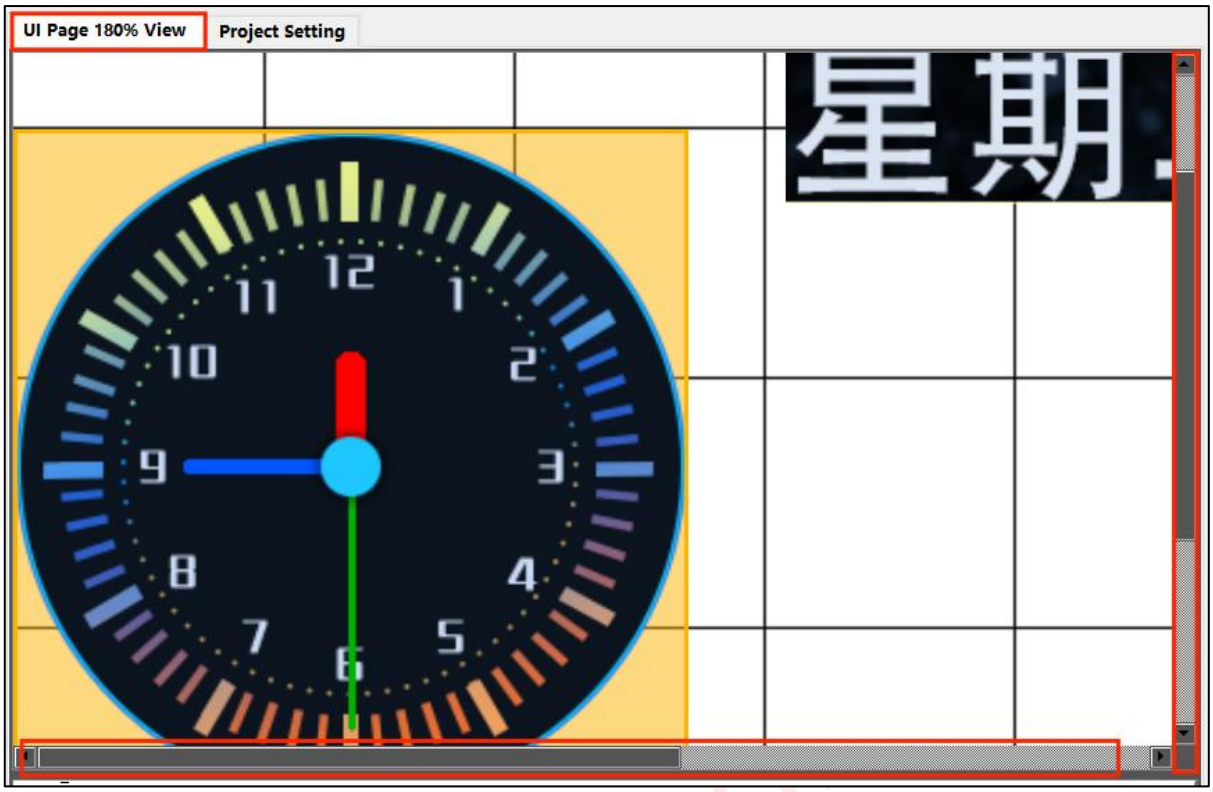


Figure 6-85：Zoom In

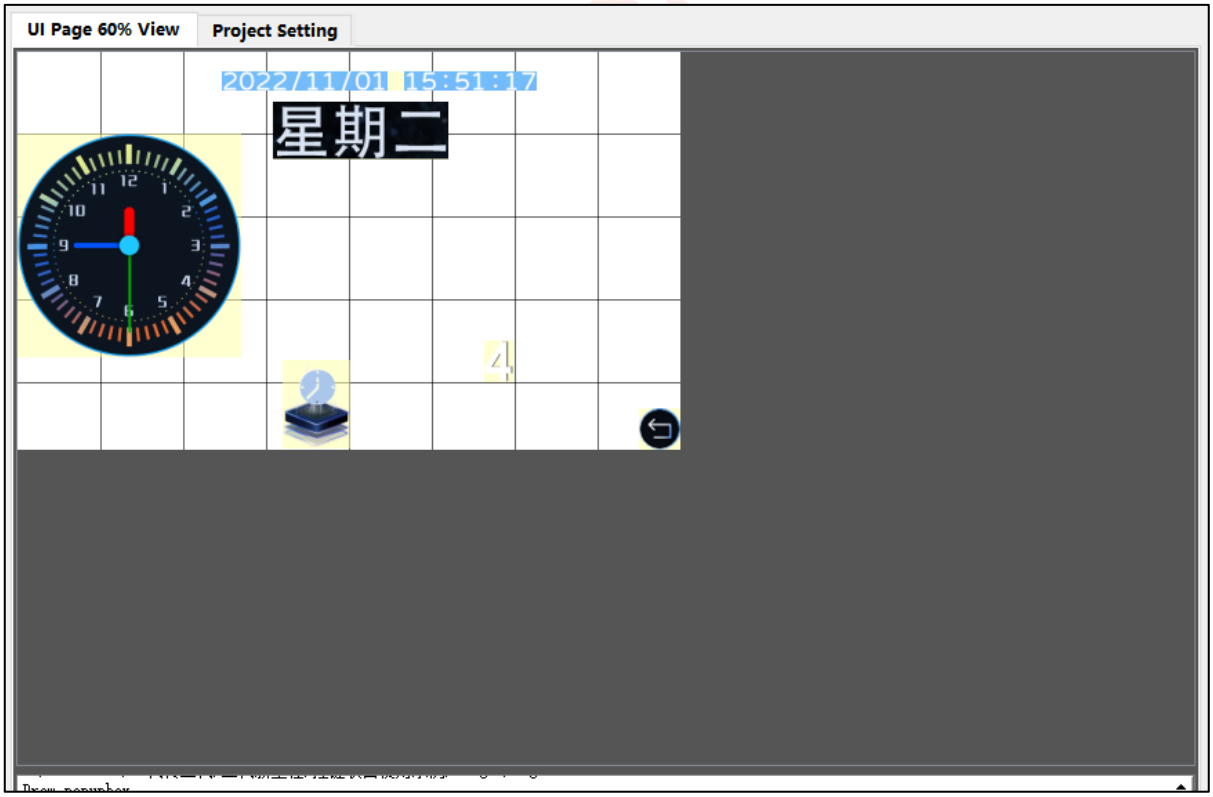


Figure 6-86：Zoom Out

7. Variable Address

7.1. RAM

LT7589 has 64KB RAM and the address range is 0x0000 ~ 0x7FFF. Each RAM address can store 2bytes of data. (**Note:** 0x7000~0x71FF is reserved for special registers.)

In UI_Editor-III, the value assigned to writeAddr or parameterAddr represents the starting address of the data. Since the amount of the data needed for each widget is not fixed, users should carefully plan the RAM address for storing these data, and avoid data overlapping issue.

7.2. writeAddr

As shown in Figure 7-1, a String_Label widget is used as an example. The WriteAddr is assigned a value, 0x1000. In addition, 5 Chinese characters, 乐升半导体, are assigned as the string data. These characters data will be stored by consecutive addresses starting from 0x1000, as illustrated in the below table on the right.

Parameter	Data
name	label_0
parameterAddr	0xFFFF
writeAddr	0x1000
defaultText	乐升半导体

Variable Address	Stored Data	Chinese
0x1000	0xC0D6	乐
0x1001	0xC9FD	升
0x1002	0xB0EB	半
0x1003	0xB5BC	导
0x1004	0xCCE5	体

Figure 7-1: writeAddr vs. Data Storing

Once the data in the above addresses are changed, the display content of the String_Label widget will be changed too. Users my change the data through a touch panel, keyboard widgets or by sending Uart commands. Refer to *Keyboard Widget* and *Uart Communication* for more details.

7.3. parameterAddr

parameterAddr is used to store the first address of the properties of a designated variable / widget. Since both writeAddr and parameterAddr share RAM spaces, users should well allocate the addresses and avoid data overlapping issue. Refer to *Modify Widget Parameter* for more details.

7.4. Special Registers

0x7000~0x71FF are the addresses of specialized registers, as illustrated below. Refer to *Write Data to Control Registers* for more detail.

1. Page Register : Address 0x7000. Developers may send the target page number through Uart interface to display designated page.

2. Backlight Register : Address 0x7001. Developers may write a number between 0 and 63 to change the brightness level. There is total 64 levels.

3. Time Register : Address 0x7002 ~ 0x7007.

Developers may write Year/Month/Day/Hour/Min/Sec to the corresponding registers to setup time and date. The system time and date will not be modified until Confirm_Time Register is written accordingly.

Table 7-1: Time Register

Address	Time	Data Range
0x7002	Year	10~99
0x7003	Month	01~12
0x7004	Day	01~31
0x7005	Hour	00~23
0x7006	Minute	00~59
0x7007	Second	00~59

4. Confirm_Time Register : Address 0x7008. Developers may write the below value to the register to confirm the modification of the time and date. 0: Y/M/D/H/M/S; 1: Y/M/D; 2: Y/M; 3: M/D; 4: H/M/S; 5: H/M; 6: M/S

Updating Time Register through Uart command does not need to write any value to register 7008 to confirm the operation.

Table 7-2: Confirm_Time Register

Address	Write Data	Target
0x7008	0	Year
	1	Month
	2	Dau
	3	Hour
	4	Minute
	5	Second
	6	

5. WAV Control Register	:	Address 0x700A. This register is used to play Wav file. Write 0x0000 to stop playing; write 0x0001 (N) to play the 1st (N) song; write 0x8001 to play the 1st (N) song in loop.
6. Volume Register	:	Address 0x700B. There are 17 level of volume adjustment, ranging from 0 ~ 16. 0: Mute; 16: Maximum volume.
7. RTP Calibration	:	Address 0x700C. Write 0x005A to execute RTP calibration. The register content will be reset to 0 after the calibration is done.
8. Widget Trigger Register	:	Address 0x700D, refer to *Widget Trigger: triggerValue* for more detail.
9. Auto Backlight Control Register	:	Address 0x700E. Same as [Auto Dimming] in Project Setting page 0: Turn off auto backlight control 1: Turn on auto backlight control
10. Register for setting the dimming value	:	Address 0x700F. Same as [Normal(0~63)] in Project Setting page
11. Register for setting the waiting time to enter dimming mode	:	Address 0x7010. Unit: Second. Same as [Hold time(s)] in Project Setting page
12. Register for setting the upgrade mode	:	Address 0x7011, write 0xAA55 to enter Uart upgrade mode. (bootloader required.)
13. Register for Variable Association List	:	Address 0x7014, write the ID number to this register to assign values to multiple variable addresses through the Variable Association List.
14. Multi-Language	:	Address 0x703F, write designated value to switch languages.
15. Switch display direction: Lanscape/Portrait	:	Address 0x7040, write 0 or 1 to switch UI display direction. Refer to
16. Test screen	:	Address 0x7041, write 1 ~ 3 to enter LCD test mode. Refer to *Test Screen Register – 0x7041*
17. Update UartTFT-II_Flash.bin	:	Address 0x7042, update UI through the Uart interface. Refer to *Update UartTFT-V3_Flash.bin by Flash_RW-II*
18. Encoder – Rotate to switch pages	:	Address 0x7043, rotate an encoder to switch displaying pages. Refer to *Encoder: Rotate to Swich Pages*

Note:

1. Item 9, 10, and 11 are the same as the backlight control settings listed in Project Setting page. Users may utilize these registers to control the backlight by Uart commands.

8. Multi-Language

The multi-language function is implemented by switching icons of different languages. Simply write the designated value to 0x703F register, the related icons will then be switched for display. This function is supported by UI_Editor-III_V1.0 version (or above), and designated MCU code.

8.1. Implement Multi-Language Display by Switching Icons

To implement multi-language function by switching icons, developers must
(1) Set the number of the languages used in Project Setting page;
(2) create the icons of different languages;
(3) save the icons to the designated folders.

8.1.1. Create Icon Folders for Multi-Language

In a multi-language project, folders with icons of different languages are added to the original Icon folder, as shown in the Figure 8-1. These added folders are only for multi-language functions, and cannot be designated by other widgets. Also, these folders must be named as 0001 ~ NNNN.

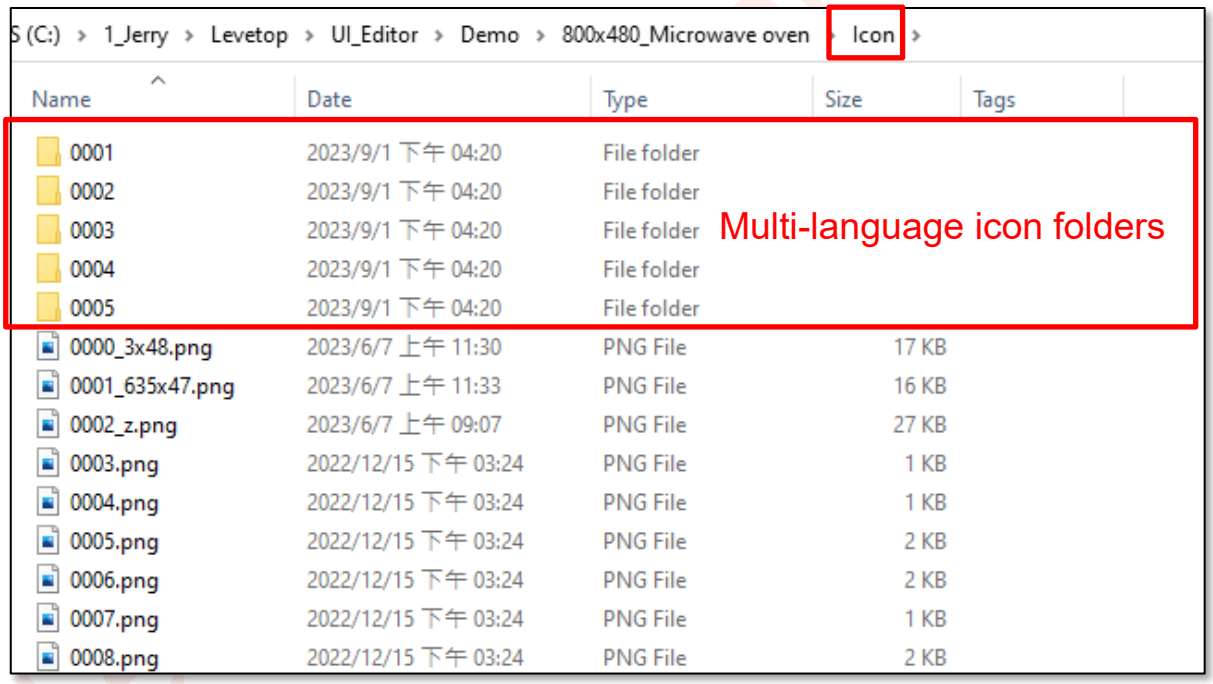


Figure 8-1: Setup Icon Folders for Multi-Language

8.1.2. Icons of different languages

As an example shown in Figure 8-2, the icon, 0000_3x48.png, has to be switched when switching languages. Therefore, the corresponding icons of different languages should be prepared and stored in the folders explained above. The corresponding icons in the folders of different languages must be named in the same number, which is 0000.png in the case here. In addition, the icon resolution and format must be the same.

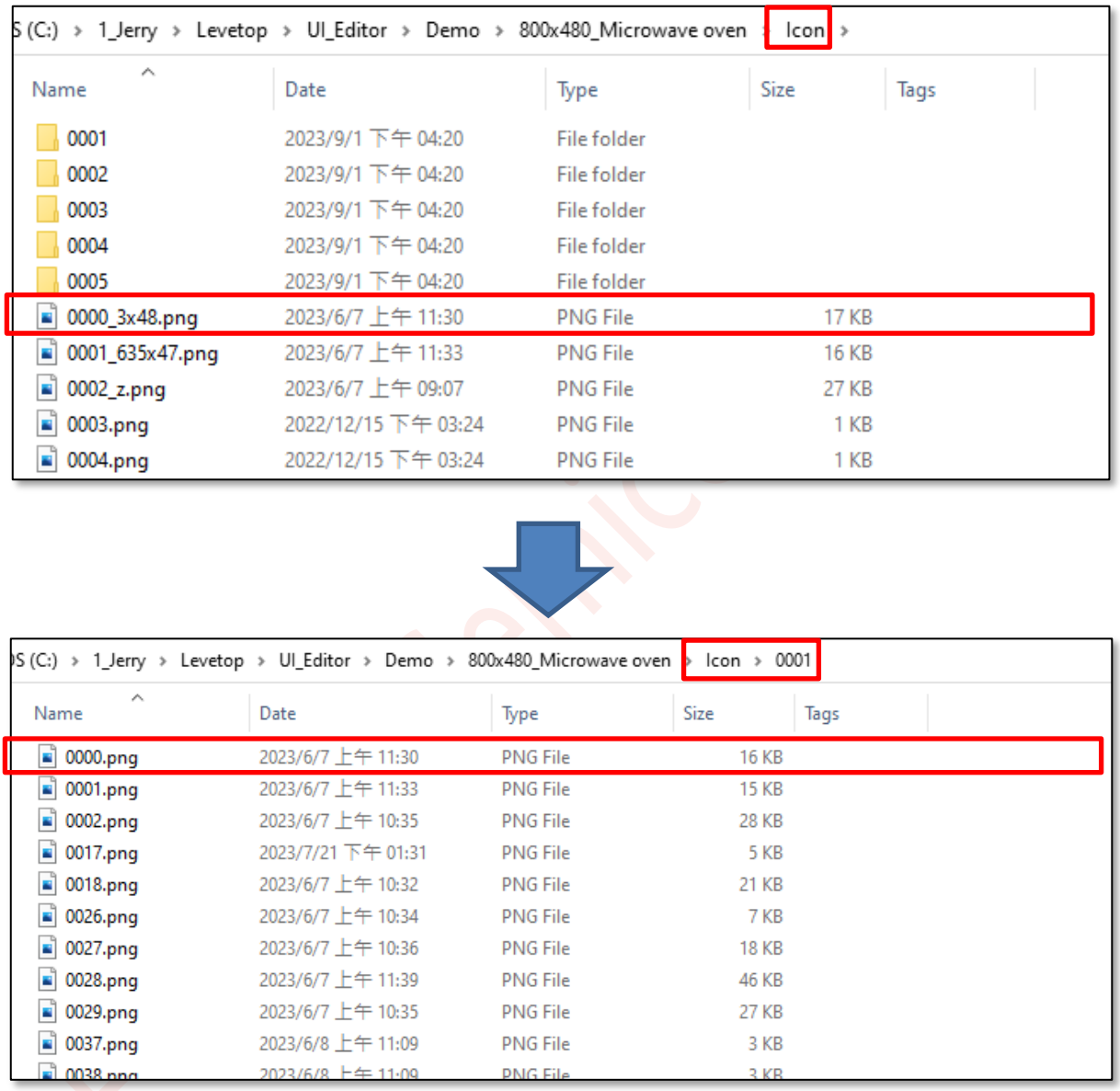


Figure 8-2: Icons of Different Languages

8.1.3. Widgets that support multi-language function

Only those widgets that apply materials in the Icon folder support multi-language function.

8.1.4. Multi-language Switching Process

Suppose the following settings:

0001 folder stores English icons

0002 folder stores Korean icons

To switch the icons,

Write 0x0001 to 0x703F register to switch to English icons.

Write 0x0002 to 0x703F register to switch to Korean icons.

Write 0x0000 to 0x703F register to switch to default language.

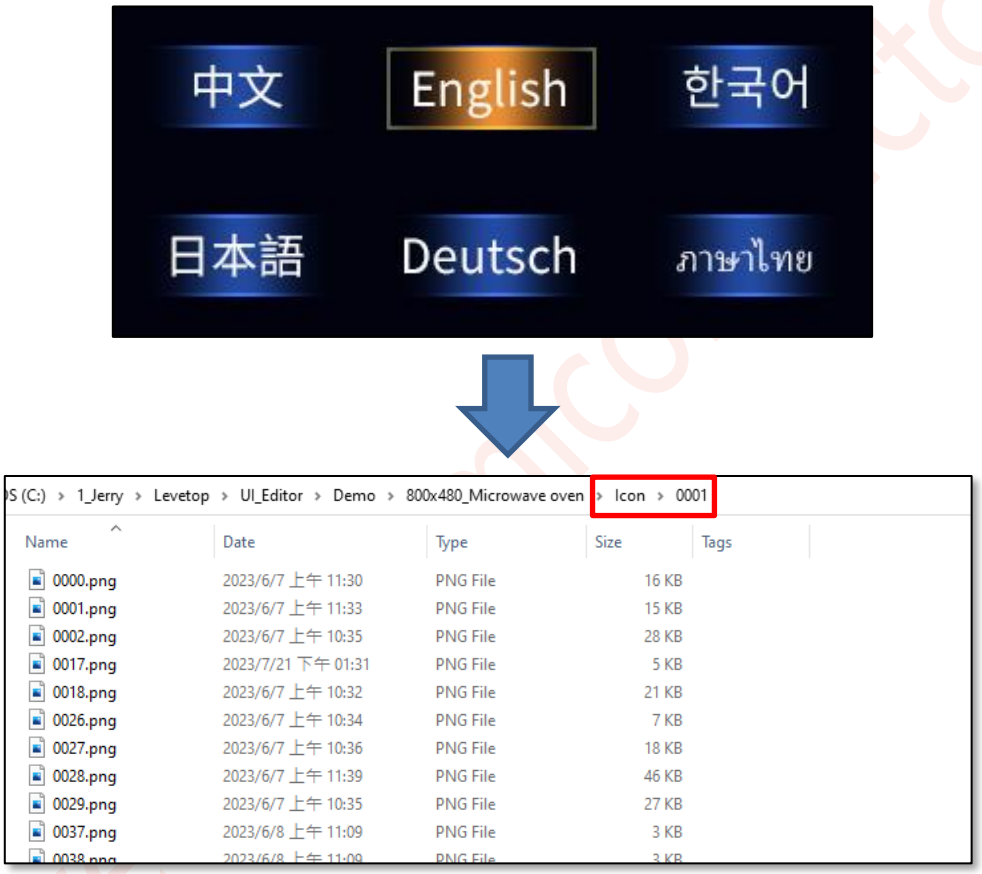


Figure 8-3: Switching to English Icons

8.2. Implement Multi-Language Display by Switching Text Code

To implement multi-language function by switching text code, developers must

(1) Set the number of languages that will be used in Project Setting page;

(2) create the font library in Unicode;

(3) setup String_Label or Text Scroll widgets in desired languages.

8.2.1. Create Font Library in Unicode

Refer to *Font Tool* for creating the desired font library. Note that the code range must cover all desired languages/characters.

8.2.2. Setup for Multi-Language Function

1. Set the “Num of Language” in Project Setting, based on the number of languages that will be used.
2. In String_Label or Text Scroll widgets, enable the parameter, “multiLanguage”, and then click on another parameter, “defaultText”. A window will pop-up as the example shown below:

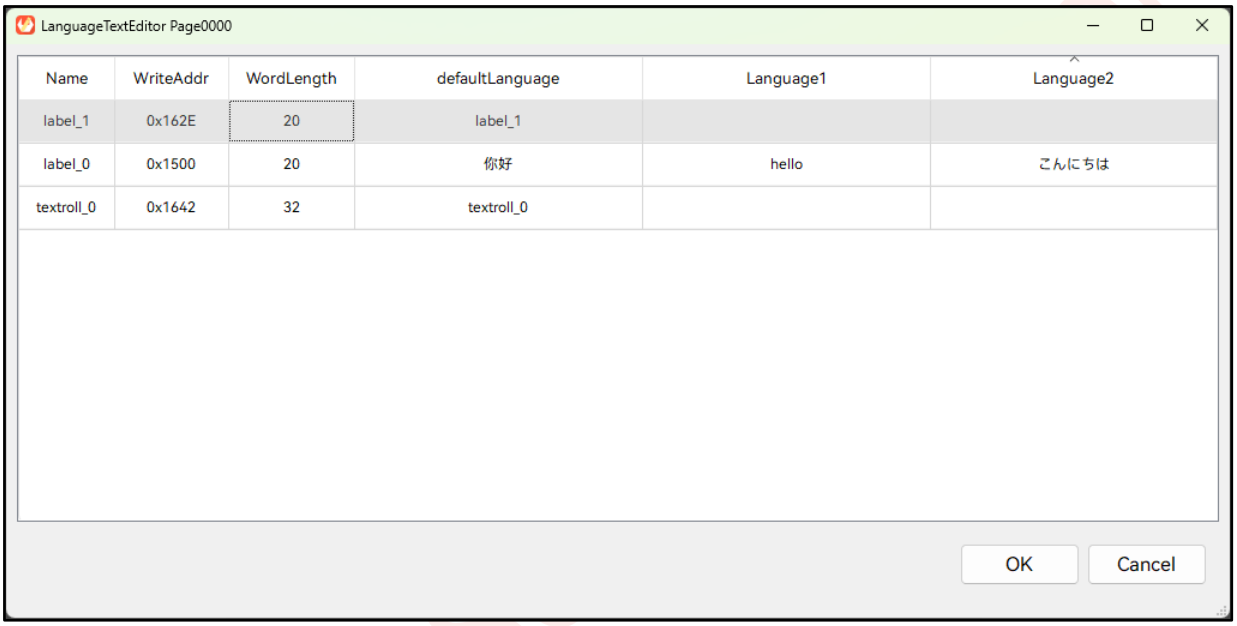


Figure 8-4: Input Multi-Languages

3. Enter the text in the corresponding language to the entry boxes. Note that each character is represented by 2Bytes of data in Unicode. For example, ‘A’ is represented by 0x0041. The maximum string length is 2Kbytes.
4. To preview how the entered characters look like, simply click on the entry box, then the widget will show the display result, as shown below:

Figure 8-5: Preview Entered Characters

8.2.3. Multi-language Switching Process

Suppose the following settings:

0001: English

0002: Korean

To switch languages,

Write 0x0001 to 0x703F register to switch to English.

Write 0x0002 to 0x703F register to switch to Korean.

Write 0x0000 to 0x703F register to switch to default language.

Per the above settings, to switch to English, the command will be: 5A A5 07 10 70 3F 00 01 0E CF

9. Auxiliary Tools

Levetop provides many useful tools for developers to best utilize Uart_Editor-III, as shown in Figure 9-1.

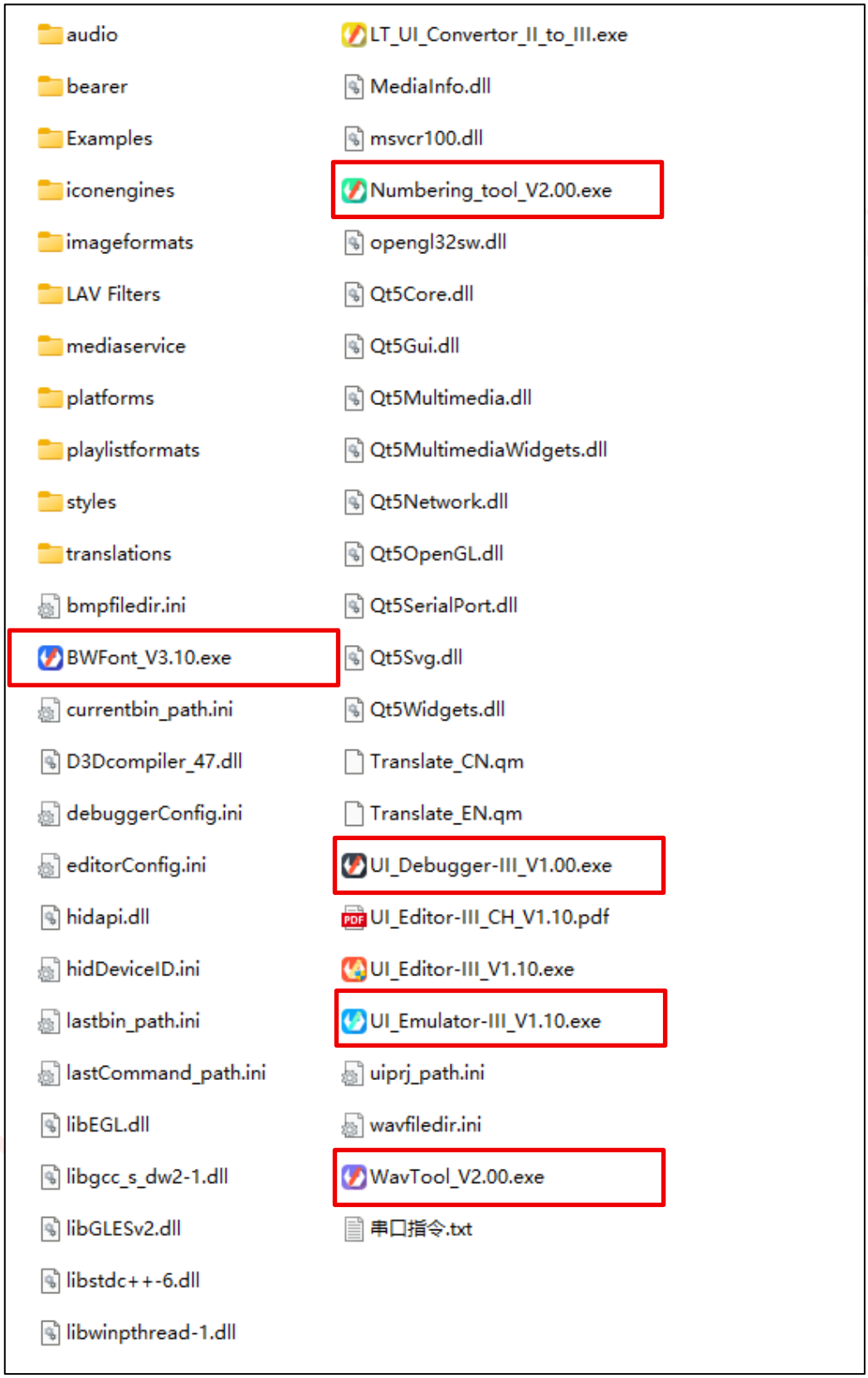


Figure 9-1：Tools for UI_Editor-III

9.1. UI_Emulator-III

9.1.1. Activate UI_Emulator-II

UI_Emulator-III is designed to simulate the working environment of UartTFT panel on a personal computer. Developers may utilize it to easily and quickly check their project design. The emulator is like a standard UartTFT panel. If a project is working well on UI_Emulator, yet does not show the same result on a real board, then the problem may be related to the board itself. A common case is that the clock or timer widget does not work correctly. The possible cause is that there is no RTC circuit or the RTC circuit is not working.

To use UI_Emulator, simply click on the [Tool] menu and then click on [Emulator] to activate the tool. UI_Emulator-III will automatically import UartTFT-V3_Flash.bin to start the emulation. Note UartTFT-V3_Flash.bin will be generated after the UI project is compiled.

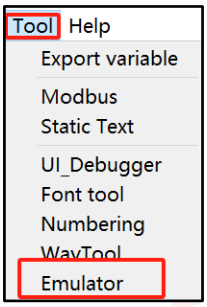


Figure 9-2: Activate UI_Emulator-III (1)

Developers may also double click on UI_Emulator-III_Vx.xx.exe to activate the tool, as shown below:

Figure 9-3: Activate UI_Emulator-III (2)

If the below message is prompted, developers may consider to install the video decoder or not.

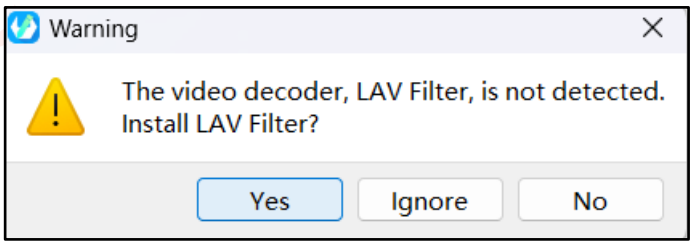


Figure 9-4: Prompt for LAV Filter Installation

The main screen of UI_Emulator-III is as shown below:

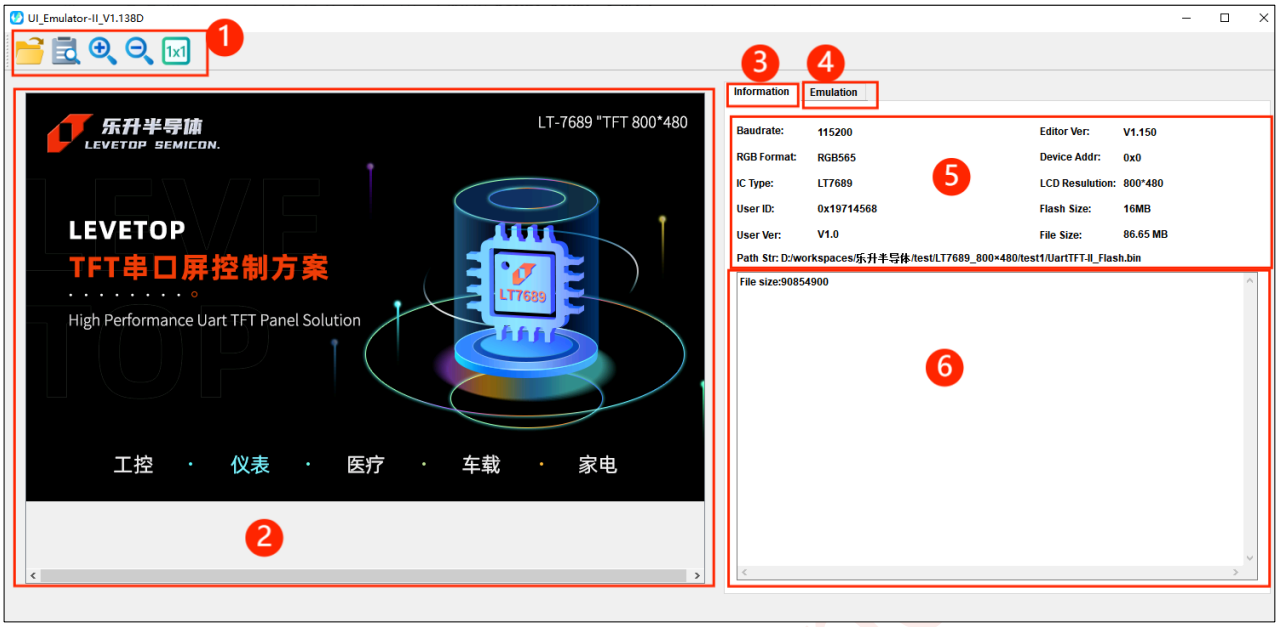


Figure 9-5：UI_Emulator-III Main Screen

❶ **Function bar:** For importing UartTFT-V3_Flash.bin, checking project setting, and zoom in/out the screen.

❷ **Display & operating area:** For checking the display and operation. Developers may click on the display to verify the touch operations.

❸ **Information：** Click on it to check the project information.

❹ **Emulation:** Click to check/verify the variable operations.

❺ **Information area:** The project information is shown here for a quick review

❻ **Operation record:** This box lists the import/operation records.

9.1.2. Variable Operation

Click on [Emulation] to enter the variable operation page:

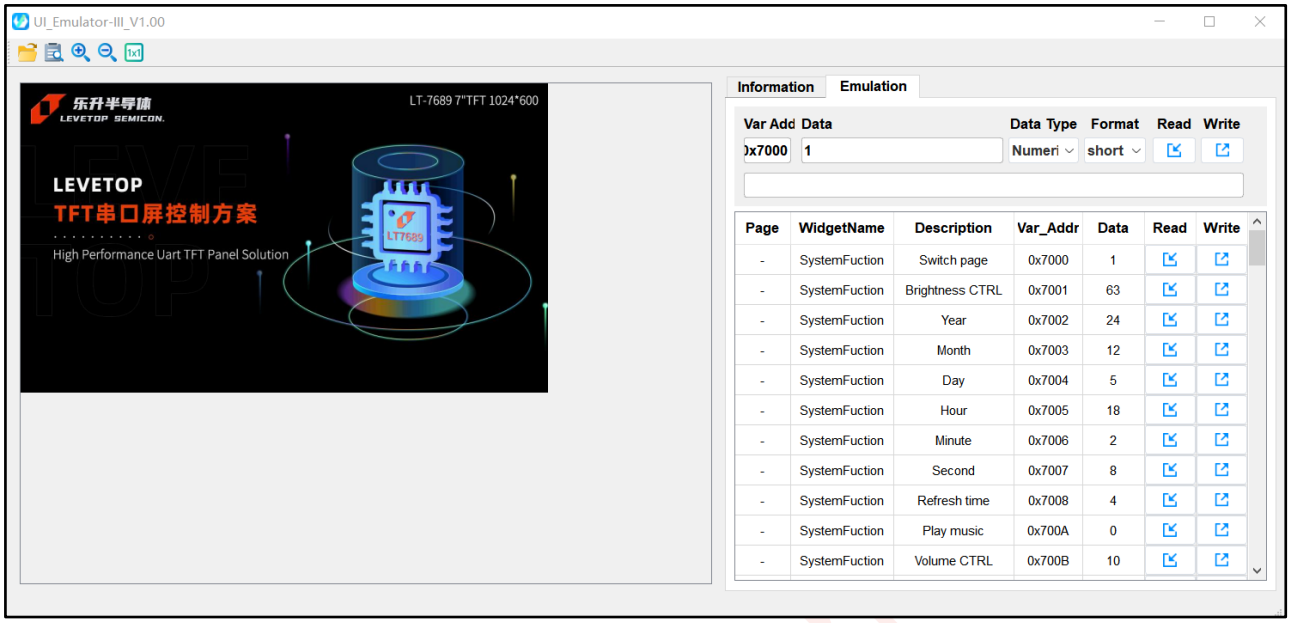


Figure 9-6: Variable Operation Page

9.1.2.1. Setting Bar

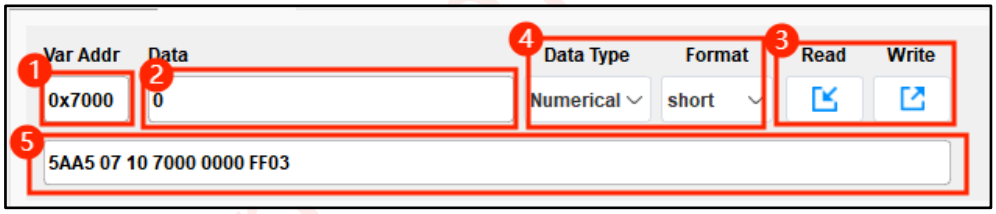


Figure 9-7: Setting Bar

❶ **Var Addr:** Input a variable address

❷ **Data:** Data entry box. Input the data to be sent here. For read operation, the read data will be shown here too. Both decimal and hexadecimal numbers are acceptable. When inputting hexadecimal numbers, 0x must be added in front of the numbers.

❸ **Read/Write**: To trigger a read or write operation. Only allowed to read from / write to one address at a time.

❹ **Data Type & Format:** There are two data types available, **Numerical** and **String**. For **Numerical** type, there are 7 data formats available, as shown in Figure 9-8. For **String** type, there are 5 encoding formats available, as shown in Figure 9-9. Refer to *Sending Data by UI_Emulator-III* for more details.

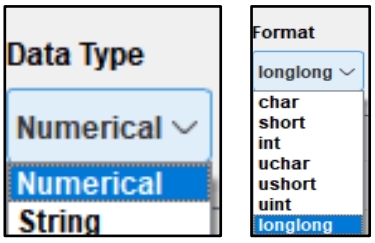


Figure 9-8: Numerical Data Type

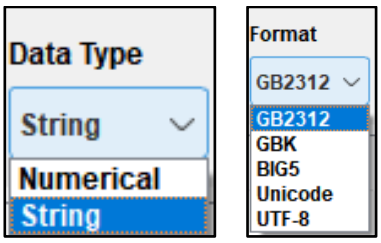


Figure 9-9: String Data Type and Format

❺ **Uart Command Preview:** A Uart command will be generated and displayed in this box, according to the settings above. Developers may utilize UI_Debugger-III to test the command.

9.1.2.2. Address List

The address list includes widgets without touch functions, as shown below:

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

Figure 9-10: Address List

❶ **Page:** The page that the widget located, no modification allowed. Right-click on the page number, a [goto page] button will pop-up. Click on the [goto page] button, the indicated page will be shown in the display area. The columns without page numbers will not respond to the right-click operation.

❷ **WidgetName:** Widget name, no modification allowed. (SystemFunction: Specialized Registers)

❸ **Description:** User-defined name of the widget, no modification allowed.

❹ **Var_Addr:** Widget address, no modification allowed. Double-click on this column, the related information of the widget will be loaded to the setting bar, including Var Addr, Data, Data Type, and, Format。

❺ **Data:** The data of the variable address. Double-click on this column to enter new data. Accept decimal numbers and string characters.

❻ **Read & Write:** Click on  to write the designated data; click on  to read the data of the designated variable address, and show it on the Data column

9.1.3. Write Data to Variable Address

1. Write numeric data to the designated variable address, see below steps:

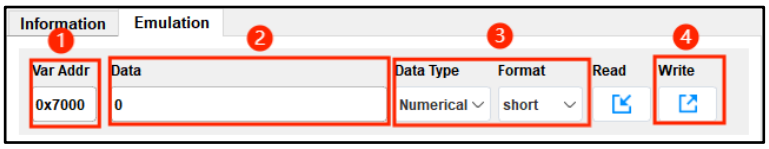


Figure 9-11: Write Numerica Data to Variable Address

❶ Enter the variable address

❷ Enter the data. For hexadecimal number, add 0x in front of the number, e.g. 0x1234.

❸ Select the data format based on the setting of the selected widget. Default setting is ushort.

❹ Click on  to write the data to the designated variable address.

Note: To enter numbers with decimal digits, the entered value must follow the widget settings. For example, if the widget is set 3 integer digits and 2 decimal digits, to display 123.45, the entered data must be 12345 (the decimal point cannot be added).

2. Write string data to the designated variable address, see below steps:

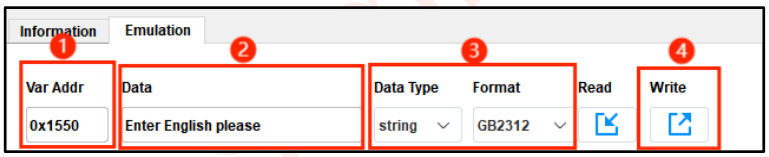


Figure 9-12: Write String Data to Variable Address

❶ Enter the variable address

❷ Enter the string data.

❸ Select the encoding format based on the used font.

❹ Click on  to write the string to the designated variable address

9.1.4. Simulate Encoder Operations

Developers may apply the following keyboard to simulate the encoder operations. The emulation is only valid to the encoder in the current page

Direction Key (Left): Encoder is rotated counterclockwise, and the data value is decreased.

Direction Key (Right): Encoder is rotated clockwise, and the data value is increased.

Numeric Key 1: Click on the encoder

Numeric Key 2: Double-click on the encoder.

Numeric Key 3: Long-pressed on the encoder.

9.1.5. For Projects with Rotated Display

Since UI_Emulator-III does not support rotated display, to emulate projects of rotated display, the project must be reset to 0° angle, and the resolution should be modified accordingly. Finally, the project has to be compiled to generate a new UartTFT-V3_Flash.bin to be loaded by UI_Emulator-III. As shown below:

1、Set the angle back to **0 Degree**

2、If the original project rotates 90° or 270°, then the **X-Pixel** and **Y-Pixel** resolution settings must be switched. (If the original project is 0° or 180°, then no need to change the resolution settings.)

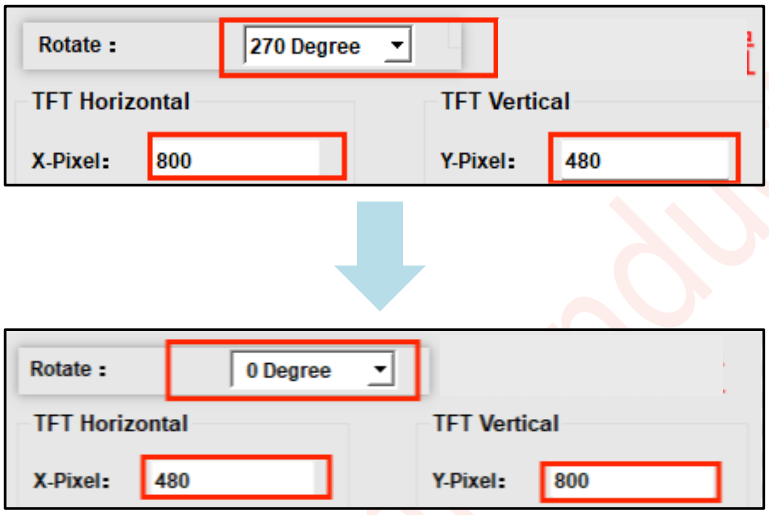


Figure 9-13: Setup the rotation angle and the resolution

9.1.6. Limitations of UI_Emulator-II

1、Trend graph display by sending data – not supported.

2、Key with beep – not supported.

3、LAV Filter is required for playing video.

4、Control by external Uart commands – not supported.

9.1.7. Sending Data by UI_Emulator-III

Table 9-1: Sending Data by UI_Emulator-III

Widget Name	Bytes	Data Type	Widget Name	Bytes	Data Type
Button	-	-	Analog Clock	-	-
SlideMenu	2	-	Digital Clock	-	-
Popupbox	-	-	Gif	2	ushort
Variable Button	Same as dataType setting	-	QRCode	(WordNumber+1)*2	String
Combo Button	-	-	Audio Play	-	-
Circular Touch	2	-	Slider Bar	2	short
SingleKey	-	-	Bit Status	2	ushort
Numeric Keypad	Same as dataType setting	-	Icon	2	ushort
EN_Keyboard	(wordLength+1)*2	-	Trend Graph	-	-
CN_Keyboard	(wordLength+1)*2	-	Encoder	2	-
String_Label	(wordLength+1)*2	String	Timer	2*3 (3 variables)	ushort
Text Scroll	(wordLength+1)*2	String	Static_Text	-	ushort
Text Number Display	Same as dataType setting	Refer to widget	Automatic Variable	Same as dataType setting	Refer to widget
Graphics Number Display	Same as dataType setting	Refer to widget	Extern Button	-	-
State Button	-	-			

Note: " - " sign means "no such option" or "not available".

9.2. UI_Debugger-III

UI_Debugger-III is designed to debug the project on a development board through Uart interface. To activate the tool, simply click on the [Tool] menu and then click on [UI_Debugger], as shown below:

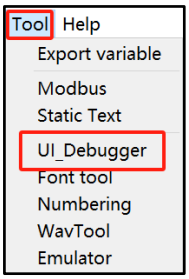


Figure 9-14: Activate UI_Debugger-III

9.2.1. Main Screen

The main screen of UI_Debugger-III is as shown below:

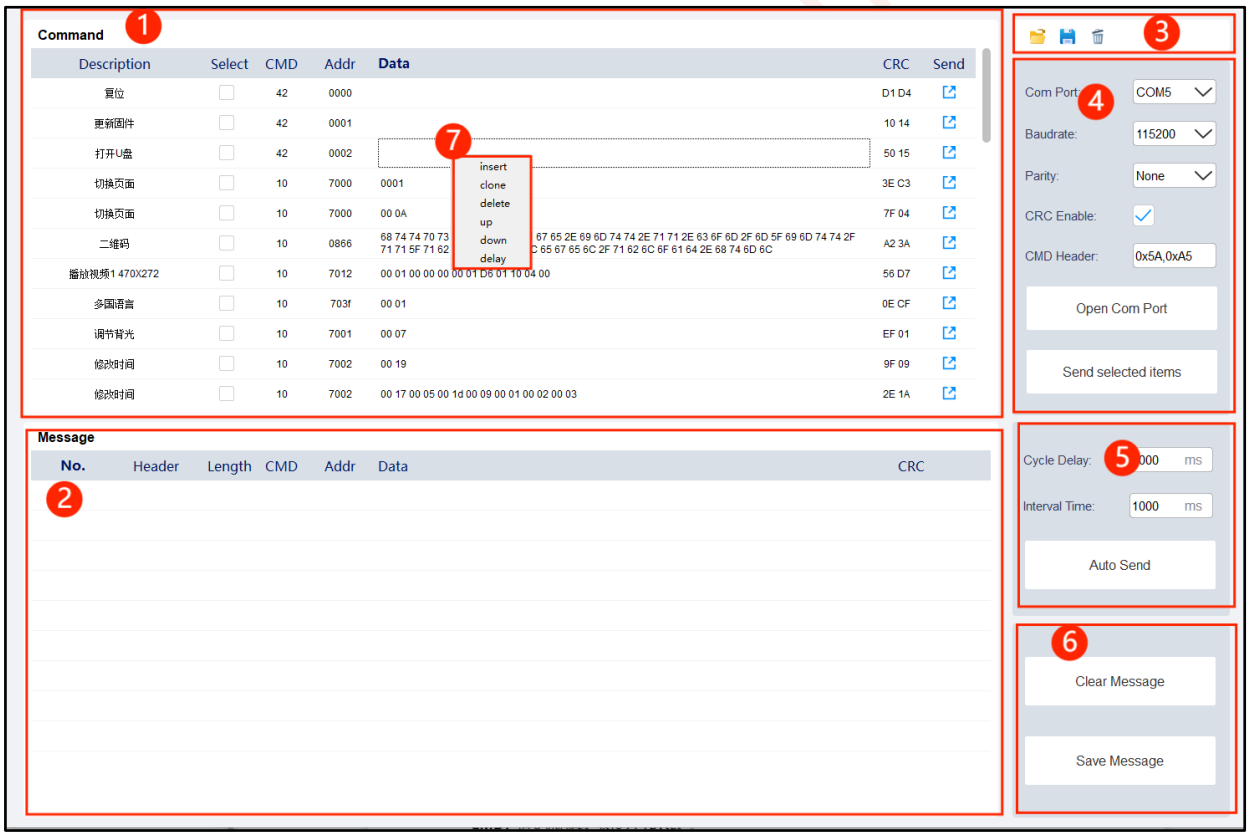


Figure 9-15: UI_Debugger-II Main Screen

❶ Command Edit Area

Description: Name of the command, user-definable.

Select: Check the box to select the command for further operation, such as “Send selected items”.

CMD: Command type. 10: Write; 03: Read; 42: Others. Data length: 1Byte

Addr: Target variable address. Data length: 2Bytes

Data: Data to be written / Data amount to be read. Data length: 2*n Bytes, where n = number of data.

CRC: Cyclic Redundancy Check. Data length: 2Bytes (Auto-generated, based on CMD, Addr, and Data)

Send: Click to send the command

❷ **Message area:** Prompt messages will be listed in this area.
Black: command sent; Blue: returned message.

No.: Message index

Header: Header of the command/returned message. Data length: 2Bytes

Length: Command/returned message. Data length: 2Bytes

CMD: Command type. Data length: 1Byte

Addr: Target variable address

Data: Data to be written / Data amount to be read. Data length: 2*n Bytes, where n = amount of data.
Refer to *Uart Communication* for more detail about command format.

CRC: Cyclic Redundancy Check. Data length: 2Bytes

❸ **Function bar**

 : Load a command list (txt format)

 : Save a command list (txt format)

 : Clear all commands in Command Edit Area

❹ **Configuration**

Com Port: Select the com port connected with the development board.

Baudrate: Set the baud rate. This value must be the same as the project setting. User-defined baudrate is available. Developers may select “custom” to define their own baudrate, as shown below:

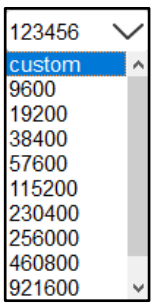


Figure 9-16: User-defined Baudrate

Parity: Set parity check. The setting must be the same as the project setting.

CRC Enable: Check to enable CRC. The setting must be the same as the project setting.

CMD Header: Command header. The setting must be the same as the User Start Bytes of the project setting.

Open Com Port: Open the selected COM port to connect with the development board.

Send selected items: Send the selected commands in the Command Edit Area in order.

❺ **Configuration Area – for sending selected commands in loop**

Cycle Delay: Time gap between each loop.

Interval Time: Time gap between each command.

Auto Send: Click to send the selected commands.

❻ **Operations for Message area**

Clear Message: Clear the messages in the Message area

Save Message: Save the messages to a txt file

❼ **Popup Menu –** Right click on a command line in the Command Edit Area

insert: Insert a blank command line above the selected one.

clone: Clone the selected command line and paste it to the line below the selected one.

delete: Delete the selected command line.

up: Move up the selected command line.

down: Move down the selected command line.

delay: Add a delay command above the selected command line. (unit: ms)

❽ **Popup Menu –** Right click on the **Select** column to choose [select all] or [select none], as shown below:

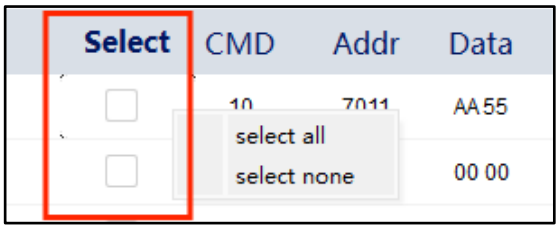


Figure 9-17: Popup Menu for Select Function

9.2.2. Tutorial – Send Commands

1、Send a single command:

(1) Setup the configuration, and then click on [Open Com Port], as the example shown below:

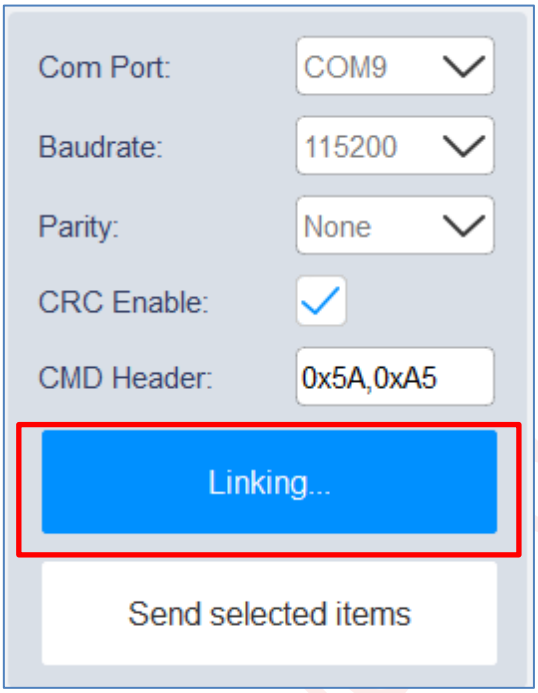


Figure 9-18: Setup the configuration

(2) Add a command: Double click on a command line to edit. Developers may also load an existed command file by clicking on  in the Function bar.

(3) Send a command: Click on [Send] column of the selected command line to send the command.

Description	Select	CMD	Addr	Data	CRC	Send
Switch Page	☑	10	7000	00 00	FF 03	
Send Data to 0901	☐	10	0901	00 20	B6 47	
Read 0901 & 0902	☐	10	0901	00 02	36 5E	
Adjust Backlight	☐	10	7001	00 2D	6E DE	
Send data to Curve 1	☐	10	0200	31 2E B1 ED B8 F1 B2 E2 CA D4 D6 D0 00 00 00 00 00 00 00 00 00 30 30 3...	FE 45	
Send data to Curve 2	☐	10	0250	36 2E BA BA D7 D6 D7 D6 B7 FB BC AF 00 00 00 00 00 00 00 00 00 35 35 3...	AD 14	
Daley(ms)	☐	++		300		
Clear Curve 1 & 2	☐	10	E003		79 C4	

Figure 9-19: Send a command

(4) Check the Message area for the sending and receiving messages.

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

Figure 9-20：Message Area

2、Send selected commands & Send commands in loop

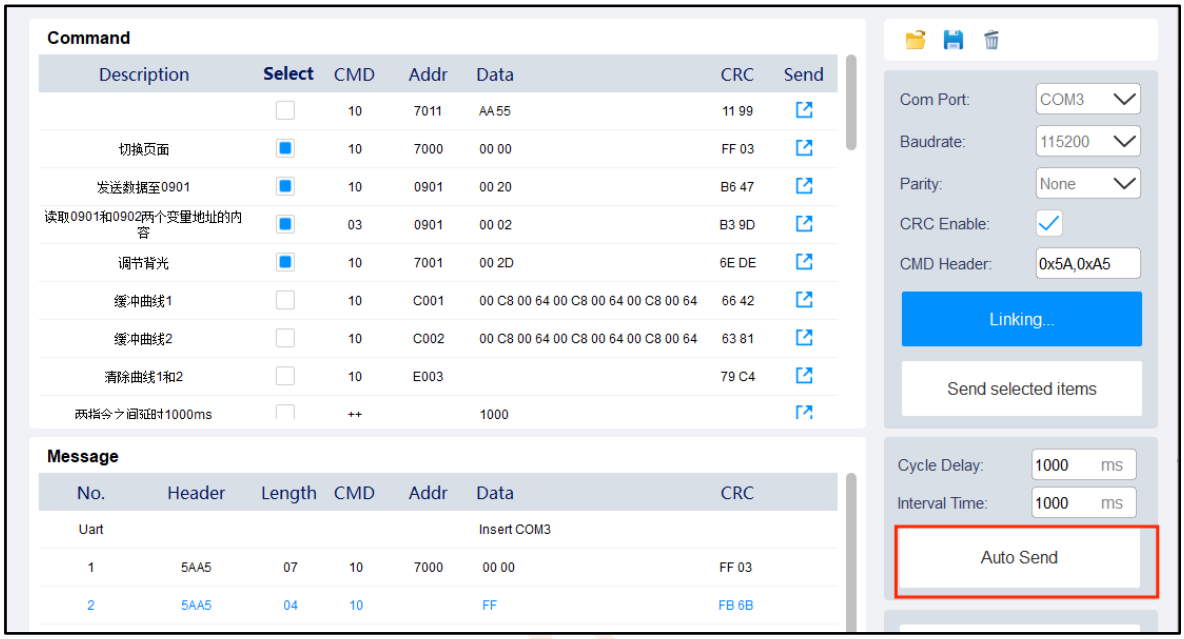


Figure 9-21：Send Multiple Commands

(1) Adjust the command order if needed. (Commands are sent from up to bottom)

(2) Select the commands to be sent by clicking on the [Select] column.

(3) Click on [Send selected items] to send selected commands, or click on [Auto Send] to send selected commands in loop

(4) For [Auto Send] function, adjust [Cycle Delay] and [Interval Time] if needed.

(5) Developers may also add user-defined delay commands, as shown in Figure 9-22 and Figure 9-23, where “++” is fixed, and cannot be modified. Add delay time in [Data] column. Click on the [Select] column to activate the delay command.

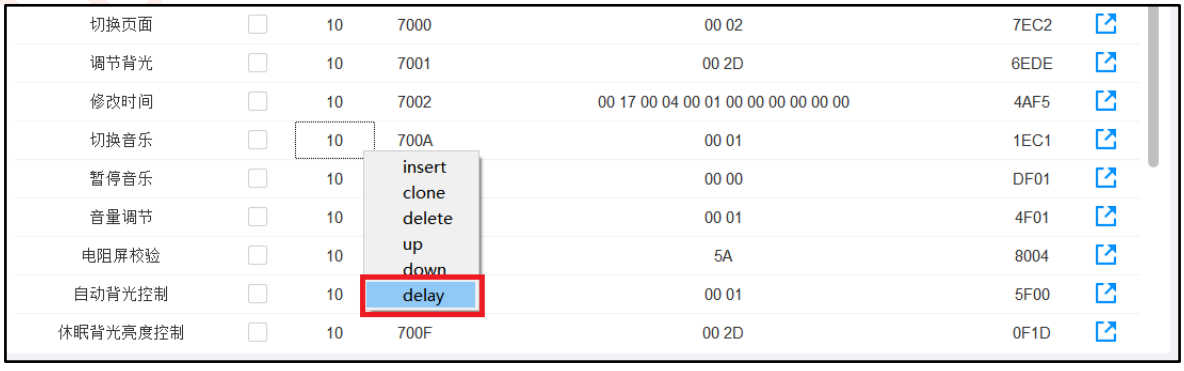


Figure 9-22：Add a delay command

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

Figure 9-23: Setup delay time

The input delay time is in decimal number, unit: ms. No need to input other columns. The total delay time = delay time + Interval Time.

For other Uart commands, refer to *Uart Communication* for more details.

9.2.3. Save Commands

Figure 9-24: Load & Save Commands

Click on  to save the commands listed in the Command Edit Area. See the example shown in Figure 9-25. Developers may also edit the commands in the txt file directly. Note that **no blank line is allowed in between**.

```
Switch Page,select,10 7000 00 00
Send Data to 0901,unselect,10 0901 00 20
Read 0901 & 0902,unselect,10 0901 00 02
Adjust Backlight,unselect,10 7001 00 2D
Send data to Curve 1,unselect,10 0200 31 2E B1 ED B8 F1 B2 E2 CA D4 D6 D0 00 00 00 00 00 00 00 00 30 30 3
Send data to Curve 2,unselect,10 0250 36 2E BA BA D7 D6 D7 D6 B7 FB BC AF 00 00 00 00 00 00 00 00 35 35 3
Daley(ms),unselect,++ 300
Clear Curve 1 & 2,unselect,10 E003
2,unselect,10 032C 00 00
2,unselect,10 034C 01 00
2,unselect,10 036C 01 00
Daley(ms),unselect,++ 300
3,unselect,10 034C 00 00
3,unselect,10 036C 00 00
3,unselect,10 038C 01 00
3,unselect,10 03AC 01 00
Daley(ms),unselect,++ 300
```

Figure 9-25: Example of a Command File

9.2.4. Message Information File

Click on **Save Message** to save the messages listed in the Message area as a txt file. See the example shown in Figure 9-26.

Note this file cannot be loaded to UI_Debugger-III.

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

Figure 9-26: Example of a Message Information File

9.3. Font Tool

BWFont is designed to customize fonts for the use of UI_Editor-III. To activate the tool, simply click on the [Tool] menu and then click on [Font tool], as shown below:

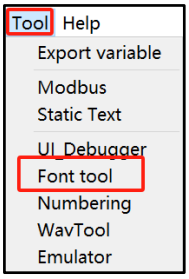


Figure 9-27: Activate BWFont

The main screen of BWFont is shown and explained below:

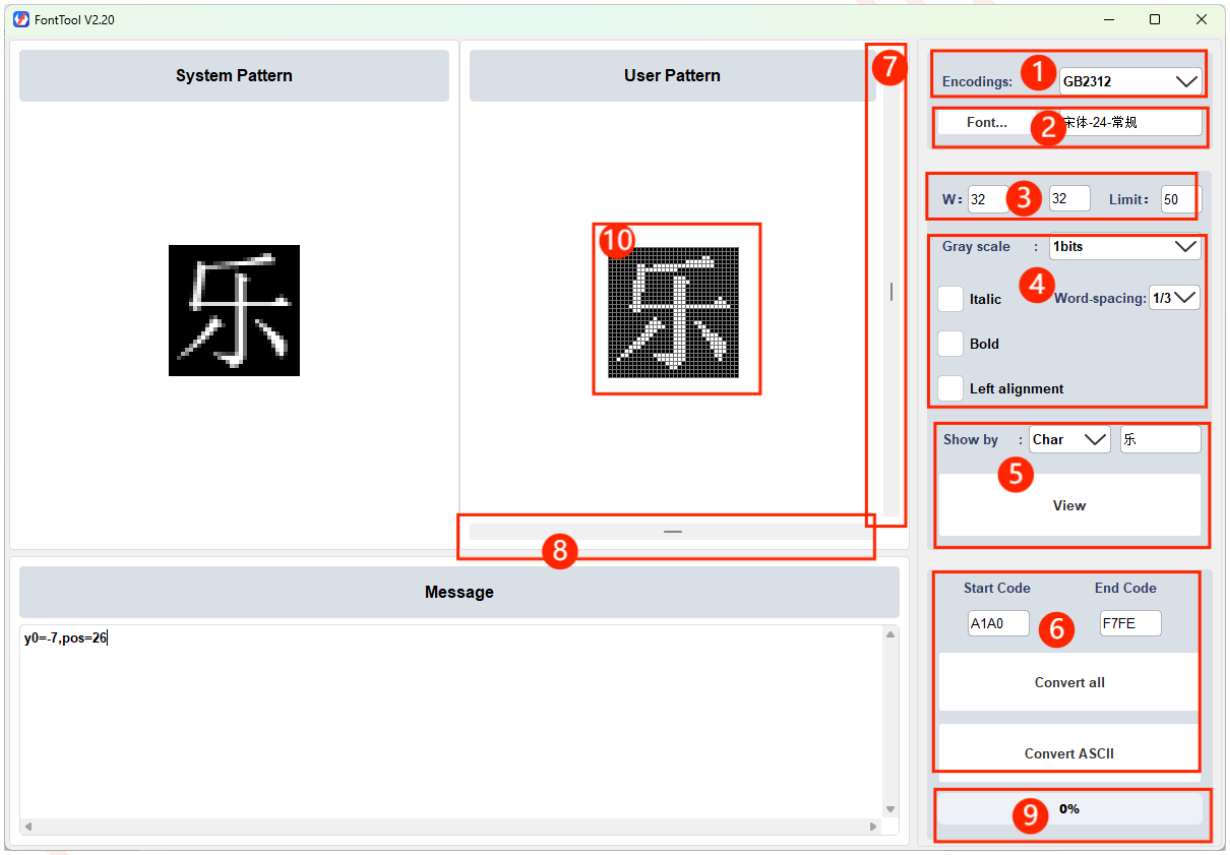


Figure 9-28: Main Screen of BWFont

❶ **Encoding types:**

GB2312: Simplified Chinese

BIG5: Traditional Chinese

GBK: Chinese, including GB2312 and BIG5

Unicode: Encodings for most of the languages in the world. Each character width is defined.

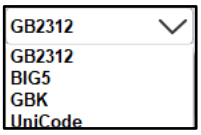


Figure 9-29: Encoding Types

❷ Click on **[Font]** to select Font, Font style, and Size in the popup window.

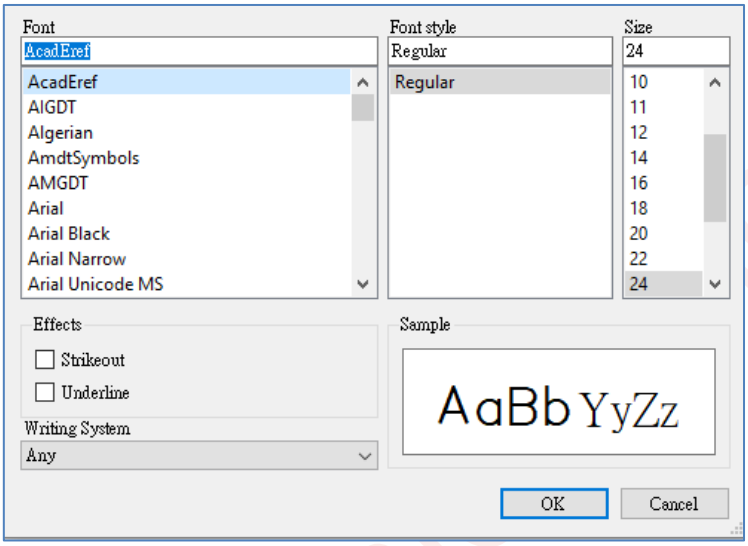


Figure 9-30: Select Font / Font Style / Size

❸ Click on the entry boxes to set the width and height of the font boundary. Width does not have to be the same as height. Unit: pixel. The default **[Limit]** setting is suggested. Usually, with the same **[Limit]** setting, when the width/height is bigger, the font will be plumper.

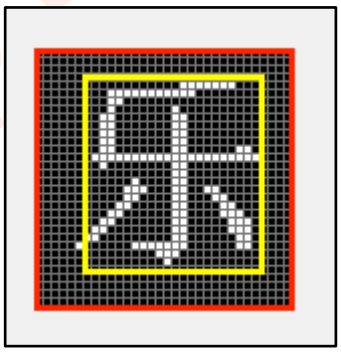


Figure 9-31: Character Example

Note: As shown in Figure 9-31, the red rectangle represents the display boundary for the font. When the font size is modified, it will only change the character size, yet the display boundary will remain the same.

❹ **Gray scale:** Click to select from 1, 2, 4, 8bits and αRGB4444. The higher the grayscale is, the better the display effect will be, however, the bigger the generated file size will be, too. Note that 8bit or αRGB4444 can only be supported by customized IC version. Contact Levetop for more details if needed.

Italic: Italic font

Bold: Bold font

Left alignment: Align the font to the left

Word-spacing: Spacing width between words. For example, if the font width is 32, and the word-spacing is set to 1/2, then the spacing width will be 16pixels.

❺ Click on **[View]** to preview the font. Users may select a character to preview by inputting the character or the code of it. When any of the above item 3, 4, or 5 is changed, the [View] button must be clicked to show the display effect.

❻ **Start code & End code:** For GBK, GB2312, and BIG5, simply apply the default values. When using Unicode, these two values must be set according the selected language coding range.

Covert: Click to generate a file for all the designated font, including ASCII

Covert ASCII: Click to generate a file for ASCII only

❼&❽ **Fine-tune:** Adjust the character position by the slider bars.

❾ **Progress bar:** Display the progress of the font file generation.

Steps of making a font (refer to Figure 9-32 & 9-33):

❶ Select a font encoding and font

❷ Set the width and height of the font boundary.

❸ Set the grayscale

❹ Set the Word-spacing

❺ Click on **[View]** to check the character position. Use the slider bars to adjust the position if needed.

❻ Click on **[Convert]** or **[Covert ASCII]** to generate the file of the font.

❼ & ❽ Assign the file name and path. The file name must not include " * ", refer to *FontBin* for more details.

❾ Click **[Save]** to save the file

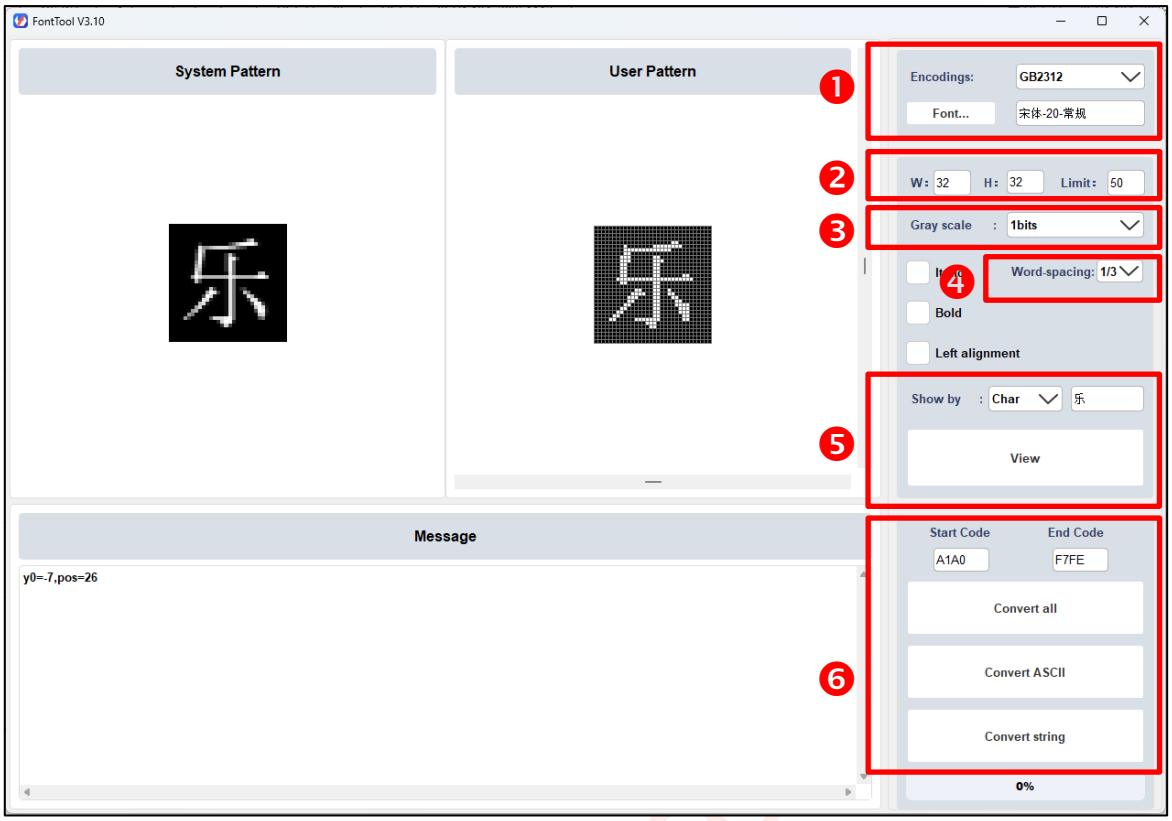


Figure 9-32：Steps of making a font (1)

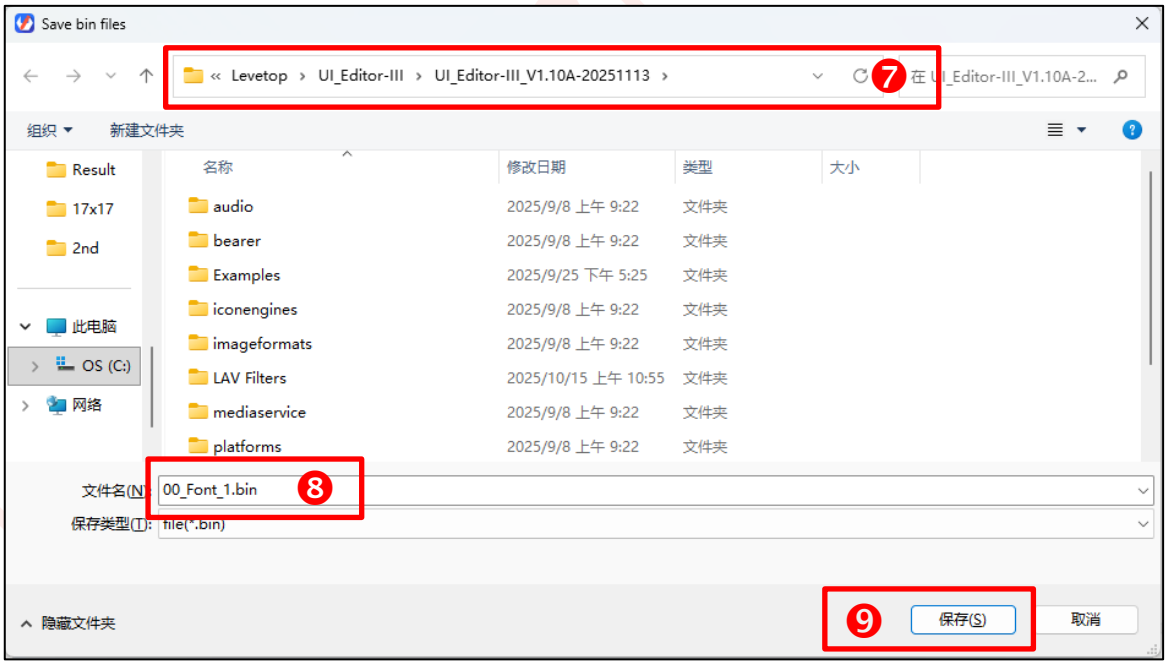


Figure 9-33：Steps of making a font (2)

9.4. Numbering Tool

Numbering_tool is designed to number the pictures and icons that will be used in UI_Editor-III. To activate the tool, simply click on the [Tool] menu and then click on [Numbering], as shown below:

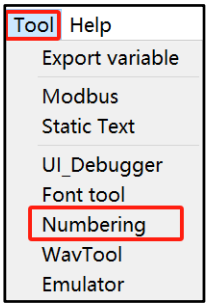


Figure 9-34: Activate Numbering_tool

The main screen of Numbering_tool is shown and explained below:

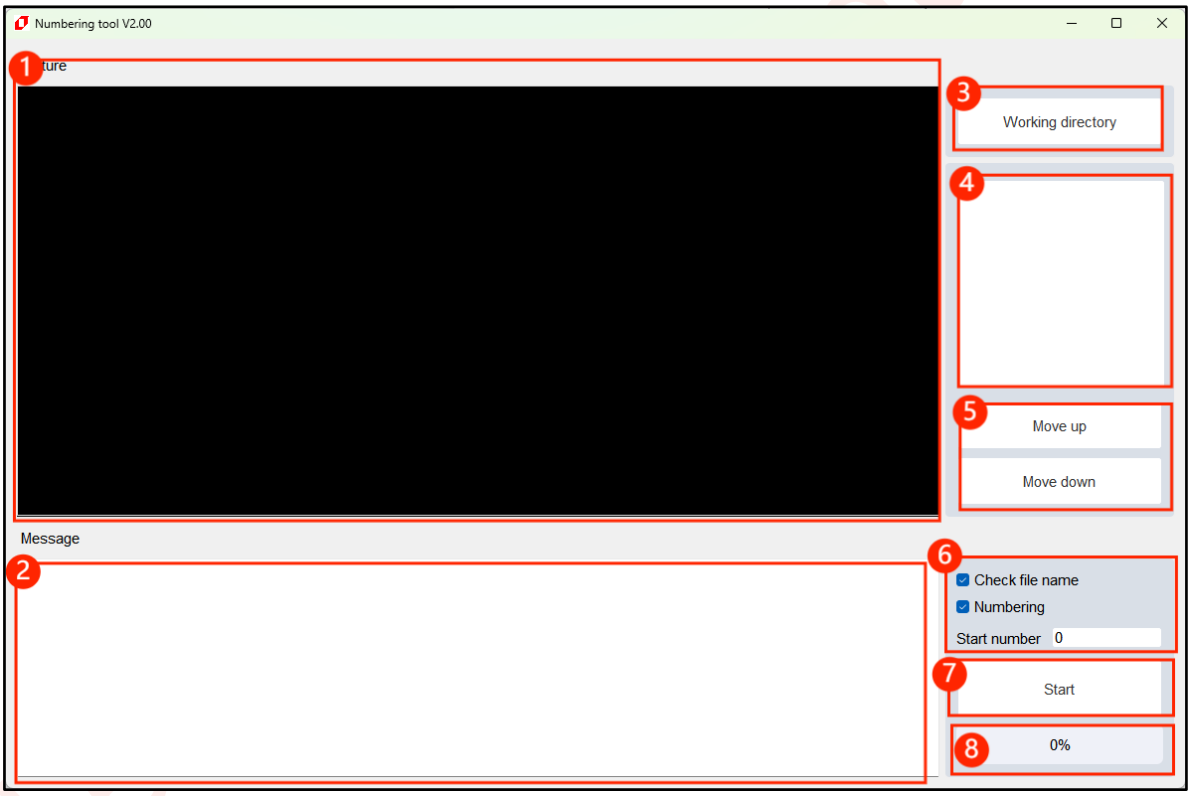


Figure 9-35: Main Screen of Numbering_tool

❶ Picture preview area

❷ Message area: Operation messages will be prompted here.

❸ Working directory: Click to add pictures/icons. Note that all pictures in the designated folder will be loaded.

❹ Picture list: The loaded picture names will be listed here

❺ Move up: Move up the selected picture

Move down: Move down the selected picture

❻ Check file name：If checked, the illegal file names will be corrected automatically.

Numbering：If checked, the pictures in the designated directory will be numbered.

Start number：Set the start number of the numbering operation.

❼ Start：Click on **[Start]** to start executing the settings of ❻ above.

❽ Display the processing progress.

Steps of numbering pictures:

1、Click on **[Working directory]** to open the target picture directory. Select a picture in the popup window, and then click **[Open].** As shown below:

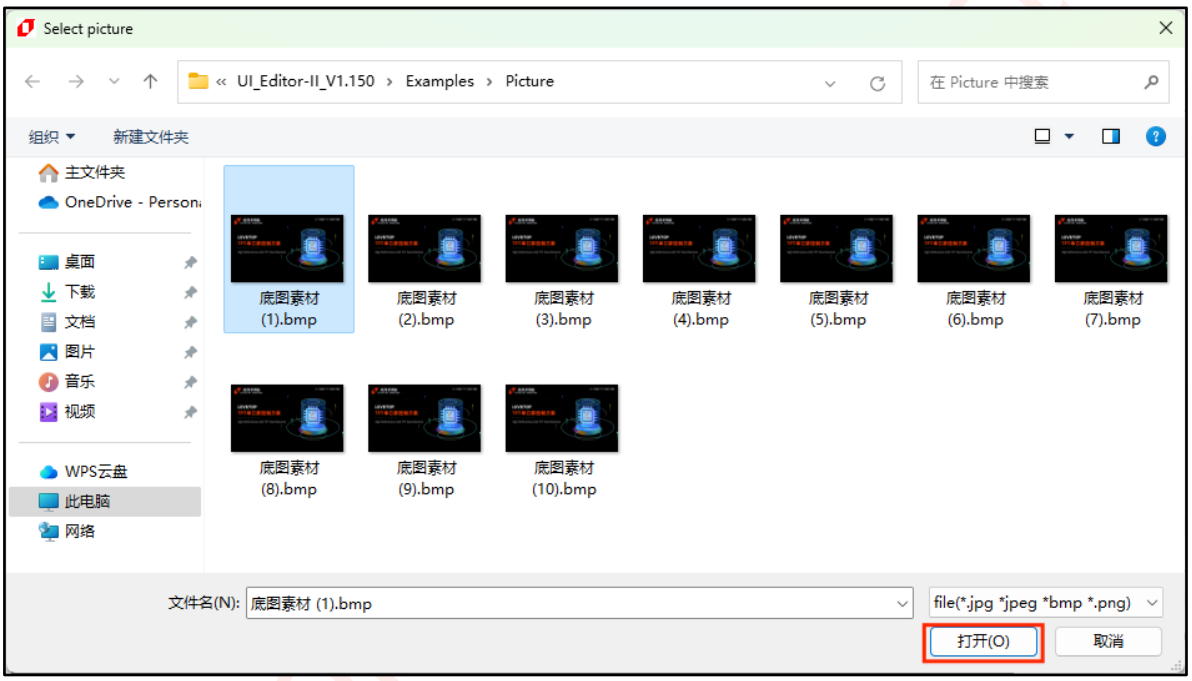


Figure 9-36：Select a Picture

2、As shown in Figure 9-37, the selected picture is displayed in the Picture preview area. All pictures of the same directory are listed on the right. Adjust the picture order by **[Move up]** and **[Move down]** buttons so that the picture order meet the design requirement.

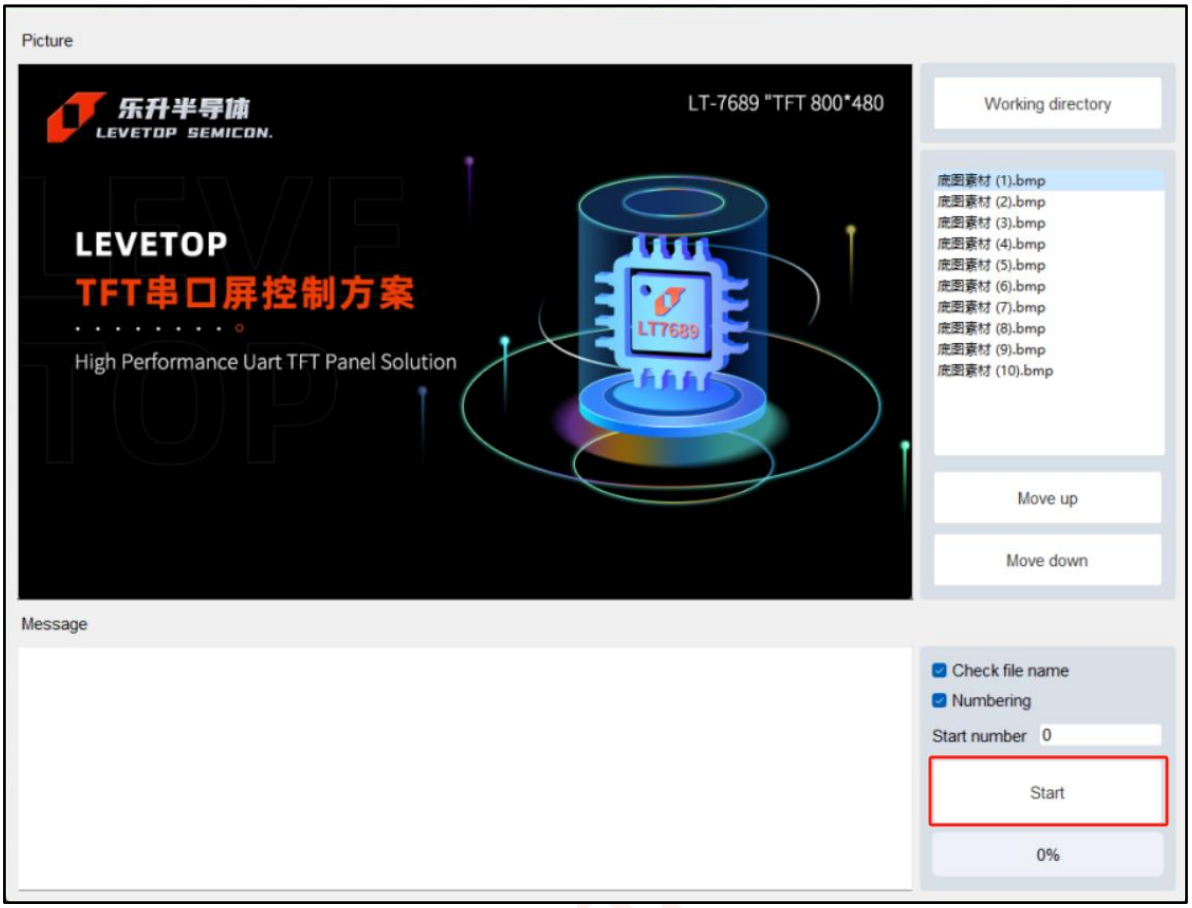


Figure 9-37：Adjust the picture order

3、Click on [Start] to number the pictures.

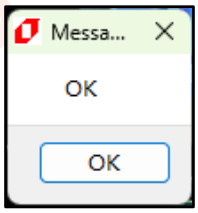


Figure 9-38：Process done

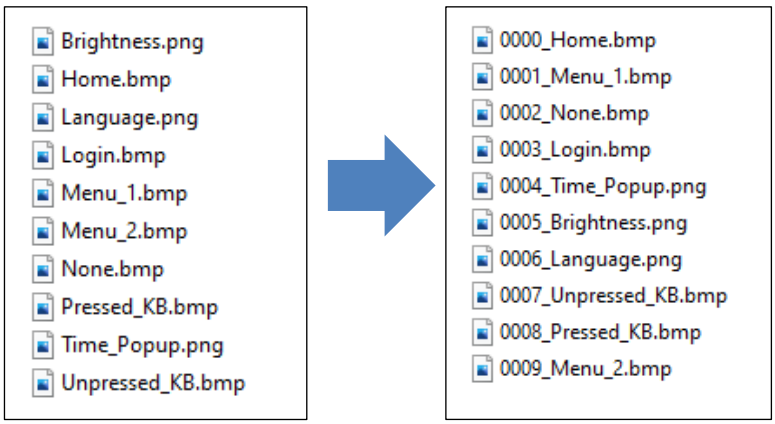


Figure 9-39：Final Result

9.5. WavTool

9.5.1. Make a Wav file

If an audio file is not in the Wav format, developers will need to convert it into Wav format in order to use the related functions in UI_Editor-III.

9.5.2. Convert Wav to Bin

WavTool is designed to convert wav files into bin files. To activate the tool, simply click on the [Tool] menu and then click on [WavTool], as shown below:

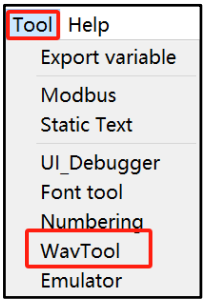


Figure 9-40: Activate WavTool

The main screen of WavTool is shown and explained below:

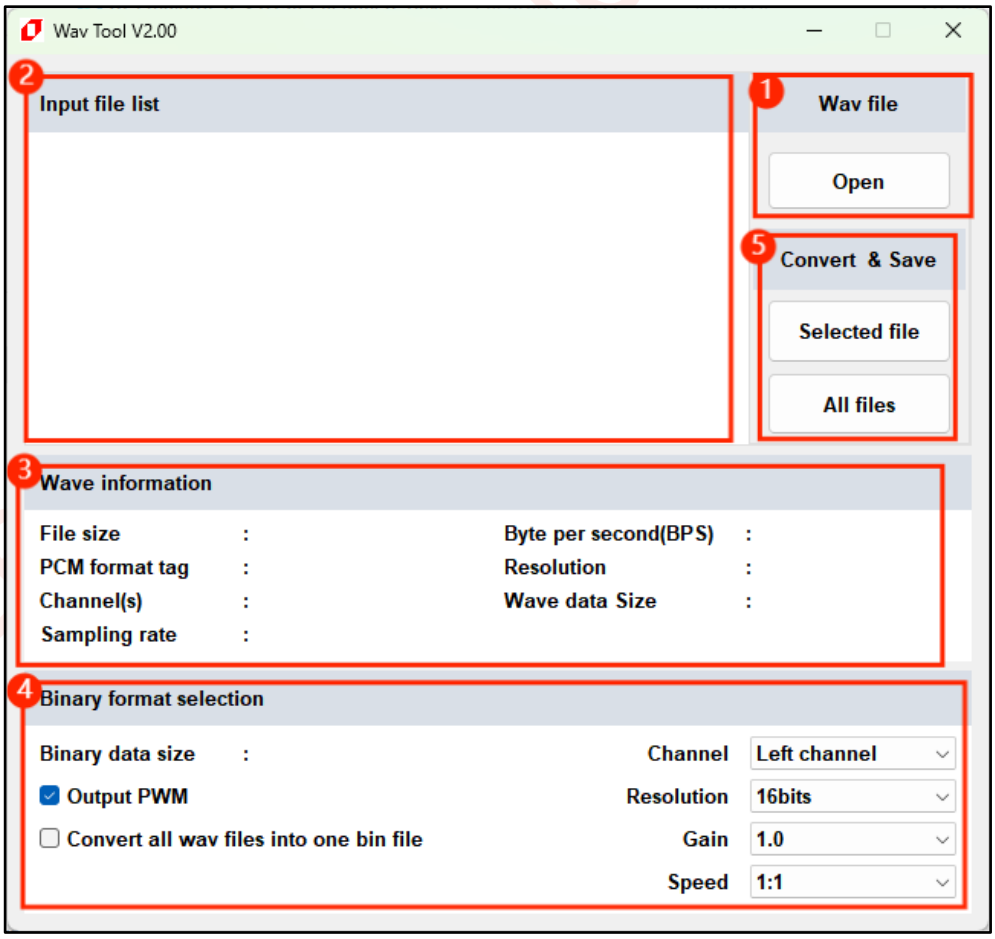


Figure 9-41: Main Screen of WavTool

❶ **Wav file:** Click on **[Open]** to load wav files

❷ **Input file list:** This area will list the names of all loaded wav files

❸ **Wav information:** The parameters of the loaded wave file will be shown here. No modification allowed.

Sampling rate: The sampling rate of the wave file must be 22050

Other parameters: No specific requirements.

❹ **Binary format selection: The parameter settings of the bin file**

Binary data size: bin file size, no modification required.

Output PWM: PWM output value, no modification required.

Convert all wav files into one bin file: If checked, all wave files will be packaged into one bin file.

Channels：Sound channel options. Default: Left channel.

Resolution：Sampling bits. Must be set to 16bits

Gain：Default: 1.0

Speed：Default: 1:1.

Note: WavTool is not professional audio converting software, adjusting above parameters may distort the audio.

❺ **Convert & Save**：

Selected file：Click to convert the selected wav file into a bin file.

All files：Click to convert all wav files into bin files.

Steps of converting wav files to bin files:

1、Activate WavTool and click on **[Open]** to add wav files. Select a wav file in the popup window, and then click **[Open].** As shown below:

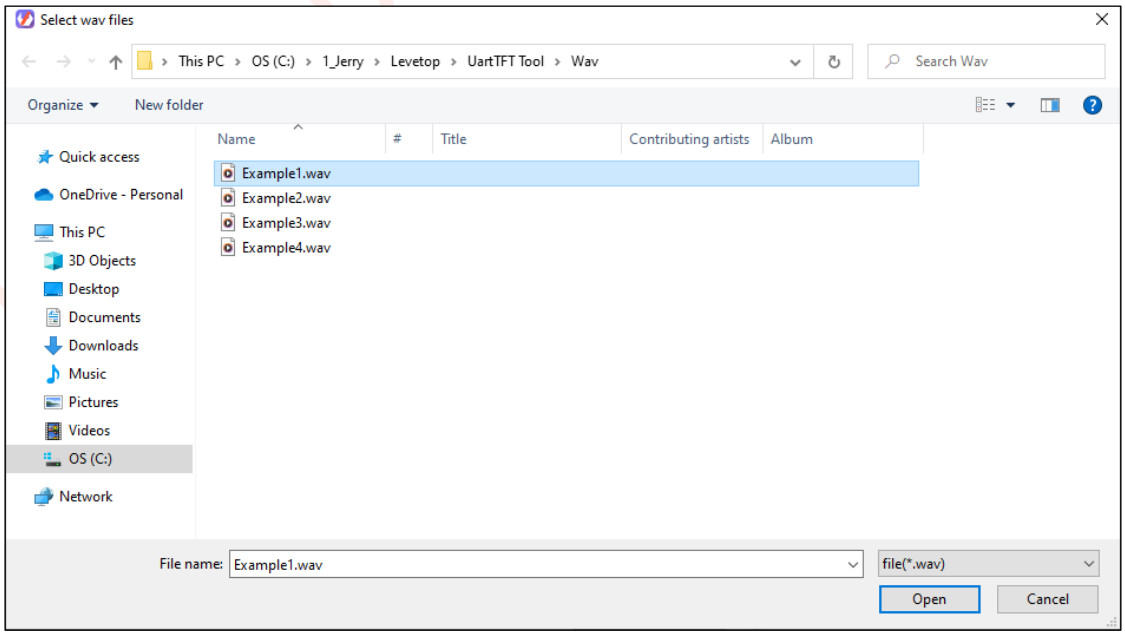


Figure 9-42：Select a Wav File

2、As shown in Figure 9-43, all the wav files are listed in the **Input file list** area. If the sample rate (as the blue rectangle indicated) is not 22050, developers must remake a new wav file by 22050 sampling rate. Click on **[Selected file]** or **[All files]** to generate the bin file(s).

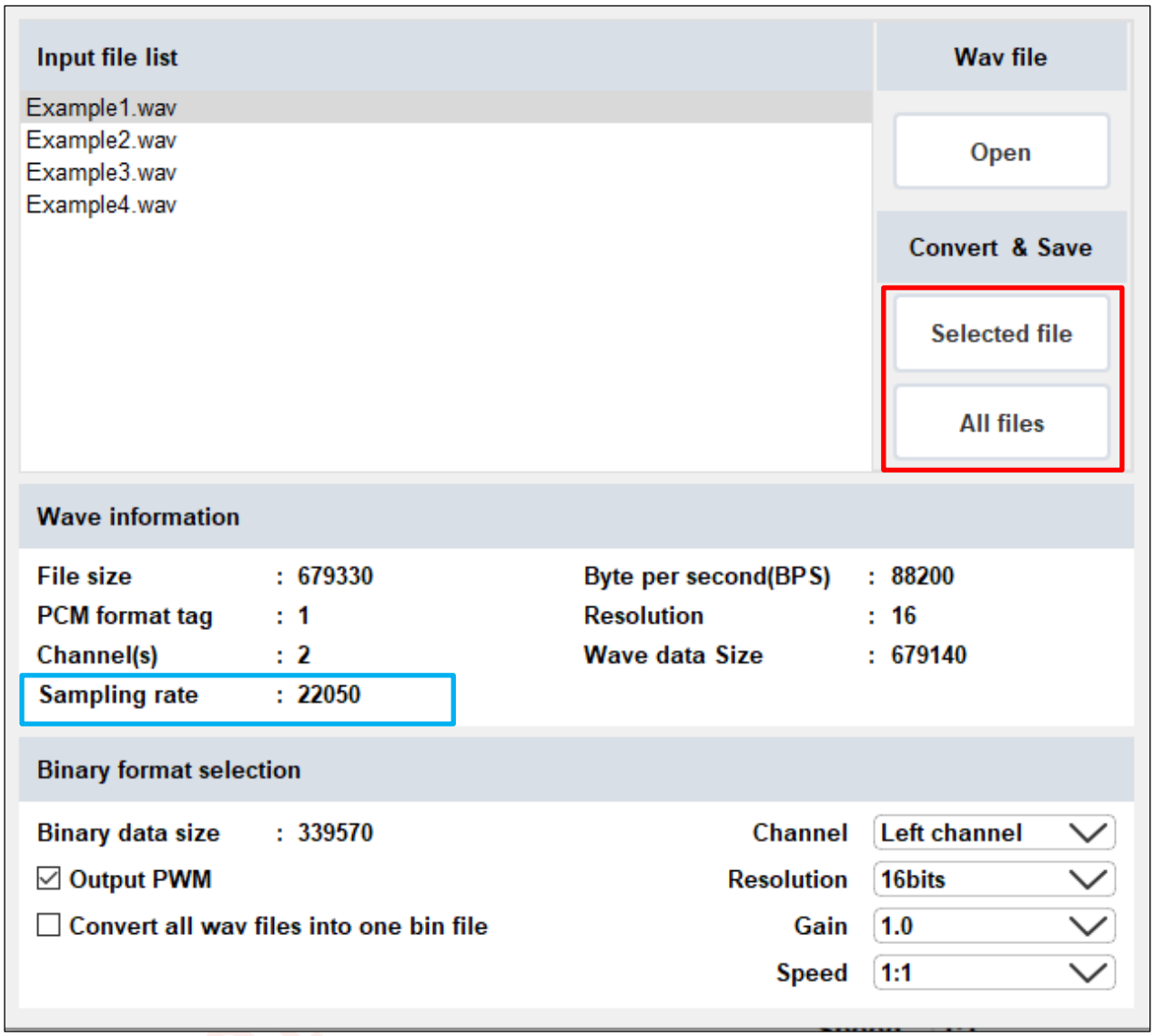


Figure 9-43：Convert Wav to Bin

3、Save the generated bin file to designated path. Note the bin file must be assigned a new name, and should not be named the same as an existed bin file.

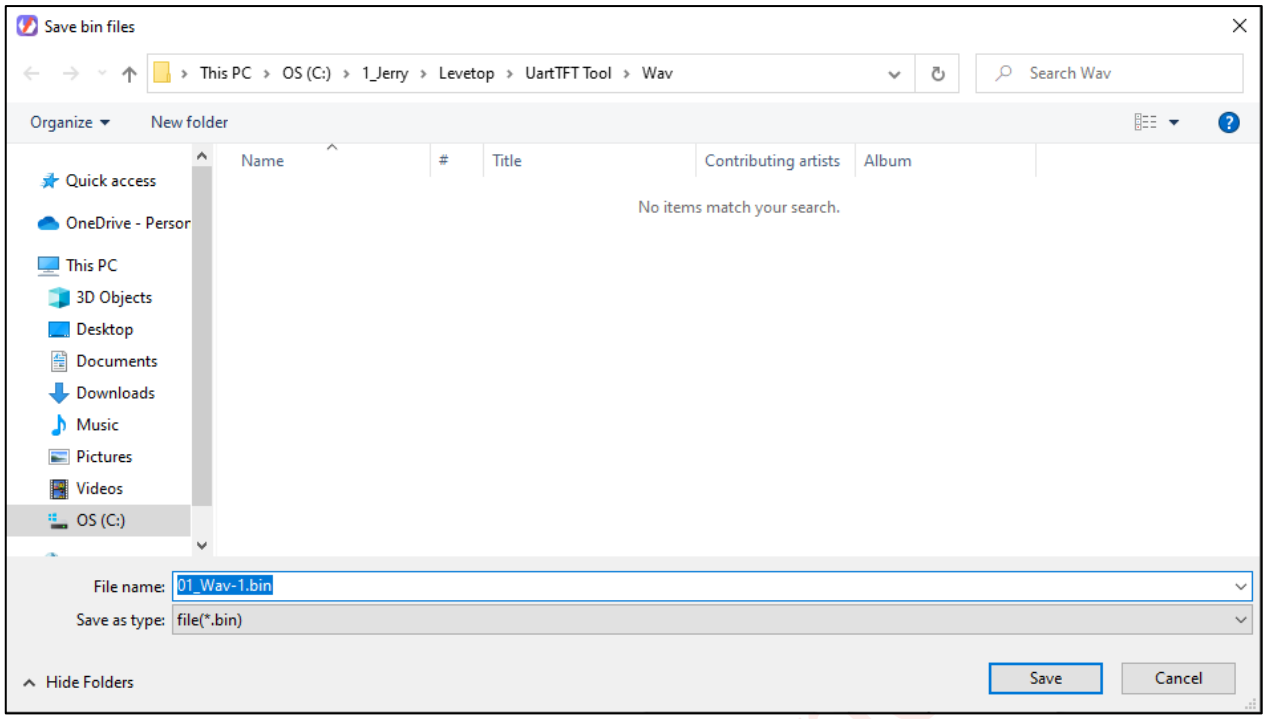


Figure 9-44: Save the bin file

Note: The file name should not include special characters such as \ / : * ? “ < > |

9.6. Convert a UI_Editor-II Project to a UI_Editor-III Project

LT_UI_Convertor_II_to_III.exe is a tool designed to convert a UI_Editor-II project to a UI_Editor-III project.

Figure 9-45: LT_UI_Convertor_II_to_III

Double click on [LT_UI_Convertor_II_to_III.exe] to activate the tool:

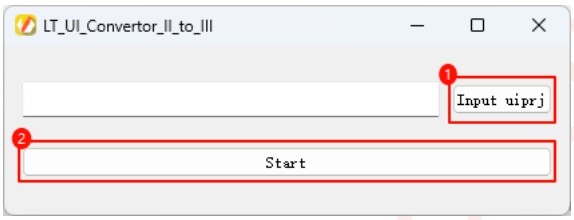


Figure 9-46: Main Screen of LT_UI_Convertor_II_to_III

1、Import the uiprj file：Select the UI_Editor-II project to be converted.

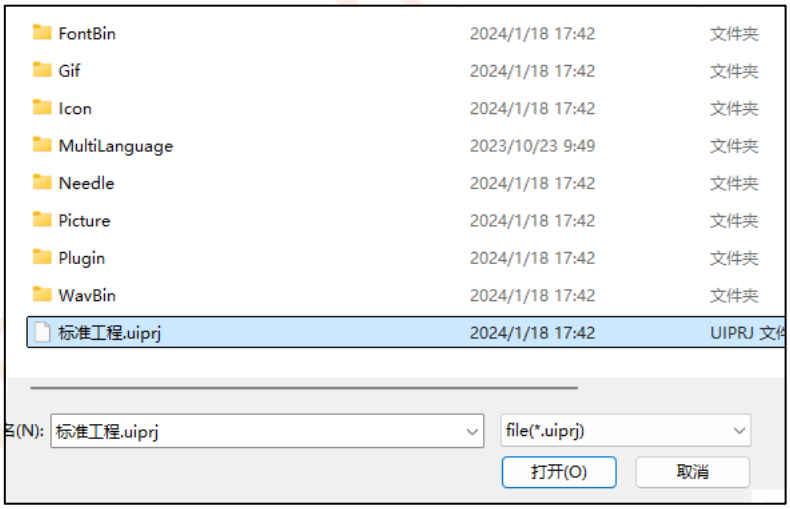


Figure 9-47: Select the UI_Editor-II Project

2、 Click [Start] to start converting

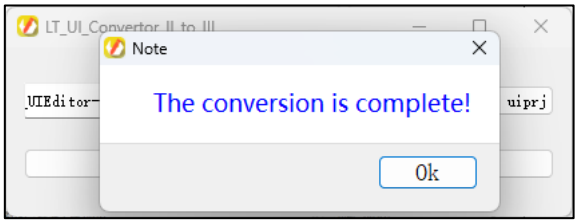


Figure 9-48: Converting Done

Note:

1. Before proceeding with the conversion, back up the project, as the conversion process is irreversible.
2. The same project cannot be converted multiple times. If issues arise after conversion, please use the original UI_Editor-II project to redo the conversion.
3. The encoder widget structure of UI_Editor-II is different from that of UI_Editor-III. Therefore, after the conversion, developers must delete the original encoder widget, add a new encoder widget, and setting it based on the UI_Editor-III design.

10. Uart Communication

There are three command types: (1) Write command, which is to write data to a designated address (register); (2) Read Command, which is to read data from a designated address (register); (3) Touch returned message, which is a returned message from the UartTFT controller when the touch panel is operated. In addition, after a write/read command is sent, a returned message will be sent by the UartTFT controller. For command formats, refer to Table 10-1. (**"0x" represents hexadecimal number, no need to include it in actual implementation**.)

A Uart tool for debugging purpose is available, refer to *UI_Debugger-III* for more details.

Table 10-1: Command Formats

Host write data	Header 0xXXXX	\	Length 0xXX	Write Command 0x10	Address 0x0000 ~ 0x7FFF	Write Data 0xXXXX……0xXXXX (2*n Bytes < 250Bytes)		CRC 0xXXXX
Host write data (>250bytes)	Header 0xXXXX	Fixed 0x00 (1byte)	Length 0xXXXX	Write Command 0x10	Address 0x0000 ~0x7FFF	Write Data 0xXXXX……0xXXXX (2*n Bytes > 250Bytes)		CRC 0xXXXX
Host read data	Header 0xXXXX	\	Length 0xXX	Read Command 0x03	Address 0x0000 ~ 0x7FFF	Read Word amount 0xXXXX		CRC 0xXXXX
Returned message	Header 0xXXXX	\	Length 0xXX	Read Command 0x03	Address 0x0000 ~ 0x7FFF	Read Word amount 0xXXXX	Data (2*n Bytes)	CRC 0xXXXX
Touch Returned message	Header 0xXXXX	\	Length 0xXX	Command 0x41	Address 0xXXXX	returnValue 0xXXXX		CRC 0xXXXX

The format of a command/returned message is described as below:

1、**Header:** Used to recognize a start of a new command or returned message. The default value is 0x5A A5. This value can be customized in the project setting page of UI_Editor-III.

2、**0x00 (Fixed):** This byte is fixed for the instructions that write more than 255 bytes data. "\" indicates that there is no data for this byte in the instruction.

3、**Length:** Command length. **Length = Command (1) + Address(2) + Write Data(2*N) + CRC(2)**

4、**Write/Read Command:** Command type. 0x10: Write; 0x03: Read; 0x41: Touch returned message

5、**Address:** Variable/Register address. Data length: 2Bytes

6、**Data:** Data to be written / Data amount to be read.

7、**CRC:** Cyclic Redundancy Check. Data length: 2Bytes

10.1. Write Command

Host may send a “Write command” to designated address of UartTFT controller to implement operations such as switching display page, adjusting backlight brightness etc. There are two kinds of address, one is widget related address such as writeAddr and parameterAddr. This kind of address must be set by developers in advance. Host may control the widgets by writing commands to the related addresses. Another kind of address is the register address. Each register has its own purpose. Refer to *Variable Address* for more detail.

Write command protocol:

Host writes a command to UartTFT controller → UartTFT controller verifies the received command → UartTFT controller returns passed message to the Host if CRC is passed, otherwise returns failed message to the Host.

Write Command Code: 0x10, refer to the command format below:

Table 10-2: Format of Write Command (<=250Bytes) and Returned Message

Host write data	Header 0xXXXX	Length 0xXX	Write Command 0x10	Address 0x0000 ~ 0x7FFF	Write Data 0xXXXX..... 0xXXXX (2*n Bytes)	CRC 0xXXXX
Write 0x5152 and 0x5354 to the address of 0x2001	0x5AA5	0x09	0x10	0x2001	0x5152 0x5354	0xBC43
CRC Pass	0x5AA5	0x04	0x10	0xFF		0x4C30
CRC Fail	0x5AA5	0x04	0x10	0x00		0x0C70

Table 10-3: Format of Write Command (>250Bytes) and Returned Message

Host write data	Header 0xXXXX	Fixed 0x00 (1byte)	Length 0xXX	Write Command 0x10	Address 0x0000 ~ 0x7FFF	Write Data 0xXXXX.....0xXXXX (2*n Bytes)	CRC 0xXXXX
Write 0x0001 ~ 0x0E0F to the address of 0x0001	0x5AA5	0x00	0x0105	0x10	0x0001	0x0001 0x0203.....0x0C0D 0x0E0F	0xE1CF
CRC Pass	0x5AA5	\	0x04	0x10	0xFF		0x4C30
CRC Fail	0x5AA5	\	0x04	0x10	0x00		0x0C70

Example (by UI_Debugger-III):

1、Write 0x1020 to the address of 0x2001： **0x10 0x2001 0x1020**

CMD	Addr	Data	CRC	Send
10	2001	1020	B3 DB	

Figure 10-1：Example of Write Command（1）

2、Write 0x1020 and 0x2022 to the address of 0x2001： **0x10 0x2001 0x1020 0x2022**

CMD	Addr	Data	CRC	Send
10	2001	1020 2022	AC B2	

Figure 10-2：Example of Write Command（2）

Note:

1、The amount of data (Data column) must be **2*n Bytes**. (The amount of data cannot be odd.)

Incorrect：0x10 0x2000 **0x31 0x32 0x33 → Data amount = 3 Bytes**

Correct：0x10 0x2000 **0x31 0x32 0x33 0x34 → Data amount = 4 Bytes**

2、If a Uart debugging tool other than UI_Debugger-III is used, the write commands must be complete, including **Header**, **Length**, and **CRC** data, as described in Table 10-2

3、The returned messages listed in Table 10-2 are fixed formats. If CRC is passed, the returned data will be 0xFF, otherwise, the returned data will be 0x00.

10.1.1. Write Commands to Control Widgets

10.1.1.1. Example: String_Label & Text Scroll widget

Parameter	Data
name	label_0
parameterAddr	0xFFFF
writeAddr	0x0720
wordLength	20
X	101
Y	167
W	189
H	188
fontWidth	32
fontHeight	32
fontID	05_Font-GBK_微...
encoding	GBK
alignment	Left
backgroundColor	Disable
_color	0xD3D3D3
fontColor	0x000000
defaultText	文字测试
passwordMode	Disable

Parameter	Data
name	textroll_0
parameterAddr	0xFFFF
writeAddr	0x0720
X	356
Y	179
W	220
H	103
wordLength	32
fontWidth	32
fontHeight	32
fontID	05_Font-GBK_微...
encoding	GBK
fontColor	0x000000
backgroundColor	0x0000FF
trailingSpace	64
interval(10ms)	50
alignment	Left
scrollMode	Enable
defaultText	文字测试
transparency	Enable

Figure 10-3: Parameters of String_Label & Text Scroll

As shown in Figure 10-3, the initial Chinese string is “文字測試”, and the address of both widgets are the same as 0x0720. Figure 10-4 shows an example of updating the text to “乐升”. The code for “乐” is C0D6, and the code for “升” is C9FD. Therefore, the command can be formed as:

Header + 0B 10 07 20 C0 D6 C9 FD 00 00 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	0720	C0 D6 C9 FD 00 00	51 AA	

Figure 10-4: Write Text Data

Note that a 2 bytes data, 00 00, must be added to the end of the text data as an ending sign

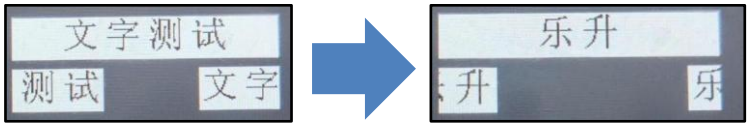


Figure 10-5: Write Command to Change Texts

Note: String_Label supports linefeed function, simply insert 0x0A to start a new line (the widget height must be set tall enough for displaying the new line). Text Scroll can only display one line, and does not support linefeed function.

10.1.1.2. Example: Text Number & Graphics Number Widgets

Parameter	Data
name	number_0
parameterAddr	0xFFFF
writeAddr	0x0280
byteLength	4
X	99
Y	131
W	161
H	163
fontWidth	32
fontID	02_Font-微软雅...
encoding	GB2312
alignment	Left
integerDigit	6
decimalDigit	3
dataType	int
unitSymbol	
_length	0
fontColor	0x000000
defaultNumber	0
leadingZero	Disable

Parameter	Data
name	pngNumber_0
parameterAddr	0xFFFF
writeAddr	0x0280
byteLength	4
X	360
Y	189
W	218
H	36
integerDigit	6
decimalDigit	3
dataType	int
alignment	Left
firstIcon	0116.png
lastIcon	0128.png
defaultNumber	0
leadingZero	Disable

Figure 10-6: Text Number & Graphics Number Widgets

As shown in Figure 10-6, the writeAddr of both widgets is the same as 0x0280.

The related parameters include,

dataType: int

integerDigit (the digit number of the integer)**:** 6

decimalDigit (the digit number of the decimal)**:** 3

defaultNumber: 0

To make the two widgets show a different value, say **6.000**, the command should look like as below:

Header + 09 10 02 80 00 00 17 70 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	0280	0000 1770	04 E9	↗

Figure 10-7: Write Number Data

As shown in the above command, the value (6.000) is transformed to **00 00 17 70**. Since the data type is int, each number will be taken as 4 bytes. The higher bytes must be filled with 0 if the input number is less than 4 bytes after transformed to its hex value. Also, decimal digits will be taken as same as the integer digits. That is,

the value 6.000 will be taken as 6000 whose hex value is 1770. Therefore, the final data is formed as 00 00 17 70. Figure 10-8 shows the display result. The upper part shows the result of a Text Number Display widget, and the lower part shows that of a Graphics Number Display widget. (For Text Number Display widget, the redundant “0” after the decimal point will be truncated.)

Here is another example. To change the number to **6.00**, then the command will be as following:

Header + 09 10 02 80 00 00 02 58 + CRC

As shown in the above command, the value (6.00) is transformed to **00 00 02 58**

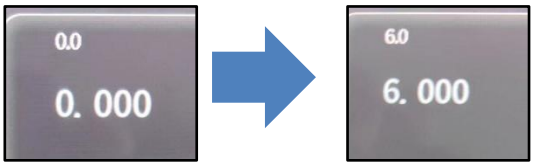


Figure 10-8: Write Commands to Change Numbers

10.1.1.3. Example: QRCode

Parameter	Data
name	qrcode_0
parameterAddr	0xFFFF
writeAddr	0x0623
byteLength	200
X	410
Y	155
W	100
H	100
size(50pixels)	2
content	ABC123

Figure 10-9: QRCode Parameters

As shown in Figure 10-9, the widget address is 0x0623, and the initial string is “ABC123”. To change the string to “abc456”, the command will be as following: (00 00 is the ending code for text input)

Header + 0D 10 06 23 61 62 63 34 35 36 00 00 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	0623	61 62 63 34 35 36 0000	EC BD	

Figure 10-10: Write QR Code Data

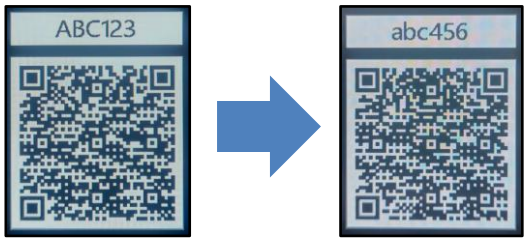


Figure 10-11：Write Texts to QRCode Widget

10.1.1.4. Example: Bit Status

Parameter	Data
name	bitIcon_0
parameterAddr	0xFFFF
writeAddr	0x0520
bitIndex	bit0
X	456
Y	232
W	78
H	78
offStateIcon	0043.png
onStateIcon	0044.png
overlap	Disable

Figure 10-12: Bit Status Parameter

As shown in Figure 10-12, the widget address is 0x0520, and the trigger bit is bit0. To trigger this widget, simply send a data to set bit0 to 1. Please refer to the below command:

Header + 07 10 05 20 00 01 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	0520	0001	25 05	↗

Figure 10-13: Write Bit Status Data

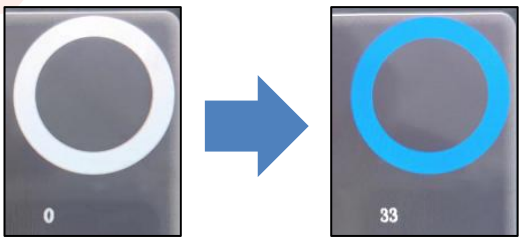


Figure 10-14: Write Command to change Bit Status

10.1.1.5. Example: Icon Widget

Parameter	Data
name	icon_0
parameterAddr	0xFFFF
writeAddr	0x0500
byteLength	2
X	483
Y	194
W	24
H	36
firstIcon	0060.png
lastIcon	0071.png
dataFormat	
defaultDisplayID	
minDisplayID	0
maxDisplayID	11
overlap	Disable

Figure 10-15: Icon Parameters

As shown in Figure 10-15, the widget address is 0x0500, and the ID range is 0 ~ 11, including pictures of 0 ~ 9, a decimal point, and a comma. To display the number 8, the command will be as following:

Header + 07 10 05 00 00 08 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	0500	0008	E4 C9	

Figure 10-16: Write Icon Data

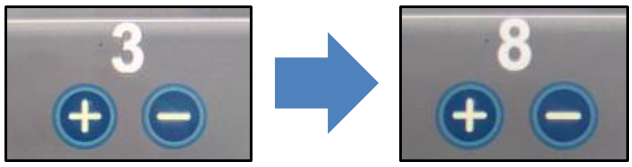


Figure 10-17: Write Command to switch icons

10.1.1.6. Example: Trend Graph

There are two kinds of command format for Trend Graph widget. One is for updating the trend graph, another is for clearing the trend graph.

Address: This parameter is used to designate the channel of the Trend Graph widget for receiving the data. There are two modes:

Single Channel: Select one channel to update/clear the graph data.

Multiple Channels: Select multiple channels to update/clear the graph data at the same time. For example, Host may send 10 sets of data (0 ~ 9) to channel 0 and channel 1 at the same time, where the 0, 2nd, 4th, 6th, and 8th sets of data will be sent to channel 0, and the 1st, 3rd, 5th, 7th, 9th sets of data will be sent to channel 1. Table 10-4 & Table 10-5 show the address definition.

Table 10-4: Address Definition – for Updating Trend Graph

	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
Single Channel	0xC001	0xC002	0xC004	0xC008	0xC010	0xC020	0xC040	0xC080
Multiple Channel (Example)	0xC003		0xC00C		0xC0F0			

Table 10-5: Address Definition – for Clearing Trend Graph

	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7
Single Channel	0xE001	0xE002	0xE004	0xE008	0xE010	0xE020	0xE040	0xE080
Multiple Channel (Example)	0xE003		0xE00C		0xE0F0			

Note: The 8 channels (0 ~ 7) are represented by the lower byte of 0xC0XX. When the bit0 of the lower byte is 1, it means Channel 0 is selected; if both bit0 and bit1 is 1, it means both Channel 0 and Channel 1 are selected. To select all the 8 channels, bit0 to bit7 should all be set to 1, which means the hex value is FF, that is, the address should be set to 0xC0FF. To clear graph data, simply send a command with the address starting with 0xE0 instead of 0xC0. For example, set the address to 0xE0FF to clear the data of all the channels.

An example is listed below:

Parameter	Data
name	curve_0
parameterAddr	0xFFFF
X	100
Y	0
W	600
H	480
y_ReferenceLine	180
_referenceValue	300
lineColor	0xFF0000
channel	0
x_Spacing(Pixels)	100
lineWidth	3
direction	R-L
maxData	480
minData	0

Parameter	Data
name	curve_1
parameterAddr	0xFFFF
X	100
Y	0
W	600
H	480
y_ReferenceLine	180
_referenceValue	300
lineColor	0x00FF00
channel	1
x_Spacing(Pixels)	100
lineWidth	3
direction	R-L
maxData	480
minData	0

Parameter	Data
name	curve_2
parameterAddr	0xFFFF
X	100
Y	0
W	600
H	480
y_ReferenceLine	180
_referenceValue	300
lineColor	0x0000FF
channel	4
x_Spacing(Pixels)	100
lineWidth	3
direction	R-L
maxData	480
minData	0

Figure 10-18: Trend Graph Parameters

Figure 10-18 shows the parameters of three Trend Graph widgets. These widgets are set to channel 0 (**Red**), 1 (**Green**), and 4 (**Blue**) respectively. No zooming used, and the base-line is at (X, 400). The command format for trend graph widget is as the table shown below:

Table 10-6: Command Format for Trend Graph

Update Trend Graph	Header 0xXXXX	Length 0xXX	Write Cmd 0x10	Address 0xC000~0xCFFF	Write Data 0xXXXX….0xXXXX (2*n Bytes)	CRC 0xXX 0xXX (2 Bytes)
Clear Trend Graph	Start Code 0xXXXX	Length 0x05	Write Cmd 0x10	Address 0xE000~0xEFFF	NULL	CRC 0xXX 0xXX (2 Bytes)

Example: Send 200, 100, 200, 100, 200, and 100 to channel 0, the command is as below

Header + 11 10 C0 01 00C8 0064 00C8 0064 00C8 0064 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	C001	00C8 0064 00C8 0064 00C8 0064	66 42	

Figure 10-19: Write Trend Graph Data (1)

The display result is as shown in Figure 10-20.

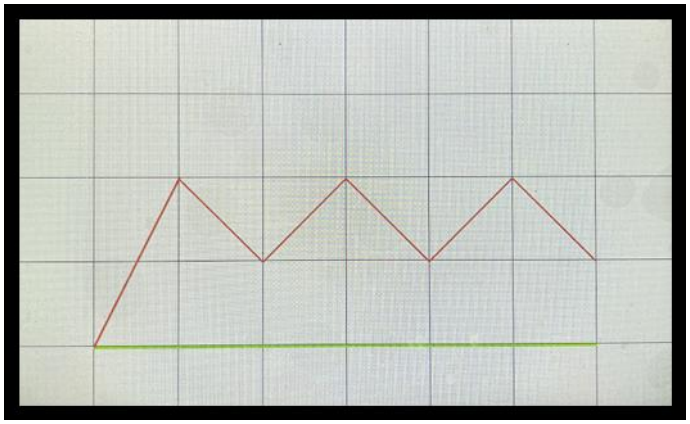


Figure 10-20：Display Result of Channel 0

Example: Send 200, 100, 200, 100, 200, and 100 to channel 1, the command is as below

Header + 11 10 C0 02 00C8 0064 00C8 0064 00C8 0064 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	C002	00C8 0064 00C8 0064 00C8 0064	63 81	↗

Figure 10-21：Write Trend Graph Data（2）

The display result is as shown in Figure 10-22.

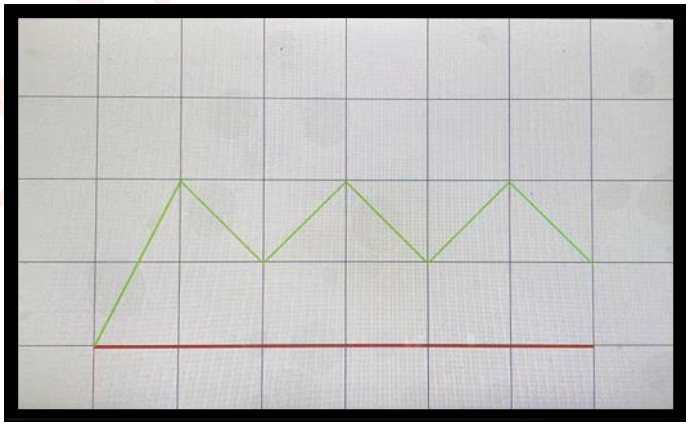


Figure 10-22：Display Result of Channel 1

Example: Send 200, 100, 200, 100, 200, and 100 to channel 0 and channel 1, the command is as below,

Header + 11 10 C0 03 00C8 0064 00C8 0064 00C8 0064 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	C003	00C8 0064 00C8 0064 00C8 0064	61 00	

Figure 10-23: Write Trend Graph Data (3)

Channel 0 will receive 3 sets of 200, and channel 1 will receive 3 sets of 100. The display result is shown in Figure 10-24.

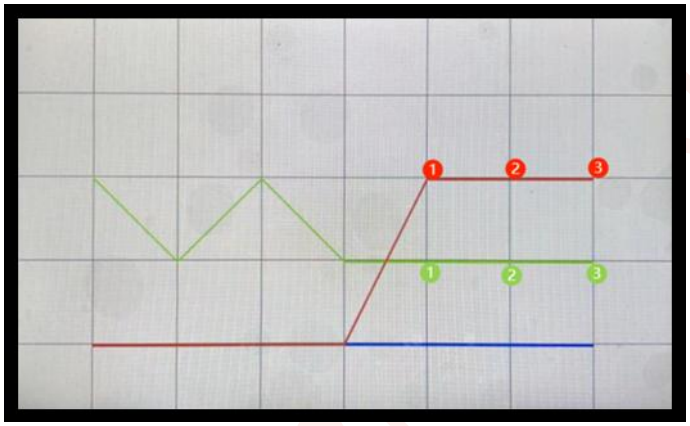


Figure 10-24: Display Result of Multi-Channels (Channel 0 & Channel 1)

Example: Clear the data in channel 0, the command is as below,

Header + 07 10 E0 01 00 00 + CRC

Select	CMD	Addr	Data	CRC	Send
☐	10	E001	0x0000	82 A1	

Figure 10-25: Clear Trend Graph Data

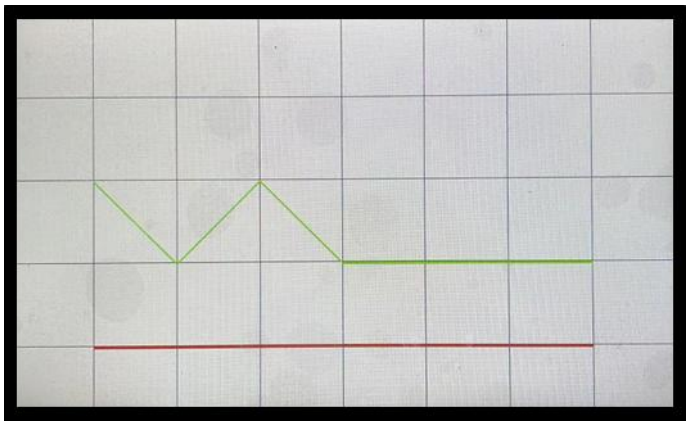


Figure 10-26: Clear the Data of Channel 0

10.1.2. Write Data to Control Registers

Similar to the general command formats, host may simply write data to the address of control registers to execute specific functions. The register addresses range from 0x7000 ~ 0x71FF. User definable addresses range from 0x0000 ~ 0x7FFFF (the control register addresses listed above are not included). Refer to *The address list of variables and special registers* for more details.

10.1.2.1. Page Register – 0x7000

Example: Send a command to jump to the 2nd (0x0002) page:

Header + 07 10 70 00 00 02 + CRC

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

Figure 10-27: Command Example of Page Register

10.1.2.2. Brightness Register – 0x7001

Example: Send a command to adjust the brightness setting to 45 (0x002D):

Header + 07 10 70 01 00 2D + CRC

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

Figure 10-28: Command Example of Brightness Register

10.1.2.3. Time Register – 0x7002 ~ 0x7007

0x7002: Year, ranging from 00 ~ 99.

0x7003: Month, ranging from 01 ~ 12

0x7004: Day, ranging from 01 ~ 31

0x7005: Hour: ranging from 00 ~ 23

0x7006: Minute, ranging from 00 ~ 59

0x7007: Second, ranging from 00 ~ 59

Example: Send a command to adjust the time to 2010/10/10/10:10:10

Header + 11 10 70 02 00 0A 00 0A 00 0A 00 0A 00 0A 00 0A + CRC

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

Figure 10-29: Command Example of Time Register

Note: Updating time through Uart commands does not need to write any value to the register 0x7008 to confirm the operation, but must start updating data from register 0x7002.

10.1.2.4. Wav Control Register – 0x700A

0x0000: Stop playing the audio

Header + 07 10 70 0A 00 00 + CRC

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

Figure 10-30: Command Example of Time Register (1)

0x0001: Play the 2nd audio (0001.bin) in the folder

Header + 07 10 70 0A 00 02 + CRC

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

Figure 10-31: Command Example of Time Register (2)

0x8002: Play the 2nd audio (0001.bin) in the folder repeatedly

Header + 07 10 70 0A 80 02 + CRC

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

Figure 10-32: Command Example of Time Register (3)

10.1.2.5. Volume Register – 0x700B

Volume: 0 ~ 16 (16: Max. volume; 0: Min. volume)

Example: Send a command to adjust volume to level 1:

Header + 07 10 70 0B 00 01 + CRC

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

Figure 10-33: Command Example of Volume Register

10.1.2.6. RTP Calibration Register – 0x700C

Write 0x005A to execute the RTP calibration.

Command example: **Header + 07 10 70 0C 00 5A + CRC**

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

Figure 10-34: Command Example of RTP Calibration Register

10.1.2.7. Widget Trigger Register – 0x700D

See *Widget Trigger: triggerValue* for more details.

10.1.2.8. Auto Backlight Control Register – 0x700E

Write 0x0001 to enable, or 0x0000 to disable the function.

Enable: **Header + 07 10 70 0E 00 01 + CRC**

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

Figure 10-35: Command Example of Enable Auto-Backlight Function

Disable: **Header + 07 10 70 0E 00 00 + CRC**

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

Figure 10-36: Command Example of Disable Auto-Backlight Function

10.1.2.9. Dimming Value Register – 0x700F

Value range: 0 ~ 63 (Same as **Sleep** parameter)

Example: Set the brightness to 10 → **Header + 07 10 70 0F 00 0A + CRC**

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

Figure 10-37: Command Example of Dimming Value Register

10.1.2.10. Register for setting the wait- time to enter sleep mode – 0x7010

Unit: Second. Same as **Hold time** parameter.

Example: Set the sleep time to 20 seconds → **Header + 07 10 70 10 00 14 + CRC**

Select	CMD	Addr	Data	CRC	Send
☐	10	7010	00 14	FE C9	

Figure 10-38: Command Example of Setting the wait-time Register

10.1.2.11. Register for setting the Uart upgrade mode – 0x7011

Write 0xAA55 to enter the upgrade mode (designated Bootloader is required)

Example: **Header + 07 10 70 11 AA 55 + CRC**

Select	CMD	Addr	Data	CRC	Send
☐	10	7011	AA55	11 99	

Figure 10-39: Command Example of Setting the Uart upgrade mode

See *Download bin files through Uart port* for more details.

10.1.2.12. Test Screen Register – 0x7041

Writing specific numbers to this register (0x7041) can make the LCD display various colors in full screen. There are four options:

0001: Display Black → White → Red → Green → Blue → Gray shade in full screen

Command: 5A A5 07 10 70 41 00 01 6E D7

Select	CMD	Addr	Data	CRC	Send
☐	10	7041	0001	6E D7	

Figure 10-40: Command Example of Test Screen Register (1)

0002: Display Black

Command: 5A A5 07 10 70 41 00 02 2E D6

Select	CMD	Addr	Data	CRC	Send
☐	10	7041	0002	2E D6	

Figure 10-41: Command Example of Test Screen Register (2)

0003: Display White

Command: 5A A5 07 10 70 41 00 03 EF 16

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

Figure 10-42：Command Example of Test Screen Register（3）

0000: Exit the test screen mode

Command: 5A A5 07 10 70 41 00 00 AF 17

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

Figure 10-43：Command Example of Test Screen Register（4）

10.2. Read Command

Host may send a “Read command” to designated an address of UartTFT controller to retrieve the required amount of data. Developers may utilize this command to get the state of the designated widgets or registers. Once a “Read Command” is received, UartTFT controller will return a “Returned Result”, which includes the required data, and a “Returned Massage”, which explains if the CRC (Cyclic Redundancy Check) is passed or not, to the host.

UartTFT controller may return the most 248Bytes of data at a time.

The command code is 0x03 for Read Command, Returned Result, and Returned Message.

Table 10-7: Format of Read Command, Returned Result, and the Returned Message

Read Command (Sent by Host)	**Header 0xXXXX**	**Length 0xXX**	**Read Cmd 0x03**	**Address 0x0000 ~ 0x5FFF (2 Bytes)**	**Data amount (Word) 0xXXXX (2 Bytes)**		**CRC 0xXXXX**
e.g. Read 2*2Bytes from the address 0x2050	0x5AA5	0x07	0x03	0x2050	0x0002		0xEA10
Returned Result (Returned by UartTFT IC)	**Header 0xXXXX**	**Length 0xXX**	**Read Cmd 0x03**	**Address 0x0000 ~ 0x5FFF (2 Bytes)**	**Data amount (Word) 0xXXXX (2 Bytes)**	**Data (2*n Bytes)**	**CRC 0xXXXX**
e.g. Return the data read from the address 0x2050	0x5AA5	0x0B	0x03	0x2050	0x0002	0x3031 0x3233	0x3E67
CRC Pass (by UartTFT IC)	0x5AA5	0x04	0x03	0xFF			0x4100
CRC Fail (by UartTFT IC)	0x5AA5	0x04	0x03	0x00			0x0140

Read Command Format: **0x03 Address Data (Word).** See the example shown below**:**

Figure 10-44: Read Command Example (1)

Returned Result Format: **Header Length 0x03 Address Data amount Data CRC**

Example:

1、Read 4 Bytes of data starting from the address of 0x0220 (which means the data of 0x0220 and 0x0221). The Read Command will be: **0x03 0x0220 0x0002**

Figure 10-45: Read Command Example (2)

2、After the "Read Command" is received, UartTFT controller returns:

0x5A 0xA5 0x04 0x03 0xFF 0x4C 0x30 → Returned Message, which explains the CRC is passed.

0x5A 0xA5 0x0B 0x03 0x0220 0x0002 0xC0D6 0xC9FD 0x8D 0xF2. → Returned Result, which includes the returned data, 0xC0D6 and 0xC9FD.

Note:

1、If a Uart debugging tool other than UI_Debugger-III is used, the write commands must be complete, including **Header**, **Length**, and **CRC** data, as described in Table 10-2

2、The returned messages listed in the above table are fixed formats. If CRC is passed, the returned data will be 0xFF, otherwise, the returned data will be 0x00.

10.3. Touch Returned Message

Touch Returned Message is a returned message from the UartTFT controller when the touch panel is operated. The widgets with reportToHost parameter can respond to touch operations. If the reportToHost parameter of a widget is enabled, when the widget is touched, a preset returnValue (user-defined) will be reported to the host. Host may therefore know which widget is touched. To enable reportToHost, refer to the below figure.

Figure 10-46: Enable reportToHost

The command code is 0x41 for Touch Returned Message. See the format below:

Table 10-8: Format of Touch Returned Message

Touch Returned Message	Header 0xXXXX	Length 0xXX	Command 0x41	Address (registers) 0xXXXX	returnValue 0xXXXX	CRC 0xXXXX
Touch Returned Message	0x5AA5	0x07	0x41	0xFFFF	0x0011	0xD827

Widgets with reportToHost parameter are listed below:

Table 10-9: Widgets with reportToHost parameter

Widget Name	Header (2 Bytes)	Length (1 Bytes)	Command (1 Bytes)	Address/Registers (2 Bytes)	returnValue / Data	CRC (2 Bytes)
Page Slide to Jump	0x5AA5	0x07	0x41	0xFFFF	returnValue 0xXXXX (2 Bytes)	0xXXXX
Page Slide to Jump (with effect)		0x07	0x41	0xFFFF	returnValue 0xXXXX (2 Bytes)	
Button		0x07	0x41	0xFFFF	returnValue 0xXXXX (2 Bytes)	
Popupbox		0x07	0x41	0xFFFF	returnValue 0xXXXX (2 Bytes)	
Variable Button		0x07	0x41	Address/Registers 0xXXXX	Data 0xXXXX (2 Bytes)	
Slider Bar		0x07	0x41		Data 0xXXXX (2 Bytes)	
SlideMenu		0x07	0x41		Data 0xXXXX (2 Bytes)	
Circular Touch		0x07	0x41		Data 0xXXXX (2 Bytes)	
Numeric Keypad		0xXX	0x41		Data 0xXXXX……0xXXXX (2*n Bytes)	
EN_Keyboard		0xXX	0x41		Data 0xXXXX…0xXXXX (2*n Bytes)	
CN_Keyboard		0xXX	0x41		Data 0xXXXX……0xXXXX (2*n Bytes)	
Timer		0x07	0x41	Target Address 0xXXXX	Value 0xXXXX (2 Bytes)	
Automatic Variable		0x07	0x41	Target Address 0xXXXX	Value 0xXXXX (2 Bytes)	
Extend Button		0x07	0x41	0xFFFF	returnValue 0xXXXX (2 Bytes)	
Combo Button		0x07	0x41	0xFFFF	returnValue 0xXXXX (2 Bytes)	
Encoder Mode 1 & 2		0x07	0x41	Address/Register 0xXXXX	Data 0xXXXX (2 Bytes)	
Encoder Mode3		0x07	0x41	0xFFFF	Page number 0xXXXX (2 Bytes)	

10.4. CRC – Code Example

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



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


10.4.1. CRC Calculation for Write/Read Command

Table 10-10: CRC Calculation for Write/Read Command

Type	Header	Length	Command	Address	Data/Data amount	CRC
Write Command	0xXXXX	0xXX	0x10	0~~0x7FFF (2 Bytes)	Write Data 0xXXXX……0xXXXX (2*n Bytes)	CRC 0xXXXX
Read Command	0xXXXX	0xXX	0x03	0~~0x7FFF (2 Bytes)	The number of bytes to be read 0xXXXX (2 Bytes)	CRC 0xXXXX

As shown in the above table, the data of the green part will be used for calculating CRC,

For Write Command（0x10）: Data to be used for calculating CRC → **Command Address Data**

For Read Command（0x03）: Data to be used for calculating CRC → **Command Address Data amount**

10.4.2. CRC Calculation for Returned Result of Read Command

Table 10-11: CRC Calculation for the Returned Result of a Read Command

Returned Result	Header	Length	Command	Address	Data amount	Returned Data	CRC
Returned Result	0xXXXX	0xXX	0x03	0~~0x7FFF (2 Bytes)	n 0xXXXX (2 Bytes)	0xXXXX (2*n Bytes)	CRC 0xXXXX

As shown in the above table, the data of the green part will be used for calculating CRC,

Data to be used for calculating CRC → **Command Address Data amount (n Word) Returned Data**

10.4.3. CRC Calculation for Touch Returned Message

The below table shows the 4 types of touch returned message. See *Touch Returned Message* for more information.

Table 10-12: CRC Calculation for Touch Returned Message – 4 Types

Header 0x5AA5	Length 0x07	Command 0x41	Address 0xFFFF	returnValue/Data 0xXXXX	NULL	CRC 0xXXXX
Header 0x5AA5	Length 0x07	Command 0x41	Address/Register 0xXXXX (2 Bytes)	Data 0xXXXX (2 Bytes)	NULL	CRC 0xXXXX
Header 0x5AA5	Length 0xXX	Command 0x41	Address/Register 0xXXXX (2 Bytes)	Data 0xXXXX……0xXXXX (2*n Bytes)		CRC 0xXXXX
Header 0x5AA5	Length 0xXX	Command 0x41	(Address/Register + Data) * 8 sets 0xXXXX 0xXXXX……0xXXXX 0xXXXX (4*8 Bytes)			CRC 0xXXXX

As shown in the above table, the data of the green part will be used for calculating CRC.

10.5. Modify Widget Parameter

Host may modify the parameters of a widget by “Write Command”. Simply update the data in parameterAddr, the address of the widget parameters, to modify the parameters such as font color, background color, and text content etc. Refer to Table 10-13 for the widgets with parameterAddr (Y: with; N: without). Note that the modified data will not be saved once power off. In addition, if the updated data is out of the designed range, it may cause abnormal display. Therefore, developers should implement this function with caution.

Table 10-13: Widgets with parameterAddr

Touch widgets	parameterAddr	Display widgets	parameterAddr
Button	N	String_Label	Y
SlideMenu	N	Text Scroll	Y
Popupbox	N	Text Number Display	Y
Variable Button	N	Graphics Number Display	Y
Multi-Variable Button	N	Analog Clock	Y
Circular Touch	N	Digital Clock	Y
Slider Bar	N	Timer	Y
Numeric Keypad	N	Gif	Y
CN_Keyboard	N	QRCode	Y
EN_Keyboard	N	Needle	Y
SingleKey	N	Bit Status	Y
Encoder	N	Automatic Variable	Y
Audio Play	N	Icon	Y
State Button	N	Trend Graph	Y
Static Text	N		
Extend Button	N		

10.5.1. parameterAddr

Since parameterAddr and writeAddr share the same RAM spaces, developers should make sure each of them has enough room for data allocation, and is not overlapped with others. Refer to Table 10-14 for the data length needed by various widgets.

Table 10-14: Data Length of Various Widget parameterAddrs

Widget Name	Data Length/Bytes	Occupied Spaces
String_Label	32	parameterAddr+0x0010
Text Number Display	22 + N	parameterAddr+0x000B+N/2
Text Scroll	29	parameterAddr+0x000F
Graphics Number Display	19	parameterAddr+0x000A
Analog Clock	27	parameterAddr+0x000E
Digital Clock	12	parameterAddr+0x0006
Timer	22	parameterAddr+0x000B
Gif	31	parameterAddr+0x0010
QRCode	10	parameterAddr+0x0005
Bit Status	15	parameterAddr+0x0008
Icon	18	parameterAddr+0x0009
Automatic Variable	38	parameterAddr+0x0013
Trend Graph	21	parameterAddr+0x000B
Needle	60	parameterAddr+0x001E

Note:

1、N means the data length of the unitSymbol

2、After the content of parameterAddr is updated, the widget must be refreshed in order to show the updated result.

10.5.2. String_Label: parameterAddr

The parameter arrangement and relative addresses of each parameter for the String_Label widget are shown in the table below. Each address corresponding to two bytes of data.

Table 10-15: String_Label Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
Xs	2	Changeable	0x0001
Ys	2	Changeable	0x0002
Xe	2	Changeable	0x0003
Ye	2	Changeable	0x0004
fontColor	4	Changeable	0x0005 0x0006
_color	4	Changeable	0x0007 0x0008
fontID	2	Changeable	0x0009
wordLength	2	Changeable	0x000A
other	10	Unchangeable	0x000B ~ 0x0010

The parameter addresses in the following descriptions are using 0x2000 as an example, i.e., parameterAddr=0x2000. Users must configure the parameter address according to the specific project requirements when utilizing this function.

Parameter Name	RGB565	RGB888	Absolute Address
writeAddr	0x0300	0x0300	0x2000
Xs	0x0146	0x0146	0x2001
Ys	0x0039	0x0039	0x2002
Xe	0x020E	0x020E	0x2003
Ye	0x009D	0x009D	0x2004
fontColor	0xFF57 0x0000	0xFFFF 0x5500	0x2005 0x2006
_color	0x40FD 0x0000	0x00AA 0xFF00	0x2007 0x2008
fontID	0x0002	0x0002	0x2009
wordLength	0x0014	0x0014	0x200A

Parameter	Data
name	label_0
parameterAddr	0x2000
writeAddr	0x0300
wordLength	20
X	326
Y	57
W	200
H	100
fontWidth	32
fontHeight	32
fontID	00_Font_1bit-...
encoding	GB2312
alignment	Left
Horizontal	10
Vertical	10
backgroundC...	Enable
_color	0xFFAA00
fontColor	0x55FFFF
defaultText	label_0
passwordMode	Disable
multiLanguage	Disable

Figure 10-47: parameterAddr vs. Widget Parameters

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

1、writeAddr: The variable address of the String_Label widget

2、Xs、Ys: The left-top coordinate of the widget

3、Xe、Ye: The right-bottom coordinate of the widget, where Xe(Ye) = Xs(Ys) + W(H) -1

4、fontColor:

As shown in Figure 10-47, the original color is set as 0x55FFFF (RGB888). The data of "B" color will be stored first, followed by "G", and finally "R" color. For the example here, 0xFFFF is stored in 0x2005, and 0x5500 is stored in 0x2006. If the UI project is set to aRGB4444, then the color data have to be converted first as below:

	R	G	B
RGB888 (Hex)	: 55	FF	FF
RGB888 (Bin)	: **0101 0**101	**1111 11**11	**1111 1**111 (将红色的 0 或 1 去掉)
RGB565 (Bin)	: **01010**	**111111**	**11111**
RGB565 (Hex)	: 57FF		

Next, swap the converted RGB565 data (57FF → FF57), and then add another byte of 0x0000 to the end of the color data (FF57 → FF570000). See the example shown in the left table of Figure 10-21.

Example: To modify the font color to "Blue" (RGB888), the "Write Command" will be as below:

0x10 (parameterAddr+0x0005) 0x00 0xFF 0x00 0x00

5、_color: The description is the same as fontColor above.

6、fontID: The ID of the font.

To modify fontID, developers must make sure that the font library of the new fontID is available in the UartTFT-V3_Flash.bin file. Note that if the related settings (e.g. fontWidth, fontHeight, and encoding etc.) of the new fontID is different from the original fontID, then the display effect will be different too.

7、wordLength: The data length of the string.

Example – Modify the Parameters of a String_Label Widget (take Figure 10-47 as an example)

1、Modify the writeAddr through a Uart command

To do: Change the "writeAddr" from 0x0300 to 0x1234

The Uart command is as shown below:

Select	CMD	Addr	Data	CRC	Send
☐	10	2000	3412	78 CE	

Figure 10-48: Modify the writeAddr

As the red box shown in the above figure, the lower byte of the data must be placed before the higher byte of the data.

2、Modify the widget location through a Uart command

To do: Change the widget location from (326, 57) to (0, 0)

The Uart command is as shown below:

Select	CMD	Addr	Data	CRC	Send
☐	10	2001	0000 0000 c800 6400	4A 7E	

Figure 10-49: Modify the widget location

Each code is explained as following:

10: Command type. 10 represents Write command

2001: Start address for writing the data

0000: new "Xs" value to be written to 0x2001, lower byte first.

0000: new "Ys" value to be written to 0x2002, lower byte first.

C800: new “Xe” value to be written to 0x2003, lower byte first.

6400: new “Ye” value to be written to 0x2004, lower byte first

The original Xe value is 0x020E, since the coordinate is changed from (**326**, 57) to (**0**, 0), the updated Xe value should be 0x020E – 326(decimal) = 0x00C8. As the data are stored in a lower byte first order, the new value of Xe becomes 0xC800.

The original Ye value is 0x009D, since the coordinate is changed from (326, **57**) to (0, **0**), the updated Ye value should be 0x009D – 57(decimal) = 0x0064. As the data are stored in a lower byte first order, the new value of Ye becomes 0x6400.

3、Modify the font color through a Uart command

To do: Modify the font color as shown in the below figure,

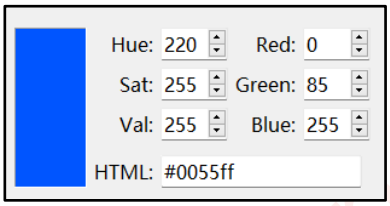


Figure 10-50：Modify the font color

Assume that the color mode is set as RGB565, therefore the above color data (0x0055FF, RGB888) must be first converted to RGB565, which is 0x02BF in the case here. Next, writing the data in a lower byte first order, that is, 0xBF02, and then add 0000 to complete the fontColor update.

CMD	Addr	Data	CRC	Send
10	2005	BF 02 00 00	45 B5	

Figure 10-51：Modify the color of the font

10.5.3. Text Number Display: parameterAddr

Table 10-16: Text Number Display Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
X	2	Changeable	0x0001
Y	2	Changeable	0x0002
fontColor	4	Changeable	0x0003 0x0004
fontID	2	Changeable	0x0005
other	10+N	Unchangeable	0x0006 ~ 0x000B+N/2

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

writeAddr: The variable address of the Text Number Display widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

fontColor: The color of the font

fontID: The ID of the font

Note: To modify the Text Number Display parameters, refer to the String_Label example described above.

10.5.4. Text Scroll: parameterAddr

Table 10-17: Text Scroll Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
Xs	2	Changeable	0x0001
Ys	2	Changeable	0x0002
Xe	2	Changeable	0x0003
Ye	2	Changeable	0x0004
wordLength	2	Changeable	0x0005
other	17	Unchangeable	0x0006 ~ 0x000F

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

writeAddr: The variable address of the Text Scroll widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

wordLength: The data length of the string.

10.5.5. Graphics Number Display: parameterAddr

Table 10-18: Graphics Number Display Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
X	2	Changeable	0x0001
Y	2	Changeable	0x0002
firstIcon	2	Changeable	0x0003
lastIcon	2	Changeable	0x0004
other	9	Unchangeable	0x0005 ~ 0x000A

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

writeAddr: The variable address of the Graphics Number Display widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

firstIcon & lastIcon: To modify these parameters, note that (1) the new pictures have to be included in the current UartTFT-V3_Flash.bin; (2) the designated Icon number (4-digit decimal number) has to be converted to hexadecimal number; (3) if the new picture width or height is different from the original one, it may result in abnormal display.

10.5.6. Analog Clock: parameterAddr

Table 10-19: Analog Clock Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
X	2	Changeable	0x0000
Y	2	Changeable	0x0001
background	2	Changeable	0x0002
other	21	Unchangeable	0x0003 ~ 0x000E

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

background: The ID of the background picture

10.5.7. Digital Clock: parameterAddr

Table 10-20: Digital Clock Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
X	2	Changeable	0x0000
Y	2	Changeable	0x0001
firstIcon	2	Changeable	0x0002
lastIcon	2	Changeable	0x0003
other	4	Unchangeable	0x0004 ~ 0x0006

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

writeAddr: The variable address of the Digital Clock widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

firstIcon & lastIcon: To modify these parameters, note that (1) the new pictures have to be included in the current UartTFT-V3_Flash.bin; (2) the designated Icon number (4-digit decimal number) has to be converted to hexadecimal number; (3) if the new picture width or height is different from the original one, it may result in abnormal display.

10.5.8. Timer: parameterAddr

Table 10-21: Timer Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
presetAddr	2	Changeable	0x0000
countAddr	2	Changeable	0x0001
controlAddr	2	Changeable	0x0002
X	2	Changeable	0x0003
Y	2	Changeable	0x0004
firstIcon	2	Changeable	0x0005
other	10	Unchangeable	0x0006 ~ 0x000B

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

presetAddr: The address of the preset value

countAddr: The address of the counting value

controlAddr: The control address of the counter

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

firstIcon: To modify this parameter, make sure that the new pictures are included in the UartTFT-V3_Flash.bin

10.5.9. GIF: parameterAddr

Table 10-22: GIF Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
X	2	Changeable	0x0001
Y	2	Changeable	0x0002
W	2	Changeable	0x0003
H	2	Changeable	0x0004
gifName	2	Changeable	0x0005
interval(10ms)	2	Changeable	0x0006
other	17	Unchangeable	0x0007 ~ 0x0010

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

writeAddr: The variable address of the GIF widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

gifName: To modify this parameter, developers should make sure the new Gif has been included in the current UartTFT-V3_Flash.bin, and the W & H parameters should also be updated accordingly to avoid abnormal display.

W、H: The width and height of the GIF widget

Interval: The display interval between frames

10.5.10. QRCode: parameterAddr

Table 10-23: QRCode Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
byteLength	2	Changeable	0x0001
X	2	Changeable	0x0002
Y	2	Changeable	0x0003
other	2	Unchangeable	0x0005

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

writeAddr: The variable address of the QRCode widget.

byteLength: The data length of the QRCode.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

10.5.11. Bit Status: parameterAddr

Table 10-24: Bit Status Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
X	2	Changeable	0x0001
Y	2	Changeable	0x0002
other	9	Unchangeable	0x0003 ~ 0x0008

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

writeAddr: The variable address of the Bit Status widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

10.5.12. Icon: parameterAddr

Table 10-25: Icon Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
X	2	Changeable	0x0001
Y	2	Changeable	0x0002
firstIcon	2	Changeable	0x0003
lastIcon	2	Changeable	0x0004
minDisplayID	2	Changeable	0x0005
maxDisplayID	2	Changeable	0x0006
other	4	Unchangeable	0x0007 ~ 0x0009

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

writeAddr: The variable address of the Icon widget.

X: The left-top X coordinate of the widget

Y: The left-top Y coordinate of the widget

firstIcon、lastIcon: If these parameters are not set in UI_Editor-III, then the content will be 0xFFFF. Otherwise, the content will be the icon number (Hexadecimal).

minDisplayID、maxDisplayID: The minimum and maximum ID of the Icon widget

10.5.13. Automatic Variable: parameterAddr

Table 10-26: Automatic Variable Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
targetAddr	2	Changeable	0x0000
other	36	Unchangeable	0x0001 ~ 0x0013

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

targetAddr: The target address of the Automatic Variable widget

10.5.14. Trend Graph: parameterAddr

Table 10-27: Trend Graph Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
Xs	2	Changeable	0x0000
Ys	2	Changeable	0x0001
Xe	2	Changeable	0x0002
Ye	2	Changeable	0x0003
y_ReferenceLine	2	Changeable	0x0004
_referenceValue	2	Changeable	0x0005
x_Spacing(Pixels)	2	Changeable	0x0006
other	7	Unchangeable	0x0007 ~ 0x000B

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the "Absolute address".

Xs & Ys: The left-top coordinate of the widget.

Xe & Ye: The right-bottom coordinate of the widget → Xe (Ye) = X (Y) + W (H) To modify the widget location, the coordinates of Xs, Ys, Xe, and Ye must all be updated.

10.5.15. Needle: parameterAddr

Table 10-28: Needle Parameters

Parameter Name	Data Length/Bytes	Feature	Relative Address
writeAddr	2	Changeable	0x0000
background	2	Changeable	0x0001
X	2	Changeable	0x0002
Y	2	Changeable	0x0003
W	2	Changeable	0x0004
H	2	Changeable	0x0005
pivot_X	2	Changeable	0x0006
pivot_Y	2	Changeable	0x0007
startAngle	2	Changeable	0x0008
finalAngle	2	Changeable	0x0009
other	40	Unchangeable	0x000A ~ 0x001E

Absolute Address = parameterAddr + Relative Address. To modify a parameter through the Uart port, the designated address must be the “Absolute address”.

writeAddr: The variable address of the Needle widget.

Background: To modify this parameter, developers should make sure the designated picture has been included in the current UartTFT-V3_Flash.bin.

X & Y: When modifying the widget location, the coordinates of pivot_X, pivot_Y, _promptNum_X, and _promptNum_Y must all be updated.

W & H: The width and height of the background picture. If the background picture is changed, these two parameters must be modified according to the new background picture.

pivot_X & pivot_Y: The coordinate of the meter center.

startAngle & finalAngle: If needleType is set as Animation, then these two parameters are unchangeable.

10.6. Widget Trigger: triggerValue

Table 10-29 shows the list of widgets that can be triggered by Host through the parameter, triggerValue:

Table 10-29: Widgets that can be triggered by Host

Widget Name	Triggered by Host (Y/N)	Widget Name	Triggered by Host (Y/N)
Button	Y	Digital Clock	N
Popupbox	Y	GIF	N
Variable Button	Y	QRCode	N
Multi-Variable Button	Y	Audio Play	N
State Button	Y	Bit Status	N
Extend Button	Y	Icon	N
Numeric Keypad	Y	Trend Graph	N
CN_Keyboard	Y	Encoder	N
EN_Keyboard	Y	Static Text	N
SingleKey	N	Automatic Variable	N
String_Label	N	Needle	N
Text Scroll	N	Analog Clock	N
Text Number Display	N	Graphics Number Display	N
SlideMenu	N	Timer	N
Circular Touch	N	Slider Bar	N

Y: Supported; **N:** Not supported

As mentioned above, Host may send a designated value to a widget to trigger designed operation(s). For example, a button widget whose [pageGoto] is set as Page0001, [hostControl] is set to Enable, and [_triggerValue] is set as 0x0001. When Host sends 0x0001 to Widget Trigger Register (0x700D), UartTFT controller will execute the preset operation which is “jump to Page0001”.

Note:

1、Once [hostControl] is enabled, touch control will be invalid

2、All _triggerValue should be set to different values from each other

3、Once [hostControl] is enabled, the widget will not be displayed no matter pictures are assigned to the widget or not.

4、The available setting range of _triggerValue is 0x0001 ~ 0xFFFF.

10.7. HID and Vcom

HID and Vcom communication protocols share the same commands used in Uart protocol. To apply HID and Vcom communication protocols on Levetop development board, a designated MCU code is needed.

10.7.1. HID – Hardware Configuration & Software settings

To apply HID communication protocol, follow the below steps:

1. In the MCU source code, set [UARTBUS_OPTION] to 3, and then generate a new MCU_Code.bin file. Program the MCU_Code.bin to LT7589.

```
module_select.h
13
14  #define IIC_BUS              0       // Communication,
15  #define UARTBUS_OPTION       3       // Communication,
16  #define DualFlash            0
17  #define encoder_on           0       // 1:Open 0:Close
18  #define Touch_selection      0       // 0:CTP  1:RTP
```

Figure 10-52: Select HID Mode

2. As shown in the follow figure,
   (1) Weld 22 ohm resistors at the location of R24 and R30
   (2) Use the USB port (CON9) to connect with the computer.

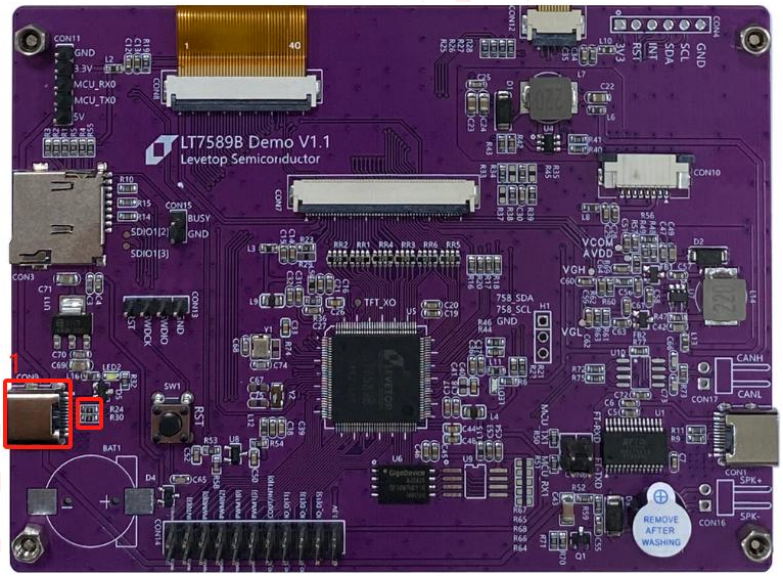
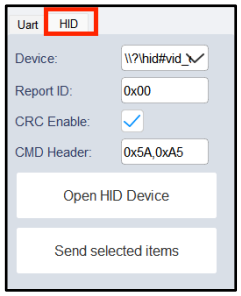


Figure 10-53: Hardware Configuration

3. Activate the software tool, UI_Debugger-III. Click on HID page, and then click on [OpenHID Device]. Refer to below figure:



10.7.2. Vcom – Hardware Configuration & Software Settings

To apply Vcom communication protocol, follow the below steps:

1. In the MCU source code, set [UARTBUS_OPTION] to 4, and then generate a new MCU_Code.bin file. Program the MCU_Code.bin to LT7589.

```
module_select.h*
13
14  #define IIC_BUS              0       // Communication,
15  #define UARTBUS_OPTION       4       // Communication,
16  #define DualFlash            0
17  #define encoder_on           0       // 1:Open 0:Close
18  #define Touch_selection      0       // 0:CTP  1:RTP
```

Figure 10-54: Select Vcom Mode

2. As shown in the follow figure,
   (1) Weld 22 ohm resistors at the location of R24 and R30
   (2) Use the USB port (CON9) to connect with the computer.

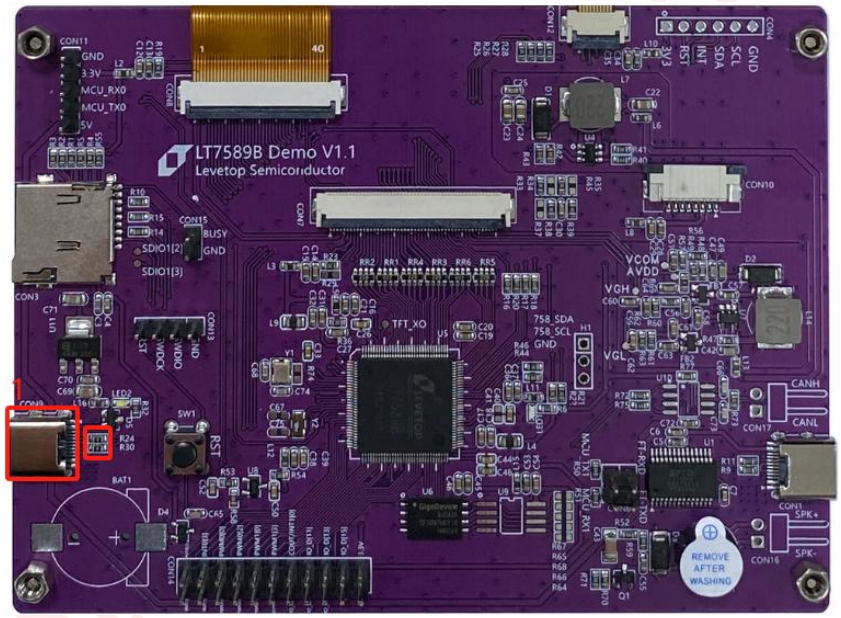
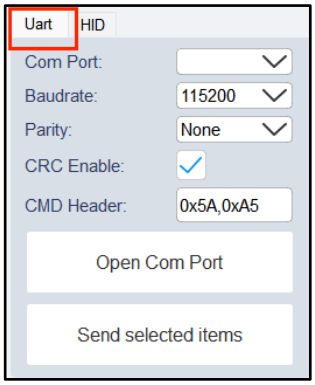


Figure 10-55: Hardware Configuration

3. Activate the software tool, UI_Debugger-III. Click on Uart page and select the connected COM port. Finally, click on [Open Com Port]. Refer to below figure:



11. ModBus

Developers may apply ModBus protocol instead of UartTFT controller protocol. When a UartTFT controller is used as the master device, it can send commands through the Device Addr of ModBus to slave devices. When a UartTFT controller is used as a slave device, it can receive commands from the master device.

Levetop applies standard Modbus protocol and supports RTU mode.

When Modbus protocol is used,

1. UartTFT controller protocol is not valid.

2. UartTFT controller supports Register and Coil operation if it is used as the Master.

3. UartTFT controller only supports Register operation if it is used as a Slave.

11.1. Create a ModBus Command File

Note: Refer to this section if the UartTFT controller is used as the Master.

The name of the ModBus command list is [**command.list**] which is a TXT file with a suffix of [**.list**]. The command.list file has to be saved under the project directory, as shown in Figure 11-1. Developers may create a command.list file by (1) adding a new TXT file and then rename it to **command.list**; or (2) export a **command.list** file by clicking on [**Save Cmdlist**] button.

Figure 11-1: Create a ModBus Command File

Note: When a UartTFT controller is acted as a Master, there must be one and only one command.list file.

11.2. ModBus Command Setting Page

Note: Refer to this section if the UartTFT controller is used as the Master.

Click on the Tool menu, and select [Modbus] to enter Modbus command setting page, as shown below:

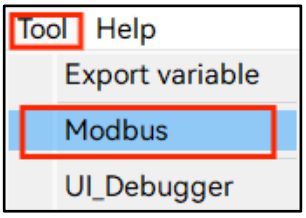


Figure 11-2: Enter ModBus Command Setting Page

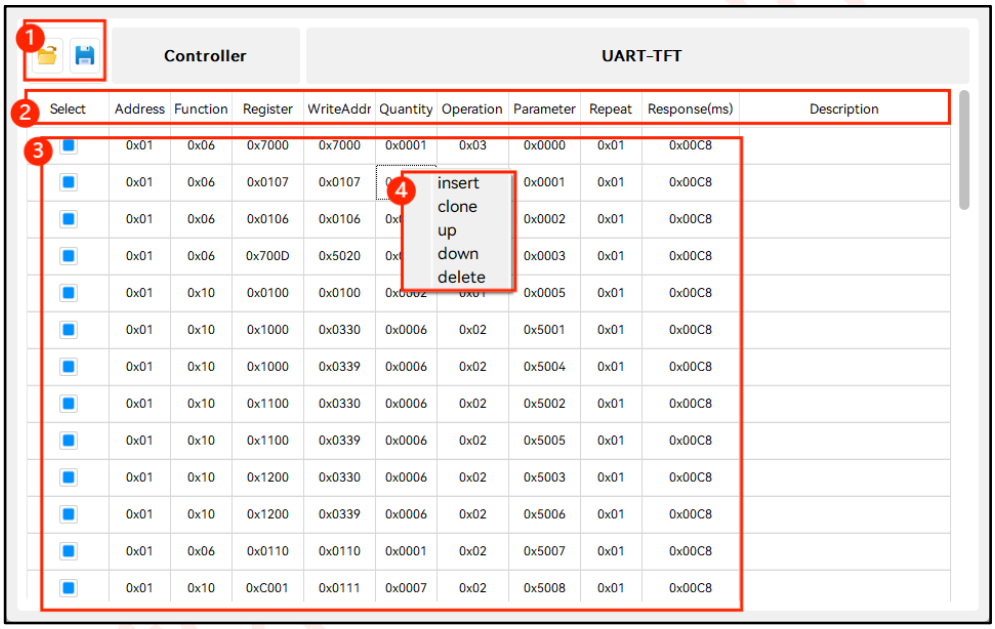


Figure 11-3: ModBus Command Editing Page

❶ Click on  to import the command.list. Click on  to save as the command.list

❷ **Command composition:**

Select	:	Only checked commands will be included when exporting UartTFT-II_Flash.bin. Each checked item must be a complete command.
Address	:	The address of the slave device. The available range is 0x01 ~ 0xFF.
Function	:	Function codes.
Register	:	The register/coil address (starting address of Write/Read operation) of the slave.
WriteAddr	:	The variable starting address of Write/Read operation of the Master
Quantity	:	The number of the coils / registers. Unit: byte. A register = 2bytes.

Operation	:	Operation mode, 4 options.
Parameter	:	This parameter should be set based on the setting of Mode mentioned above.
Repeat	:	When Master sends a command to the slave, if the salve does not respond within the response time, then the Master will send the command again. Master will send a command the most Repeat + 1 times, if there is still no response from the slave, the Master will skip the current operation and execute the next command.
Response(ms)	:	The response time after Master sends a command to the Slave. Unit: ms.

❸ **Edit Area**

❹ **Command Operation: (right click on the target command to activate the pop-up window)**

Insert	:	Compose a new command and insert it to the above of the selected command.
Clone	:	Clone the selected command.
UP	:	Move the selected command up.
Down	:	Move the selected command down.
Delete	:	Delete the selected command.

Note: To use Modbus, the edited commands must be saved by clicking on  before exporting UartTFT-V3_Flash.bin

11.3. ModBus Command Structure

Note: Refer to this section if the UartTFT controller is used as the Master.

Table 11-1: ModBus Command Structure

	Slave Address	Read/Write	Parameters of Master/Slave			Command conditions		Command settings	
Name	Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
Bytes	1	1	2	2	2	1	2	1	2

Slave Address: The address of the slave device. **It must NOT be set to 0x00.**

Function Code: As shown in Table 11-2

Table 11-2: Function Code

Function Code	Function	Number of Coils/Registers
0x03	Read Multiple Registers	1~125
0x04	Read Input Register	1~125
0x06	Write Single Register	1
0x10	Write Multiple Register	1~123
0x01	Read Coils	1~2000
0x02	Read Input Discrete	1~2000
0x05	Write Single Coil	1
0x0F	Write Multiple Coils	1~1968

Slave register address : The coil address (starting address of the Write/Read operation) of the slave.

Master variable address : The variable starting address of the Write/Read operation of the Master.

Data length : The number of the coils / registers.

Repeat Times : When Master sends a command to the slave, if the salve does not respond within the response time, then the Master will send the command again. Master will send a command the most Repeat + 1 times, if there is still no response from the slave, the Master will skip the current operation and execute the next command.

Response Time : The response time after Master sends a command to the Slave. Unit: ms.

Operation Mode : Operation Mode and Parameter construct the condition of sending commands. There are 4 options as described below. Refer to *Modbus Operation Mode Setting Tutorial* for more details.

Table 11-3: Operation Modes

Operation Mode	Parameter
0x00	0x0000
0x01	Page number
0x02	Variable address
0x03	Designated number

0x00: The command is executable in all pages. Set 0x0000 to [Parameter].

0x01: Only execute the command under the designated page. Set the page number to [Parameter]. For example, set 0x0003 to [Parameter] to designate Page0003

0x02: Only execute the command when the data of the variable address is 0x4C54. Set the variable address to [Parameter].

0x03: Customization mode. Only execute the command if the designated location is set to 1. Set the designated location in [Parameter]. Each location represents a fixed operation. When the Master detects that the designated location is set to 1, it will then send the corresponding command to the Slave.

11.4. ModBus Command

During Modbus communication, when Master sends a command, Slave will then respond accordingly. Unlike usual serial communication protocol, a Modbus command does not need to include the contents of the data, except for the variable addresses of both Master and Slave, and the data length. The content of the data is retrieved from the designated variable address. Each command of the command list will be checked and if it meets the command conditions (command mode & command parameter), it will be sent out. Otherwise the command will be skipped.

Note: When the Master sends multiple commands, an interval time must be added between the commands.

11.4.1. Example: Master Request Slave for Data

Function Code: 0x03 – Master reads single/multiple registers data from Slave

Table 11-4: Master Request Slave for Data

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x03	0x0000	0x0020	0x0009	0x00	0x0000	0x05	0x00C8

This command will be sent to the Slave whose address is 0x01. The Slave will then respond to the Master with the data stored in the registers whose addresses are from 0x0000 to 0x0008. Master will then store the received data to the addresses from 0x0020 ~ 0x0028.

A command example is as shown below:

Master send	Slave address (1 Byte)	Function code (1 Byte)	Register address (2 Bytes)	Data amount (Word) to read (2 Bytes)	CRC (2 Bytes)
	0x01	0x03	0x0000	0x0009	0x85 0xcc
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Returned data length (1 Byte)	Returned data (2*n Bytes)	CRC (2 Bytes)
	0x01	0x03	0x12	0x0001 0x0002 0x0003 0x0004 0x0005 0x0006 0x0007 0x0008 0x0009	0x9c 0xb4

11.4.2. Example: Master Read Input Register

Function Code: 0x04 – Master reads input register data from Slave

Table 11-5: Master Read Input Register

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x04	0x0000	0x0020	0x0009	0x00	0x0000	0x05	0x00C8

Command 0x04 is used by the Master to read input register from the Slave.

A command example is as shown below:

Master send	Slave address (1 Byte)	Function code (1 Byte)	Register address (2 Bytes)	Datat amount (Word) to read (2 Bytes)	CRC (2 Bytes)
	0x01	0x04	0x0000	0x0009	0x30 0x0C
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Returned data length (1 Byte)	Returned data (2*n Bytes)	CRC (2 Bytes)
	0x01	0x04	0x12	0x0001 0x0002 0x0003 0x0004 0x0005 0x0006 0x0007 0x0008 0x0009	0x29 0x03

11.4.3. Example: Master Write Single Input Register

Function Code: 0x06 – Master writes data to single register of Slave

Table 11-6: Master Write Single Input Register

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x06	0x0000	0x0020	0x0001	0x00	0x0000	0x05	0x00C8

This command will assign the 2 bytes data stored in the designated address (0x0020) of the Master to the Slave whose address is 0x01. The data will be stored to the Slave register whose address is 0x0000.

A command example is as shown below:

Master send	Slave address (1 Byte)	Function code (1 Byte)	Register address (2 Bytes)	Data to write (2 Bytes)	CRC (2 Bytes)
	0x01	0x06	0x0000	0x0000	0x89 0xca
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Register address (2 Bytes)	Data to write (2 Bytes)	CRC (2 Bytes)
	0x01	0x06	0x0000	0x0000	0x89 0xca

11.4.4. Example: Master Write Multiple Registers

Function Code: 0x10 – Master writes data to multiple registers of Slave

Table 11-7: Master Write Multiple Registers

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x10	0x0000	0x0000	0x0009	0x00	0x0000	0x05	0x00C8

This command will assign the 18 bytes (data length: 0x0009) of data stored in the Master variable addresses from 0x0000 to 0x0008 to the designated Slave registers whose addresses are from 0x0000 to 0x0008.

A command example is as shown below:

Master send	Slave address (1 Byte)	Function code (1 Byte)	Register address (2 Bytes)	Register amount (Word) (2 Bytes)	Data length (1 Byte)	Data to be written (2 Bytes)	CRC (2 Bytes)
	0x01	0x10	0x0000	0x0009	0x12	0x0001 0x0002 0x0004 0x0008 0x0010 0x0020 0x0040 0x0080 0x0000	0x95 0x3c
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Register address (2 Bytes)	Register amount (Word) (2 Bytes)	NULL		CRC (2 Bytes)
	0x01	0x10	0x0000	0x0009	NULL		0x00 0x0f

11.4.5. Example: Master Read Coil Status

Function Code: 0x01 – Master reads coil status from Slave

Table 11-8: Master Read Coil Status

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x01	0x0009	0x0001	0x000A	0x00	0x0000	0x05	0x00C8

This command will read 10 (0x000A) coils status starting from the designated Slave coil address. The received data will be allocated to the Master variable address calculated as below:

(1) Slave coil address % 0x10 = 0x0009 % 0x10 = 9 → the read data will be stored to the Master variable address (0x0001), starting from bit9

(2) Since a variable address can store 2bytes (bit0~bit15) of data, the read data will be stored to the designated Master variable address (0x0001), starting from bit9 to bit15. The rest of the read data will then be stored to the Master variable address, 0x0002, from bit0~bit2.

A command example is as shown below:

Master send	Slave address (1 Byte)	Function code (1 Byte)	Coil address (2 Bytes)	Coil amount to be read (2 Bytes)	CRC (2 Bytes)
	0x01	0x01	0x0009	0x000a	0x6c 0x0f
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Returned data length (Bytes) (1 Byte)	Data (2*n Bytes)	CRC (2 Bytes)
	0x01	0x01	0x02	0xde 0x03	0xa0 0x5d

The returned data (0xde 0x03) is based on below assumptions:

(1) The status of the coils (0x0009 ~ 0x0012) is 0111 1011 11.

(2) For the status of 0x0009 ~ 0x0010 (0111 1011) is converted to 1011 0111, which is 0xde

(3) For the status of 0x0011 ~ 0x0012 (11 → 1100 0000) is further converted to 0000 0011, which is 0x03

11.4.6. Example: Master Read Input Discrete

Function Code: 0x02 – Master reads input discrete from Slave

Table 11-9: Master Read Input Discrete

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x02	0x0009	0x0001	0x000A	0x00	0x0000	0x05	0x00C8

The allocation method of the read data is the same as the one described above in “Master Read Coil Status”

A command example is as shown below:

Master send	Slave address (1 Byte)	Function code (1 Byte)	Coil address (2 Bytes)	Coil amount to be read (2 Bytes)	CRC (2 Bytes)
	0x01	0x02	0x0009	0x000a	0x28 0x0f
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Returned data length (Bytes) (1 Byte)	Data (2*n Bytes)	CRC (2 Bytes)
	0x01	0x02	0x02	0xde 0x03	0xa0 0x19

11.4.7. Master Write to Single Coil

Function Code: 0x05 – Master writes to single coil of Slave

Table 11-10: Master Write Single Coil

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x05	0x0013	0x0001	0x0001	0x00	0x0000	0x05	0x00C8

This command will write data to a designated Slave coil address. The written data is based on the content of the designated Master variable address, as explained below:

(1) Slave coil address % 0x10 = 0x0013 % 0x10 = 3 → bit3 of the Master variable address (0x0001)

(2) If bit3 = 0, then Master sends 0x0000 to Slave

If bit3 = 1, then Master sends 0xFF00 to Slave

Data other than 0x0000 and 0xFF00 are not valid, and will have no effect on coils.

A command example is as shown below. The bit3 status of Master variable address 0x0001 is 1.

Master send	Slave address (1 Byte)	Function code (1 Byte)	Coil address (2 Bytes)	Coil status (2 Bytes)	CRC (2 Bytes)
	0x01	0x05	0x0013	0xff00	0x7d 0xff
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Coil address (2 Bytes)	Coil status (2 Bytes)	CRC (2 Bytes)
	0x01	0x05	0x0013	0xff00	0x7d 0xff

11.4.8. Master Write to Multiple Coils

Function Code: 0x0F – Master writes to multiple coils of Slave

Table 11-11: Master Write to Multiple Coils

Slave Address	Function Code	Slave register address	Master variable address	Data length	Command Mode	Command Parameter	Repeat Times	Response Time
0x01	0x0F	0x0009	0x0001	0x000F	0x00	0x0000	0x05	0x00C8

This command will write data to 15 (0x000F) Slave coil addresses. The written data is based on the content of designated Master variable address, as explained below:

(1) Slave coil address % 0x10 = 0x0009 % 0x10 = 9 → bit9 of the Master variable address (0x0001)

(2) Master will send data (0x0000 or 0xFF00) to Slave, based on the content of the designated Master variable address, starting from the address 0x0001, bit9~bit15, to 0x0002, bit0~bit7.

(3) The designated Slave coil address, 0x0009, is related to bit9 of the Master variable address 0x0001; and the Slave coil address, 0x0017, is related to bit7 of the Master variable address 0x0002.

A command example is as shown below. The content of master variable address 0x0001 is 0x5400, and the content of master variable address 0x0002 is 0x0005.

Master send	Slave address (1 Byte)	Function code (1 Byte)	Coil address (2 Bytes)	Coil amount (Word) to be written (2 Bytes)	Data length (Bytes) (1 Byte)	Data to be written (2 Bytes)	CRC (2 Bytes)
	0x01	0x0f	0x0009	0x000f	0x02	0xaa 0x02	0x1a 0x0c
Slave return	Slave address (1 Byte)	Function code (1 Byte)	Coil address (2 Bytes)	Coil amount (Word) to be written (2 Bytes)	NULL		CRC (2 Bytes)
	0x01	0x0f	0x0009	0x000f	NULL		0xc5 0xcd

The written data (0xaa 0x02) is calculated by below steps:

(1) The content of Master variable address (0x0001 ~ 0x0002) is 0x5400 0x0005

(2) Swap the above data → 0x0005 0x5400

(3) Convert the data to binary form → 0000 0000 **0000 0101 0101 0100** 0000 0000 (bit0 is the first bit on the right)

(4) The above data marked in yellow, bit23~bit9, will be written to Slave coil.

(5) bit16~bit9 → 1010 1010, which is 0xaa

(6) bit23~bit17 → 0000 010. Add one '0' on the higher bit → 0000 0010, which is 0x02

11.5. Modbus Command – CRC Calculation

The whole portion (except for the CRC part) of Modbus command is used for calculating CRC. Refer to *CRC – Code Example* for more details.

11.6. Modbus Setting Example

11.6.1. Use a UartTFT panel as a Modbus slave

To use a UartTFT panel as a Modbus slave, the related UI_Editor project and the MCU_Code needs to be set accordingly.

1. Set the device address (Device Addr, **must NOT be 0x00**) in the Project Setting page, as shown below:

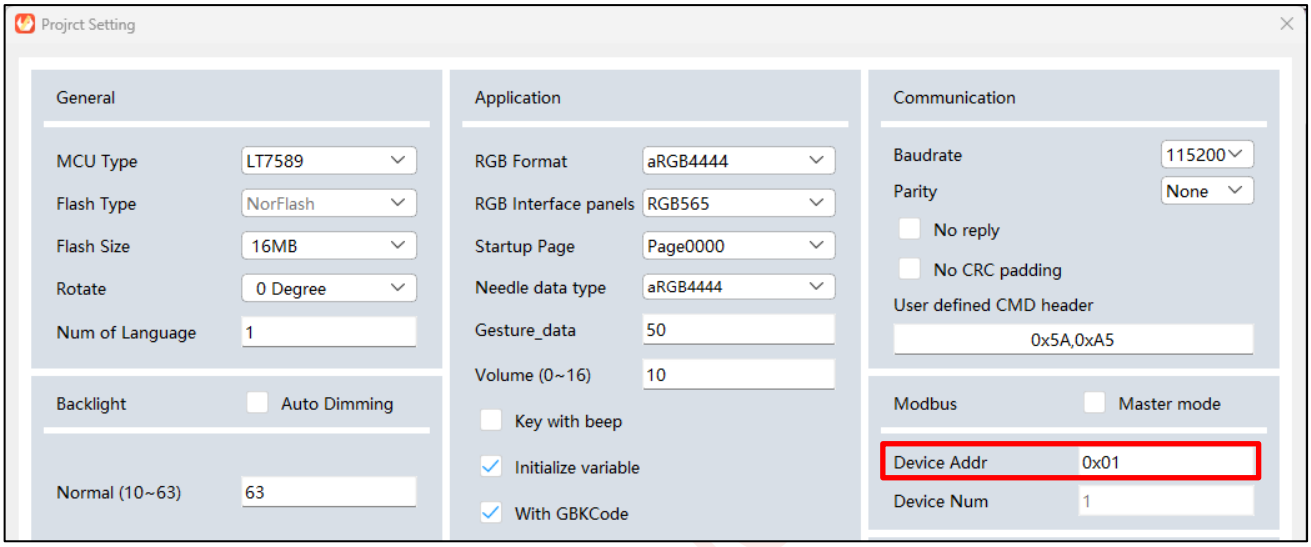


Figure 11-4: Setting Slave Mode

2. Modify MCU_Code

For LT7589, modify the macro definition in the MCU_Code as below:

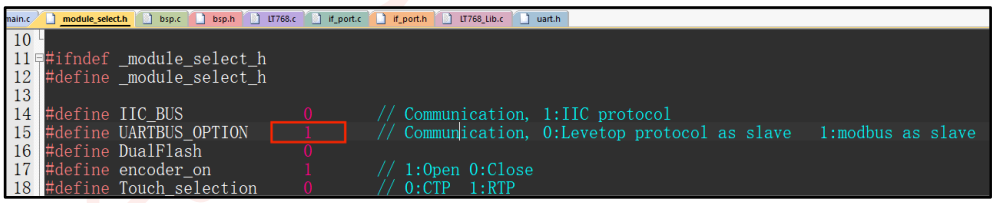


Figure 11-5: LT7589 – Set Slave Mode

11.6.2. Use a UartTFT panel as the Modbus master

To use a UartTFT panel as a Modbus master, the related UI_Editor project and the MCU_Code needs to be set accordingly.

1. Check the [Master mode] in the Project Setting page. No need to set the device address. As shown below:

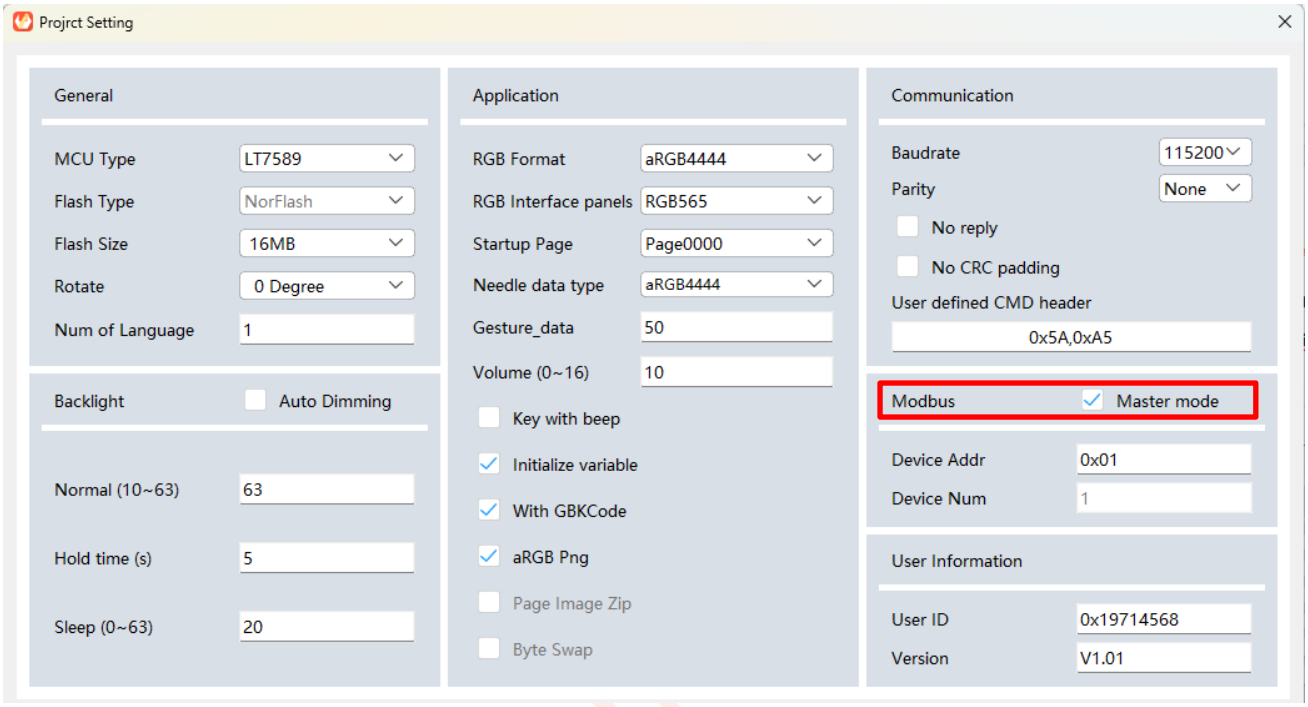


Figure 11-6: Setting Master Mode

2. Modify the MCU_Code

For LT7589, modify the macro definition in the MCU_Code as below:

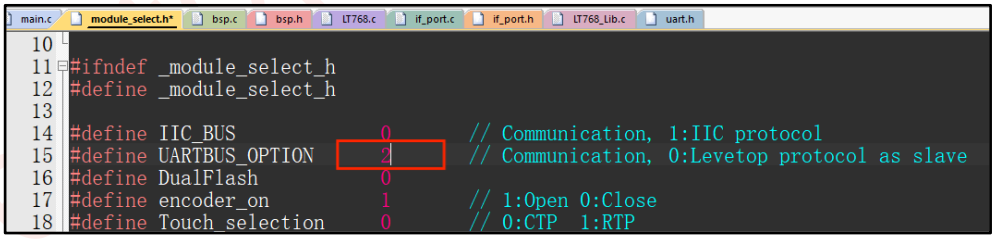


Figure 11-7: LT7589 – Set Master Mode

11.7. Modbus Operation Mode Setting Tutorial

11.7.1. Operation Mode – 0x00

No extra settings required. The command will be executed unconditionally.

11.7.2. Operation Mode – 0x01

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

Figure 11-8: Operation Mode – 0x01

As shown in the above table, the [Operation] mode is 0x01, which means the command will be executed at the display page number designated by [Parameter]. In this example, since [Parameter] is 0x0002, the command will be executed when the UartTFT panel displays page0002, as shown below:

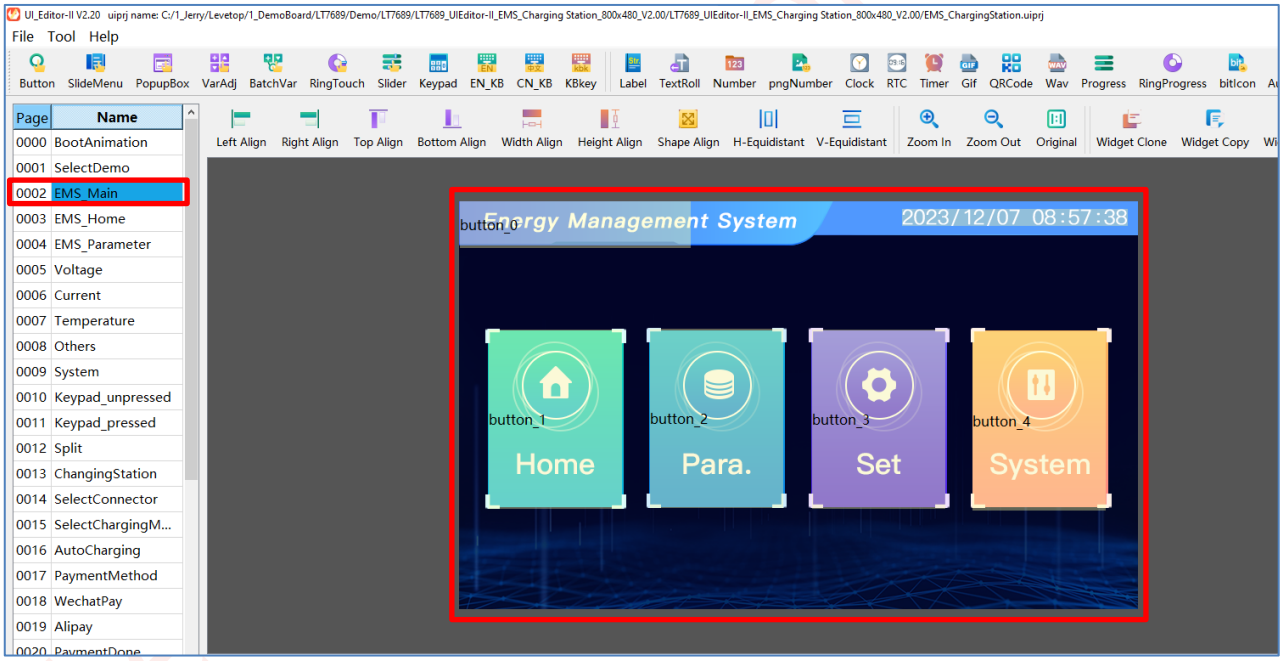


Figure 11-9: Example: Operation Mode – 0x01

11.7.3. Operation Mode – 0x02

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

Figure 11-10: Operation Mode – 0x02

As shown in the above table, the [Operation] mode is 0x02, which means the command will be executed when the content of the designated variable address is 0x4C54. The variable address is assigned to [Parameter]. In this example, [Parameter] is 0x0010, which means the command will be executed when the content of the address 0x0010 is 0x4C54. After the command is executed, the content of the address 0x0010 will be reset. Below figure shows a setting example of Multiple-Variable Button widget:

Parameter	Data
name	batVar_0
X	273
Y	335
W	263
H	100
unpressedIcon	
pressedIcon	
pageGoto	Page0017
writeAddr0	0x0010
_value	0x4C54
writeAddr1	0xFFFF
_value	0xFFFF
writeAddr2	0xFFFF
_value	0xFFFF

Figure 11-11: Setting values to the Multi-Variable widget

11.7.4. Operation Mode – 0x03

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

Figure 11-12: Operation Mode – 0x03

As shown in the above table, the [Operation] mode is 0x03, which means the command will be executed when the value of the designated location is set to 1 in the MCU_Code. The location is assigned to [Parameter], which is 0x0001 in this example.

Following is an MCU_Code setting example:

1. In the MCU_Code, locate the function: Uart_cmd_Send()

```
255 #elif (UARTBUS_OPTION == 2)
256         if (Sum_ModbusTX)
257             Uart_cmd_Send();
258
259         LT_ModBus_REG_Cmd();
```

Figure 11-13: Uart_cmd_Send()

2. Find the location array, Master_mode03_flag[]:

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

Figure 11-14: Master_mode03_flag

3. Set the value of the designated array location to 1

In this example, a button widget is used to trigger the command.

```
315         if (Gesture_flag)
316             Gesture_touch();    // gesture_no_sliding  滑动翻页
317
318         Basic_touch();          // Basic touch control
319         Adj_touch();            // Variable adjustment
320         Progress_bar_sliding(); // Sliding progress bar
321         data_input();           // Data input
322         slideMune();            // Slide menu
323         RingSld_touch();        // Ring progress bar with touch
324         Ascii_input();          // ASCII keyboard
325         GBK_input();            // GBK keyboard
```

Figure 11-15: Basic_touch()

In Basic_touch(), locate the below condition code:

If(gTpInfo.sta ==0 && Basci_flag == 1) // The button is touched and released

Set the designated location to 1 in Master_mode03_flag[]. Since Parameter is set to 0x0001, this means Master_mode03_flag[0x0001] should be set to 1. Refer to below figure:

```
12325   if (gTpInfo.sta == 0 && Basci_flag == 1)//手指抬起且有按压按键的标志
12326   {
12327       if (gBasci_Info[Basci_num].Code == 0xC001)
12328       {
12336       if (gBasci_Info[Basci_num].id != 0xFFFF)
12337       {
12340       Basci_flag = 0;
12341       button_Press_flag = 0;
12342       if (gBasci_Info[Basci_num].Next_id != 0xFFFF)
12343       {
12346       if(gBasci_Info[Basci_num].Keyvalue == 0x0022 )
12347       //仅串口返回值设置为0x0022的button执行时可进入
12348       {
12349       Master_mode03_flag[0x0001] = 1;//该标志位数组的第 Parameter 个元素置1
12350       }
12351   }
12352
```

Figure 11-16: Set the designated location to 1

In addition, to trigger the function, the [returnValue] of the Button widget has to be set the same as the setting in the MCU_Code, as shown below. When the command is executed, Master_mode03_flag[0x0001] will be reset to 0 automatically.

Parameter	Data
name	button_0
X	1
Y	11
W	130
H	74
returnValue	0x0022
unpressedIcon	
pressedIcon	
pageGoto	Page0015
reportToHost	Disable
hostControl	Disable
_triggerValue	0x0000

Figure 11-17: Add the returnValue

12. Access External SPI Flash and MCU EFlash

12.1. Commands

Table 12-1: Commands for accessing external Flash & MCU Eflash

Command	Header 0x5A 0xA5 (2 Bytes)	Fixed 0x00 (1 Byte)	Length (2 Byte)	Command Code (1 Byte)	Address (4 Bytes)	Data content (n Bytes, Max. 2048)	CRC 0xXX 0xXX (2 Bytes)
Write MCU EFlash	0x5A 0xA5	0x00	Data length +0x07	0x61	0x00001000 ~ 0x00034000	Data	CRC16
Return message	0x5A 0xA5	0x00	0x09	0x61		OK/NG	CRC16
Read MCU EFlash	0x5A 0xA5	0x00	0x09	0x62		Data amount to be read (2Bytes)	CRC16
Return message	0x5A 0xA5	0x00	Data length +0x07	0x62		Data	CRC16
Verify MCU EFlash	0x5A 0xA5	0x00	0x0B	0x63		Data amount to be verified (4Bytes)	CRC16
Return message	0x5A 0xA5	0x00	0x0B	0x63		CRC32 (4Bytes)	CRC16
Write SPI Flash	0x5A 0xA5	0x00	Data length +0x07	0x64	Addr	Data	CRC16
Return message	0x5A 0xA5	0x00	0x09	0x64	Addr	OK/NG	CRC16
Read SPI Flash	0x5A 0xA5	0x00	0x09	0x65	Addr	Data amount to be read (2Bytes)	CRC16
Return message	0x5A 0xA5	0x00	Data length +0x07	0x66	Addr	Data	CRC16
Verify SPI Flash	0x5A 0xA5	0x00	0x0B	0x66	Addr	Data amount to be verified (4Bytes)	CRC16
Return message	0x5A 0xA5	0x00	0x0B	0x66	Addr	CRC32 (4Bytes)	CRC16

Note:

1. OK = 0x4F 0x4B; NG = 0x4E 0x47
2. There are three types of commands, Write Data, Read Data, and Verify Data commands.
3. To write/read MCU EFlash, below rules should be followed:
   (1) Available address range is 0x00001000 ~ 0x00034000.
   (2) To write data, the target address must be divisible by 4.
   (3) For each write/read operation, the maximum data amount is 2K bytes.
4. To write/read external SPI Flash, below rules should be followed:
   (1) Spaces occupied by UartTFT-V3_Flash.bin < Accessible address < SPI Flash capacity.

(2) **To write Nand Flash**, the target address must be divisible by 128K. Before writing data, UartTFT controller will erase 128K bytes, starting from the target address. The content of the erased spaces will be 0xFF.

To write Nor Flash, the target address must be divisible by 4K. Before writing data, UartTFT controller will erase 4K bytes, starting from the target address. The content of the erased spaces will be 0xFF.

(3) For each write/read operation, the maximum data amount is 2K bytes. When the writing data amount is more than 2K bytes, UartTFT controller will send 2K byte at a time, and the last operation is usually less than 2K bytes.

5. When a great deal of data is written (e.g. several hundred Kbytes), data loss may happen.

12.2. Command Verification – CRC Verification Returned Message

When UartTFT controller receives one of the above commands sent by the Host, the communication sequence will be as follows:

1. UartTFT controller verifies the command, and returns a Pass or Fail message (CRC Verification Returned Message) to the Host
2. If the command is verified ok, then UartTFT controller will continue to execute the command and then return the execution result.

The CRC verification returned messages are listed below:

Table 12-2: CRC Verification Returned Message

Returned Message	Header (2Bytes)	Length (1Byte)	Command Code (1Byte)	Result (1Byte)	CRC16 (2Bytes)
Write MCU EFlash – Pass	0x5A 0xA5	0x04	0x61	0xFF	0x68 0x60
Write MCU EFlash – Failed	0x5A 0xA5	0x04	0x61	0x00	0x28 0x20
Read MCU EFlash – Pass	0x5A 0xA5	0x04	0x62	0xFF	0x68 0x90
Read MCU EFlash – Failed	0x5A 0xA5	0x04	0x62	0x00	0x28 0xD0
Verify MCU EFlash – Pass	0x5A 0xA5	0x04	0x63	0xFF	0x69 0x00
Verify MCU EFlash – Failed	0x5A 0xA5	0x04	0x63	0x00	0x29 0x40
Write SPI Flash – Pass	0x5A 0xA5	0x04	0x64	0xFF	0x6B 0x30
Write SPI Flash – Failed	0x5A 0xA5	0x04	0x64	0x00	0x2B 0x70
Read SPI Flash – Pass	0x5A 0xA5	0x04	0x65	0xFF	0x6A 0xA0
Read SPI Flash – Failed	0x5A 0xA5	0x04	0x66	0x00	0x2A 0xE0
Verify SPI Flash – Pass	0x5A 0xA5	0x04	0x66	0xFF	0x6A 0x50
Verify SPI Flash – Failed	0x5A 0xA5	0x04	0x66	0x00	0x2A 0x10

12.3. CRC Calculation & Code

12.3.1. CRC16

The Command Code, Address, and Data content are used for calculating CRC 16, as shown below:

Table 12-3: Command Table

Command	Header 0x5A 0xA5 (2 Bytes)	Fixed 0x00 (1 Byte)	Length (2 Byte)	Command Code (1 Byte)	Address (4 Bytes)	Content (n Bytes, Max. 2048)	CRC 0xXX 0xXX (2 Bytes)

12.3.2. CRC32

Reference code is as shown below: GetCrc32() returns CRC 32

```
uint32_t CRC32_Table[256];

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

    Crc=value;
    pch = InStr;

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

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

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

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



```
            CRC32_Table[i] = c;
        }
}
```

Note: 'InStr' is 8bits unsigned array, which will be used to store the data that needs to be verified.
' len' is the data length. For large amount of data, split the data into smaller packages and verify them respectively.
'value' represents the last calculated CRC 32 value

12.4. Flash_RW-II Tool

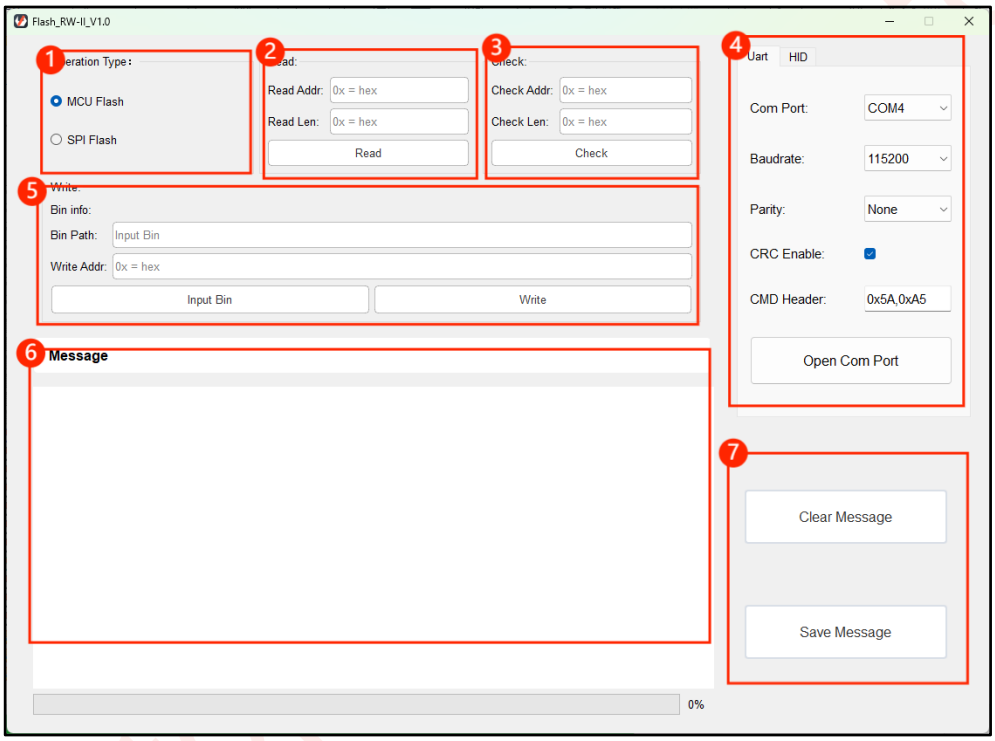


Figure 12-1: Flash_RW-II

❶ Write/Read Target: check MCU Flash or SPI Flash to select the target devices.

❷ Setup for Read operation:
Read Addr: Set the starting address of the read operation
Read Len: Set the length of the read operation, maximum 2Kbytes.
The enter values can be hexadecimal or decimal, as the figure shown below:

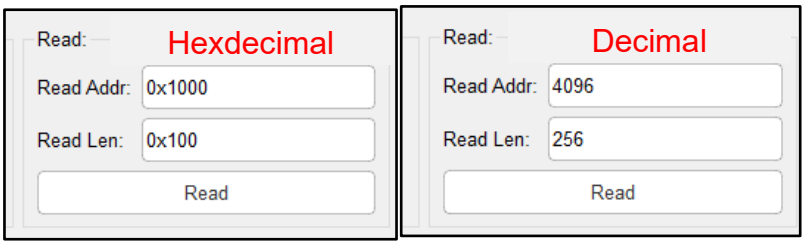


Figure 12-2: Set Address

Read: Click the [Read] button, and the read data will be shown in the Message area ❻.

❸ Setup for verification operation:

Check Addr: Set the starting address of the verification operation.

Check Len: Set the data length of the verification operation.

Check: Click the [Check] button to start verification, and the returned CRC 32 value will be shown in Message area ❻.

❹ Setup for Communication Port:

Open Com Port: Click to open the selected communication port.

❺ Setup for Write operation:

Write Addr: Set the starting address of the write operation.

Input bin: Click to add a bin document to be written.

Write: Click the [Write] button to start write operation.

❻ Message area:

Display the returned messages.

❼ Other function:

Clear Message: Click to clear the Message area

Save Message: Click to save the listed message to a TXT file.

12.5. Read/Write Flash through the Uart port

1. In the MCU Code, modify the macro definition, UARTBUS_OPTION to 0, as the figure shown below. Program the new MCU Code to LT7589.

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

Figure 12-3: Select the Uart communication mode

2. Connect MCU_TX1 to FX-RXD, and MCU_RX1 to FT-TXD, and then using CON1 port to connect to the computer. Refer to the below figure:

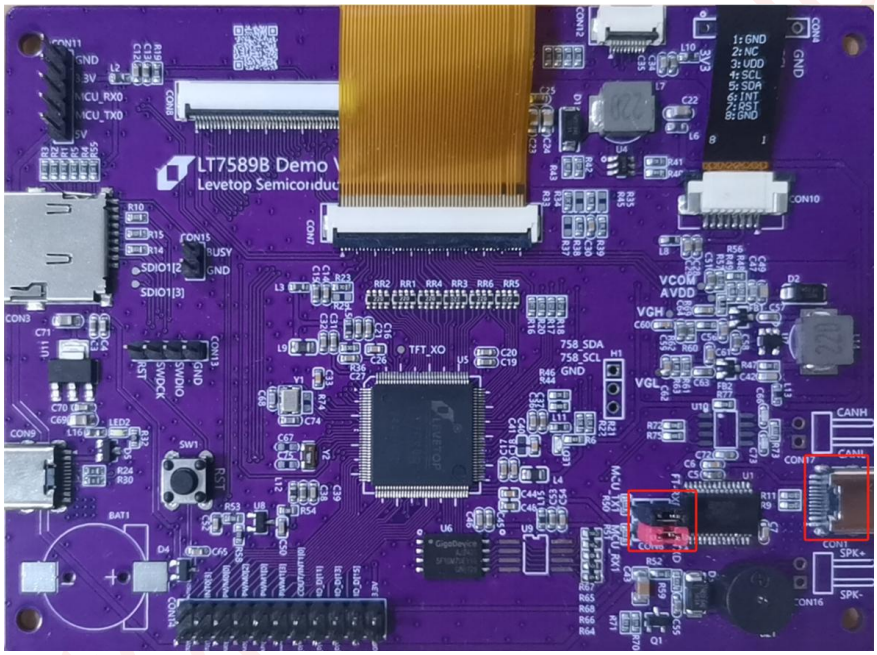


Figure 12-4: Hardware Configuration for Uart Mode

3. Activate Flash_RW-II tool, click to choose the Uart page, as the figure shown below. Select the target COM port, Baudrate, and CMD Header etc. Finally click on [Open COM Port] to start the connection.

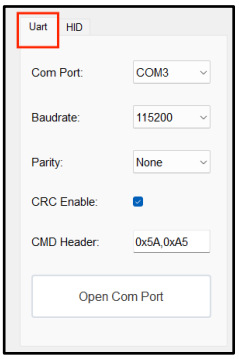


Figure 12-5: Select Uart Mode

12.6. Read/Write Flash through Vcom (USB cable)

1. In the MCU Code, modify the macro definition, UARTBUS_OPTION to 4, as the figure shown below. Program the new MCU Code to LT7589.

```
module_select.h*
13
14  #define IIC_BUS              0        // Communication,
15  #define UARTBUS_OPTION       4        // Communication,
16  #define DualFlash            0
17  #define encoder_on           0        // 1:Open 0:Close
18  #define Touch_selection      0        // 0:CTP  1:RTP
```

Figure 12-6: Select the Vcom communication mode

2. Weld 22 ohm resistors at the location of R24 and R330, and then using the CON9 port to connect to the computer. Refer to the below figure:

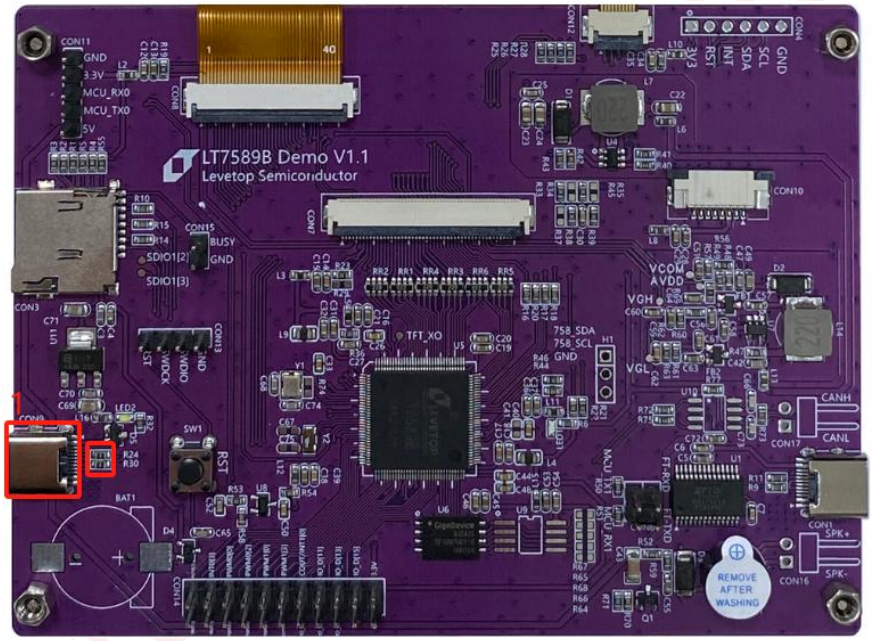


Figure 12-7: Hardware Configuration for Vcom Mode

3. Activate Flash_RW-II. Click on Uart page and select the connected COM port. Finally, click on [Open Com Port]. Refer to below figure:

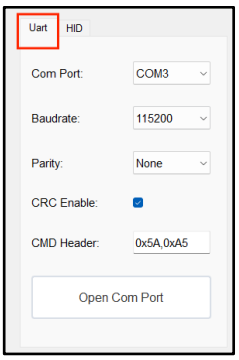


Figure 12-8: Select Vcom Mode

12.7. Read/Write Flash through HID

1. In the MCU Code, modify the macro definition, UARTBUS_OPTION to 3, as the figure shown below. Program the new MCU Code to LT7589.

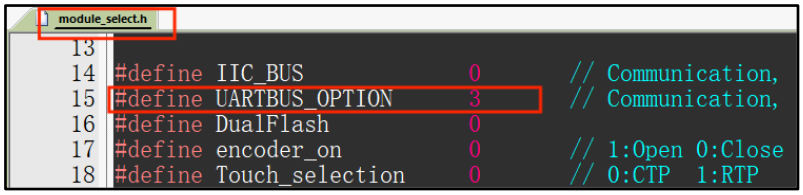


Figure 12-9: Select the HID communication mode

2. Weld 22 ohm resistors at the location of R24 and R330, and then using the CON9 port to connect to the computer. Refer to the below figure:

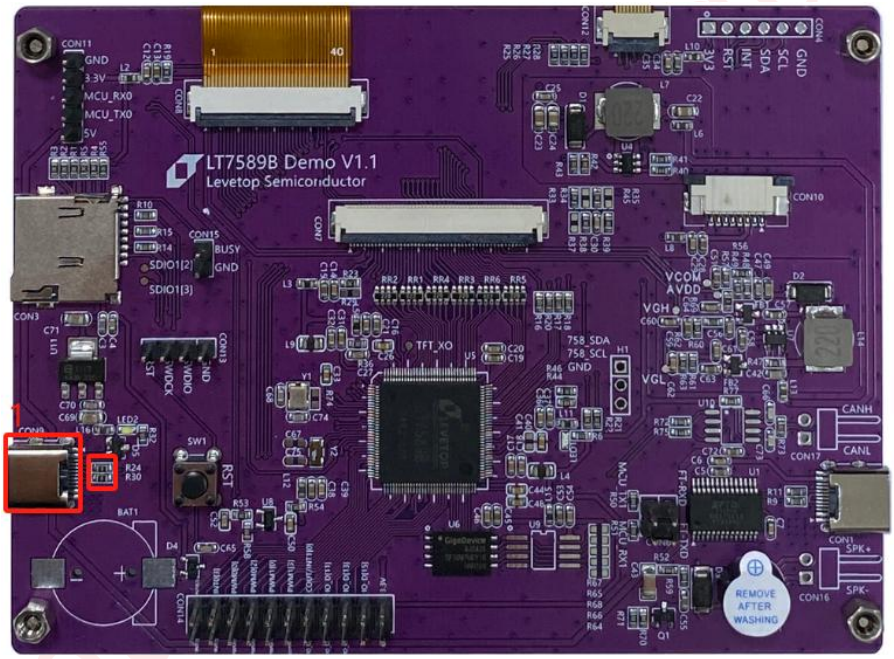


Figure 12-10: Hardware Configuration for HID Mode

3. Activate Flash_RW-II. Click on HID page, and then click on [OpenHID Device]. Refer to below figure:

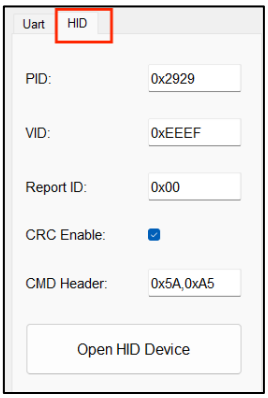


Figure 12-11: Select HID Mode

12.8. Update UartTFT-V3_Flash.bin by Flash_RW-II

Update procedure is as shown below:

1. Send the below Uart command to the UartTFT controller so that it enters receiving mode:

5A A5 07 10 7042 0001 9E D7

2. Refer to the Figure 12-9, activate the software, Flash_RW-II, and then click on [Open Com Port], as noted ❶ to connect with the UartTFT controller.

Figure 12-12: Update UartTFT-V3_Flash.bin by Flash_RW-II

3. Click on [Input Bin], as noted ❷ to assign the UartTFT-II_Flash.bin file.
4. Input 0 to [Write Addr], as noted ❸ and make sure that [SPI Flash] is chosen.
5. Click on [Write], as noted ❹ to start the update.
6. After the update is done, send the below Uart command to reset the UartTFT controller back to working mode.

5A A5 07 10 7042 0000 5F 17

13. Switch Display Direction – Landscape & Portrait

This section is to describe how users may implement an application that can switch the display direction of the same UI contents. Note that the function only supports switching the display direction at Landscape (0 degree) and Portrait (90 degrees)

To implement this function, two UI projects must be prepared in advance. One for Landscape display, and another for the Portrait display.

13.1. Requirement for UI projects

13.1.1. Page sequence and Widget address

The two UI projects must follow below rules:
1. The page sequence of the two UI projects must be set to the same, as shown in Figure 13-1.
2. The address of each widget for the two UI projects must be set to the same.

Page	Name
0000	Portrait_Home
0001	Portrait_Page1

Page	Name
0000	Landscape_Home
0001	Landscape_Page1

Figure 13-1: Setting the same page sequence for the two UI projects

As shown in Figure 13-2 and 13-3, the widget address of the Portrait UI project is set the same as that of the Landscape UI project.

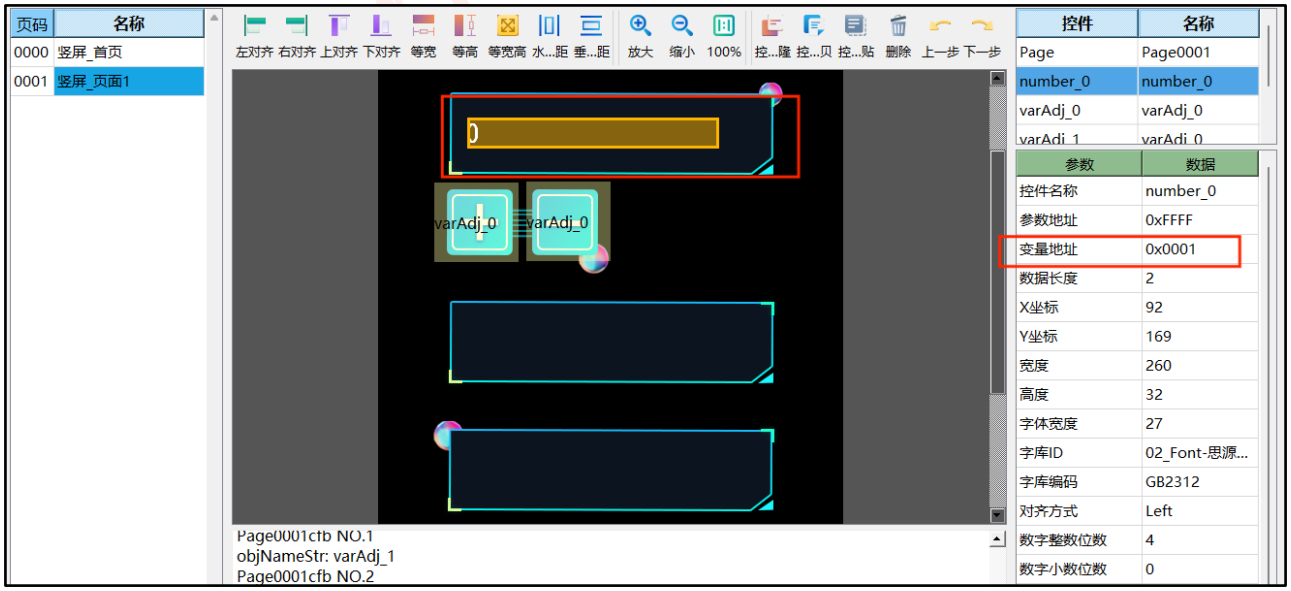


Figure 13-2: Portrait UI Project

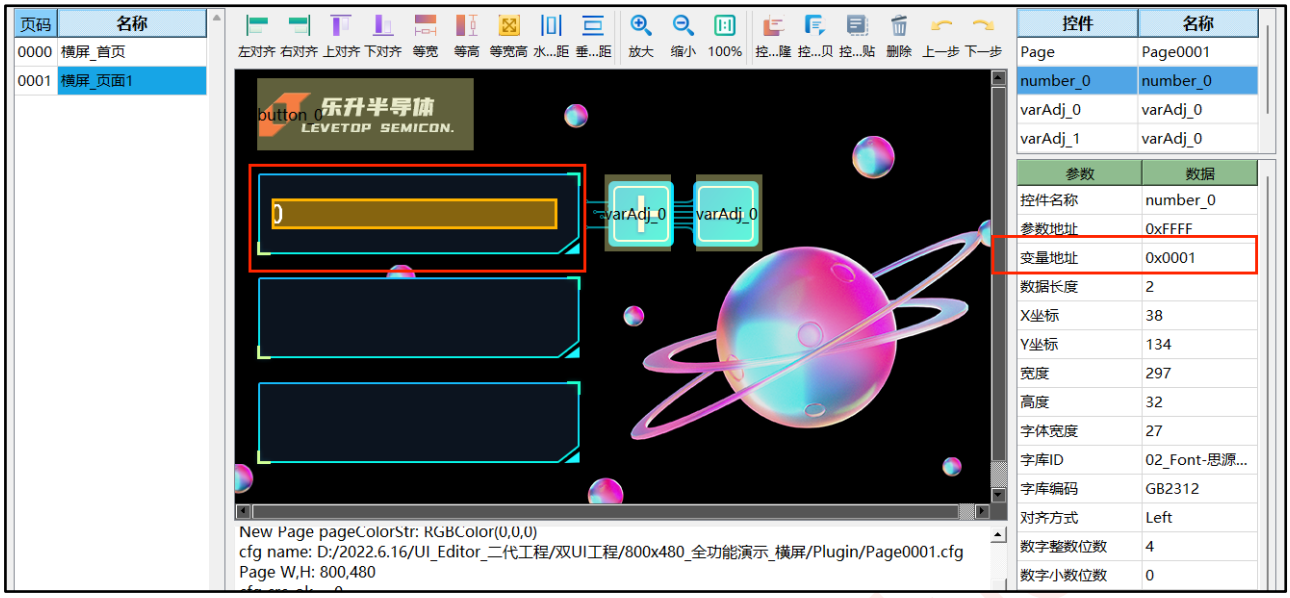


Figure 13-3：Landscape UI Project

Note that all the widget parameters of the two UI project should be set the same, except for the X and Y coordinates.

13.2. Combine two UI Projects

Both Portrait and Landscape UI projects have to be programmed to the same SPI Flash. Before the programming, the two UI projects have to be combined into one project.

Follow the below procedure to combine two UI projects:

1. Download the tool, UI_BinFiles_Merge.exe from Levetop official website, or contact Levetop for it.

2. Activate the tool, as shown below:

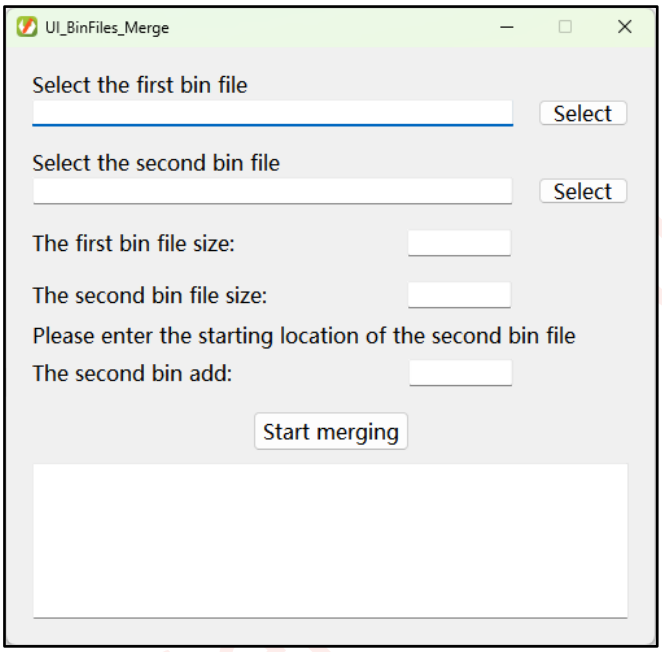


Figure 13-4: UI_BinFiles_Merge

3. As shown in Figure 13-5, click [Select] to add the first UartTFT-II_Flash.bin, this one must be the Landscape UI project.

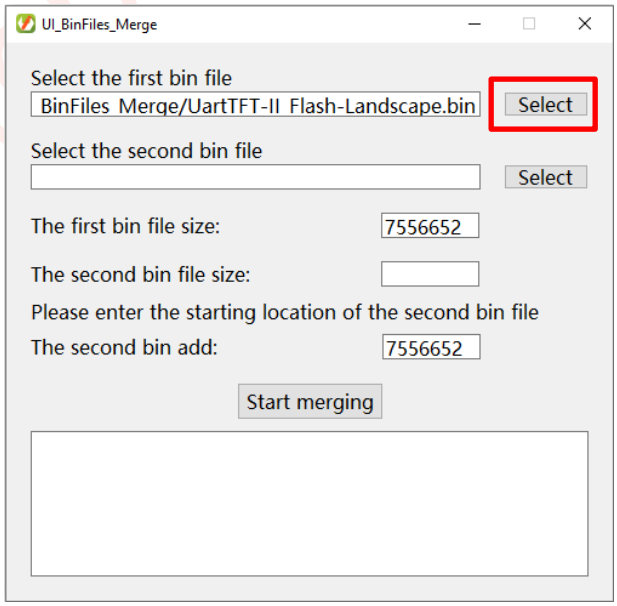


Figure 13-5: Add the Landscape UI project

4. As shown in Figure 13-6, click [Select] to add the second UartTFT-II_Flash.bin, this one must be the Portrait UI project.

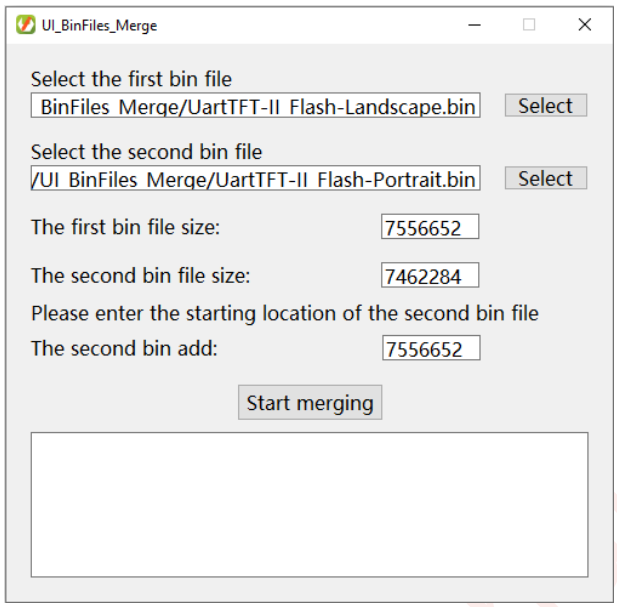


Figure 13-6: Add the Portrait UI project

5. Click [Start merging] to combine the two projects:

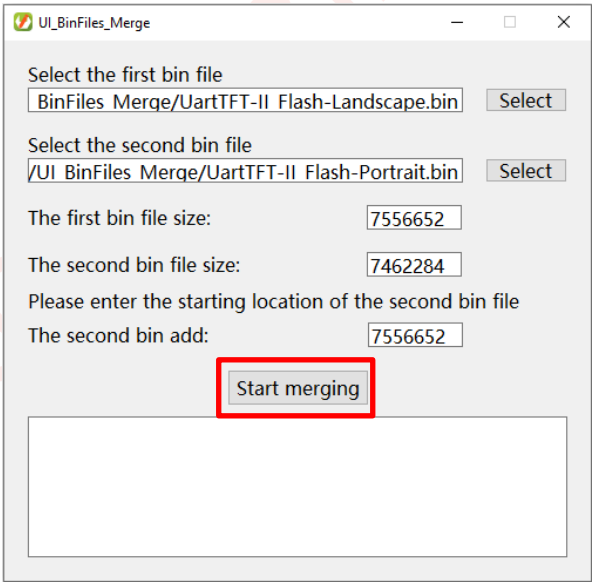


Figure 13-7: Start combining the selected projects

6. Retreive the start address of the second UI file. As shown in Figure 13-8, 7556652 is in decimal, and 0x00734E2C is in hexadecimal.

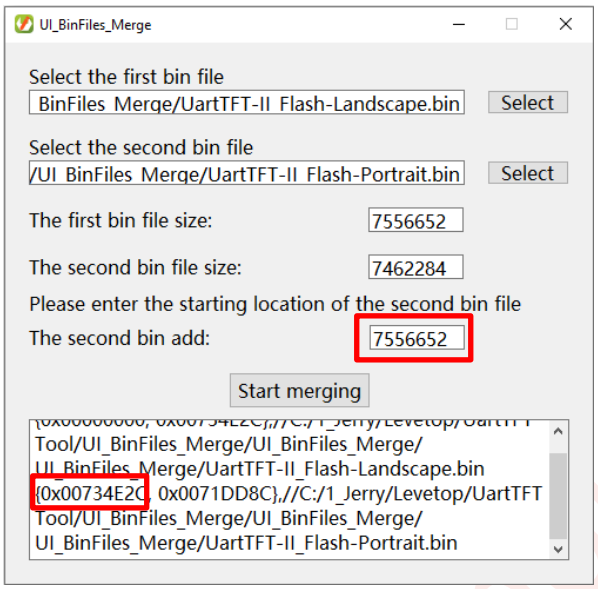


Figure 13-8: Combination Done

After the combination is done, two new files will be generated as shown in Figure 13-9, where **UartTFT-V3_Flash.bin** is the combined bin file, and **UartTFT-V3_Flash-Addr.txt** lists the start address and file size of both UI projects, as shown in Figure 13-10.

Figure 13-9: Generated Files

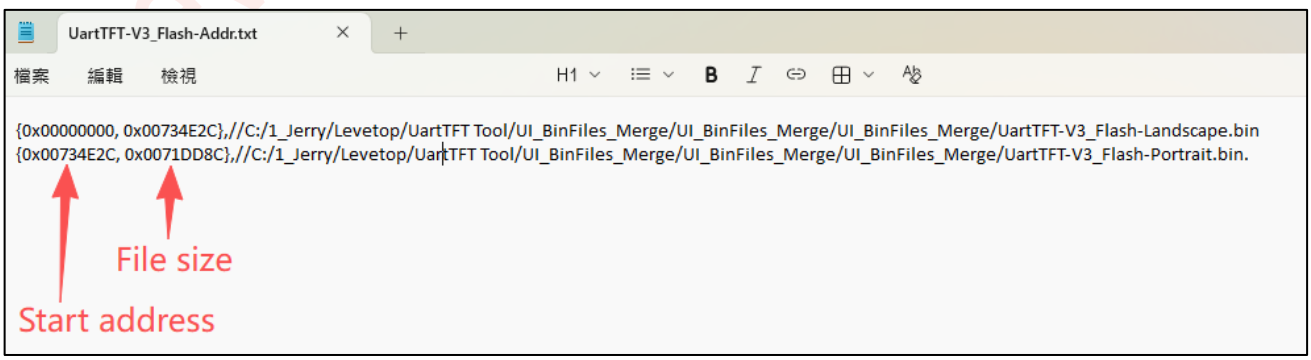


Figure 13-10: UartTFT-V3_Flash-Addr.txt

7. Modify the MCU code to assign the start address of the second UI, which is 0x00734E2C for the case here.

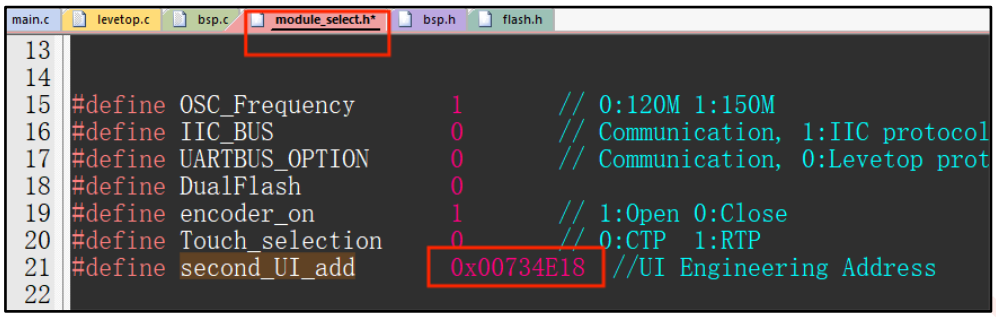


Figure 13-11: Assign the start address

8. Program the generated UartTFT-V3_Flash.bin and the new MCU code to the UartTFT panel.

13.3. Send Uart Commands to Switch Display Direction

After the bin file and MCU code is programmed as described in section 13.2, UartTFT controller will display Landscape UI by default once power on. Users may switch to Portrait UI by assigning 0x0001 to the register 0x7040, as shown below. Assigning 0x0000 to the register 0x7040 can then switch back to Landscape UI.

Landscape to Portrait: 5A A5 07 10 7040 0001 3F 17

Portrait to Landscape: 5A A5 07 10 7040 0000 FE D7

14. Widgets Overlap

To apply this function, the below requirements / conditions must be satisfied:

1. The widgets that can be overlapped include:

Foreground widgets	String_Label, Text Number Display, Graphics Number Display, Icon, Bit Status, and Digital Clock.
Background widgets	Icon, Bit Status, Gif, Button, Circular Touch, Slider Bar, Variable Button, Combo Button, and Extend Button

2. Each widget size must follow the setting limits. Refer to *Setting Limits*
3. The size of the background widget must be larger than that of the foreground widgets. A foreground widget must be **fully** placed inside the background widget, as depicted in Figure 14-1.
4. Needle widgets cannot be overlapped.
5. Maximum 2 layers overlap is supported.
6. Inside a background widget, the number of foreground widgets is not limited. However, the foreground widgets may not overlap with each other.

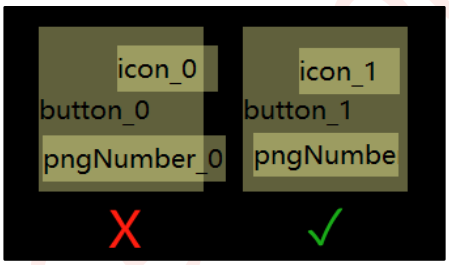


Figure 14-1: Widgets Overlap

15. Additional Information

15.1. Codes & Documents

Followings are the codes and documents related to a UI_Editor-III project:

bootloader: This is the code that enables UartTFT controller to download MCU_Code.bin and UartTFT-V3_Flash.bin.

MCU_Code.bin: This is the code that enables UartTFT controller to implement the display functions and operations edited on UI_Editor-III. Developers may add codes to customize their own functions. MCU_Code.bin is programmed to UartTFT controller internal Flash. Its size is usually less than 256KB.

UartTFT-V3_Flash.bin: This file is generated by UI_Editor-III after compilation. It includes all the required materials and settings that developers design on UI_Editor-III. UartTFT-V3_Flash.bin is programmed to external SPI Flash. Its size varies according to the imported materials.

15.2. Using an Existed Project to Create a New Project

Developers may create a new project with existing material used by other projects. Simply follow the below steps:

(1) Copy all the folders of the existing project, and paste them to another folder.

(2) Delete all the existed files in [Plugin] folder, as shown in Figure 15-1

(3) Create a new project with the copied material

Figure 15-1: Delete the Files in Plugin Folder

15.3. Screen Rotation

To use an 800x480 LCD as a 480X800 display, developers may simply setup the [Rotate] parameter in [Project Setting] page, and design their UI project accordingly. The panel resolution settings should still follow its original configuration (no need to modify). As shown in Figure 15-2.

Note: No need to modify the panel resolution settings.

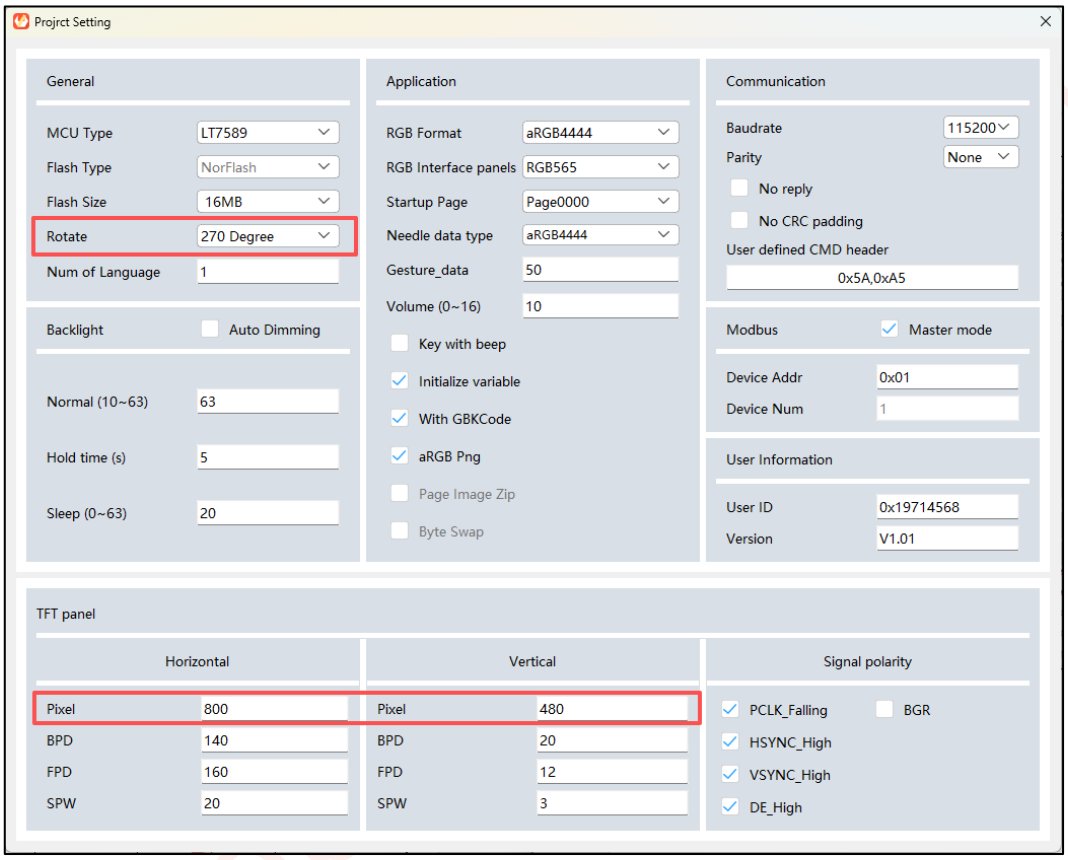


Figure 15-2: Set [Rotate] Parameter

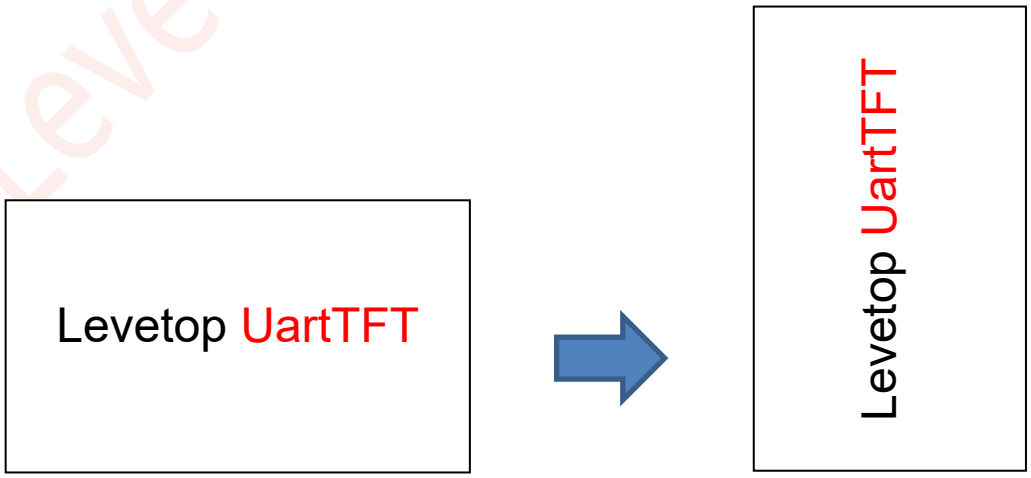


Figure 15-3: Rotate 270° Clockwise

15.4. UartTFT-V3_Flash.bin

A UartTFT-V3_Flash.bin contains font, Gif, pictures, Wav, and page information. Since the size of a UartTFT-V3_Flash.bin varies according to the materials used, developers should make sure if the SPI Flash has enough room for storing the UartTFT-V3_Flash.bin

Among the materials used in UI_Editor-III, pictures and Gifs consume storing spaces the most:

Picture: The data size of an 800x480 picture can be calculated as below:

RGB565 → 800*480*2/1024 = 750KB;

RGB888 → 800*480*3/1024 = 1125KB;

Gif: Gif is converted frame by frame in UI_Editor-III. Each frame is taken as a picture. Therefore, a Gif with high frame count will consume a great number of spaces. As shown in Figure 15-4, the size of the converted bin file is over 8 times bigger than the original one.

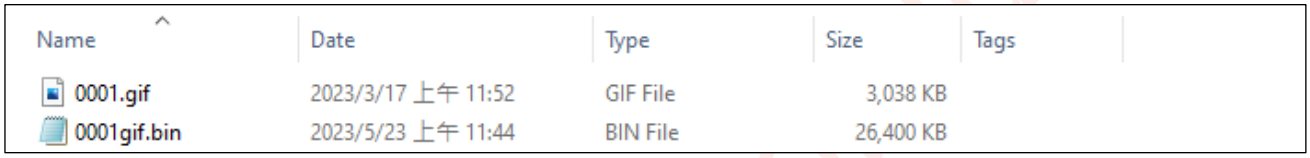


Figure 15-4: Gif converted to bin

15.5. Data Type

Table 15-1: Data Type List

Type	Address	Length	Max. Value	Range
long long	writeAddr	8bytes	0x7FFF	-2^63 ~ 2^63-1
	writeAddr + 0x0001		0xFFFF	
	writeAddr + 0x0002		0xFFFF	
	writeAddr + 0x0003		0xFFFF	
int	writeAddr	4bytes	0x7FFF	-2^31 ~ 2^31-1
	writeAddr + 0x0001		0xFFFF	
uint	writeAddr	4bytes	0xFFFF	0 ~ 2^32-1
	writeAddr + 0x0001		0xFFFF	
short	writeAddr	2bytes	0x7FFF	-2^15 ~ 2^15-1
ushort	writeAddr	2bytes	0xFFFF	0 ~ 2^16-1
char_H8	writeAddr-H	1byte	0x7F	-2^7 ~ 2^7-1
char	writeAddr-L	1byte	0x7F	-2^7 ~ 2^7-1
uchar_H8	writeAddr-H	1byte	0xFF	0 ~ 2^8-1
uchar	writeAddr-L	1byte	0xFF	0 ~ 2^8-1

Note:

writeAddr-H: the higher byte of the data stored in writeAddr;
writeAddr-L: the lower byte of the data stored in writeAddr

15.6. Digit Number of Integer & Decimal

When implementing Text Number Display and Graphics Number Display, the sum of the digit numbers of the integer and decimal should be less than the digit number of the data type.

Short&ushort: 5 digits，int&uint: 10 digits，long long: 19 digits

Also, when setting the “defaultNumber” parameter, the digit number of the integer and decimal must not exceed the preset digit value. In addition, the input number that is composed of integer and decimal digits, must be within the range of the preset data type. For example, if the integer digit is set to 3, the decimal digit is set to 2, and the data type is “short” (maximum number: 32767), then the maximum value allowed is 327.67. The above rule applies to “int” and “long long” as well.

15.7. Icon Width & Height

The width and height of all the icons in the same group (e.g. number icons) must be the same with each other. However, for the icons used in Graphics Number Display 【0、1、2、3、4、、5、6、7、8、9 、. 、 -】, and the icons used in Digital Clock 【 0、1、2、3、4、5、6、7、8、9、：、/、/、/ 】 & 【Sun、Mon、Tues、Wed、Thu、Fri、Sat】, the width of the icons in different categories (e.g. number vs. decimal point, and number vs. week day) can be set differently.

Some of the widgets provides unpressedIcon and pressedIcon parameters. The width and height of these two icons must be the same.

15.8. Font Library

When a String_Label or Text Scroll widget is set to be updated by CN_KeyBoard, the widgets (String_Label & Text Scroll) must apply GBK fonts to avoid abnormal display.

15.9. Delete Selected Image

To delete a selected image of a widget or a page, follow below procedure:

(1) Locate the image item in the Parameter Setting Window, as show in Figure 15-5;

Parameter	Data
name	varAdj_3
X	44
Y	121
W	82
H	108
writeAddr	0x3602
adjStep	30
minValue	0
maxValue	600
dataType	short
gradation	+
cyclicalCounting	Stop
longPress	Once
unpressedIcon	0013.png
pressedIcon	
reportToHost	Disable

Figure 15-5：Locate the Image Item

(2) Double click on the image item, and a file manager window will pop up, as shown in Figure 15-6;

(3) Click on [Cancel] to close the window;

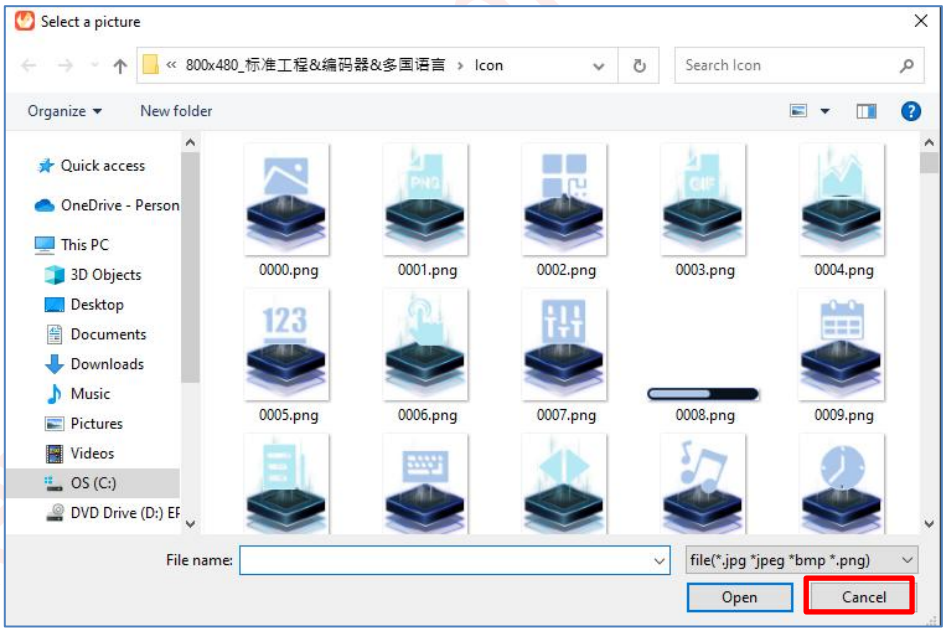
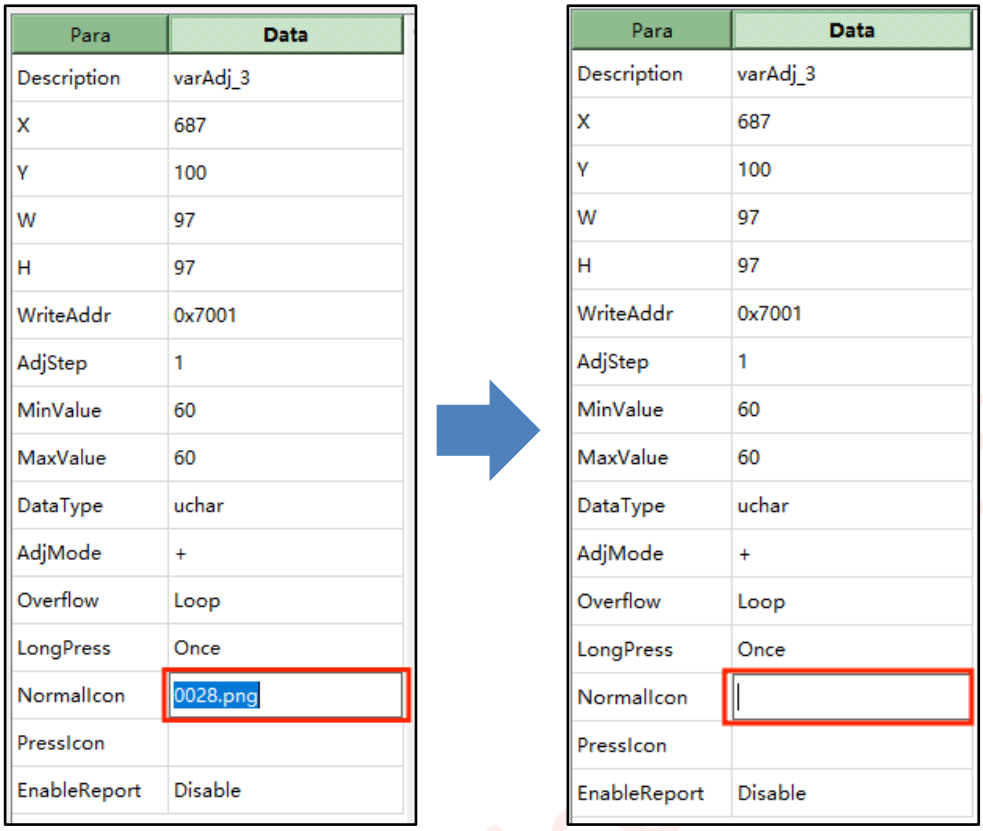


Figure 15-6：File Manager Window

(4) Delete the image name in the Parameter Setting Window, as shown in Figure 15-7, and then click on [Enter] to confirm the operation. The final result is as show in Figure 15-8



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

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

Figure 15-7: Delete the Selected Image

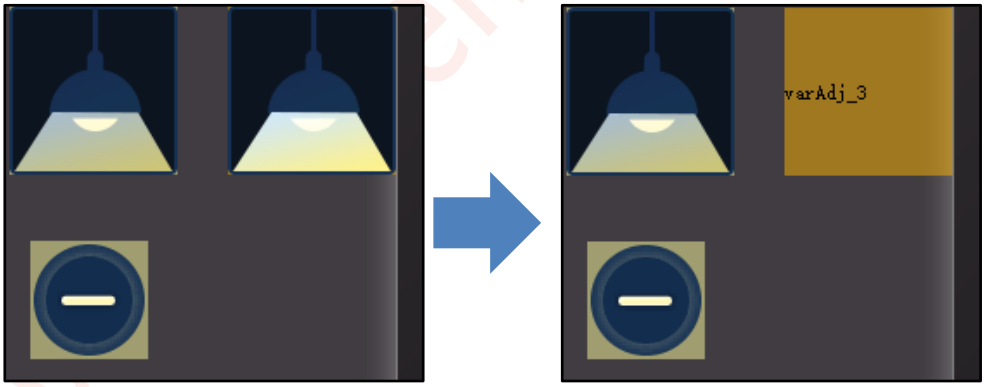


Figure 15-8: Operation Result

15.10. Widget Initial Setting

When multiple widgets share the same variable address, their initial settings should be the same as well. For example, the parameter, defaultNumber, should be the same for all the Text Number Display widgets with the same variable address.

15.11. Data Length and Address Allocation

For the widgets including CN_KeyBoard, En_KeyBoard, String_Label, Text Scroll, and QRCode, their address allocation must follow the rule expressed below.

As an example shown in Table 15-2, a widget with the starting address of 0x2000, has 3 data, that is, Data Length = 3, therefore, the data of this widget will be stored in 0x2000 ~ 0x2002. In addition, an ending code, 0x0000, will be added to the end of the data, and stored to the subsequent address, which is 0x2003 in the case here. The starting address of the next widget can therefore be concluded as below:

Starting address of the next widget >= Starting address of the current widget + Data Length + 1

Table 15-2: Data Length and Address Allocation

	Address Index	Data Length	Content
Starting Address	0x2000	3	Data
	0x2001		
	0x2002		
Ending Address	0x2003	1	0x0000
Next Starting Address	0x2004	4	Data
	0x2005		
	0x2006		
	0x2007		
Ending Address	0x2008	1	0x0000

15.12. Widget Overlap

To avoid false operations, widgets with touch functions cannot be overlapped with each other.

15.13. Widget Size

When adding a picture to a widget, the widget size will be adjusted according to the picture size automatically. For the widgets with no pictures attached, their size (width & height) should be set within the panel area, that is,

Widget left-top coordinate X (Y) + Widget Width (Height) <= Panel Width (Height)

15.14. Display Scaling

Due to various computer resolutions, UI_Editor-III may not be displayed properly for certain cases, as shown in Figure 15-9. Developers may improve it by adjusting the display scaling, as described below. (Only available in Win10)

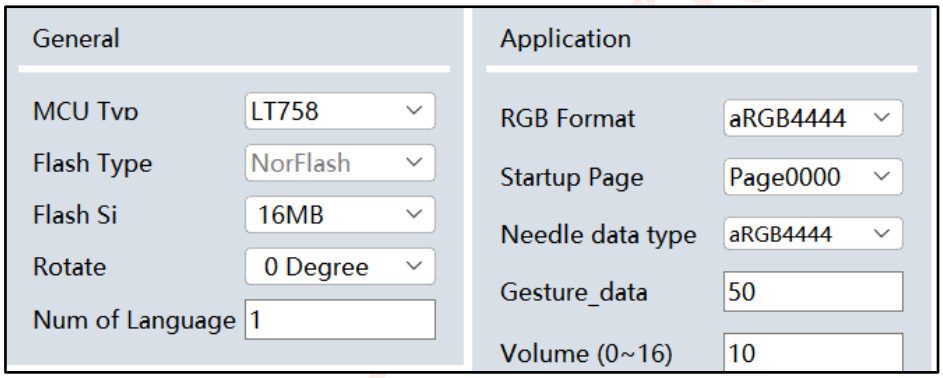


Figure 15-9: Incomplete Display

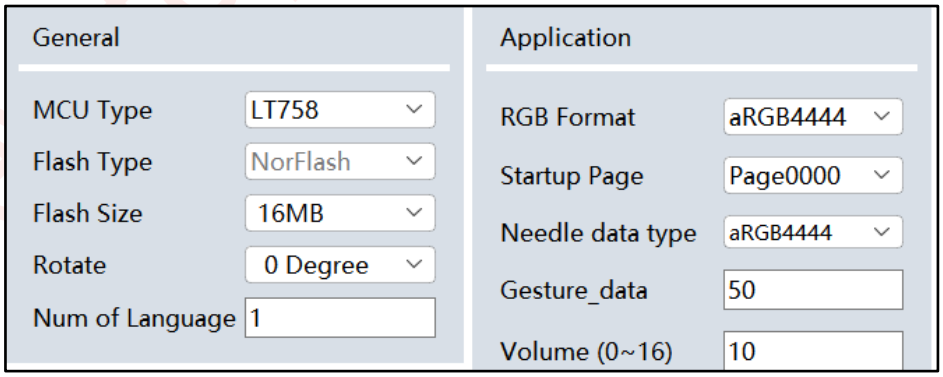


Figure 15-10: Normal Display

Step 1: Close UI_Editor-III, and then right click on the EXE file. Select [Properties] from the pop-up window.

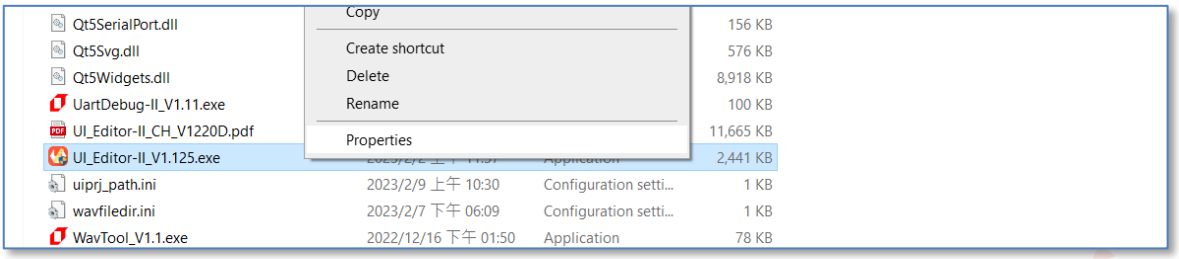


Figure 15-11: Open [Properties] Window

Step 2: Click on [Compatibility] page, and then click on [Change high DPI settings]

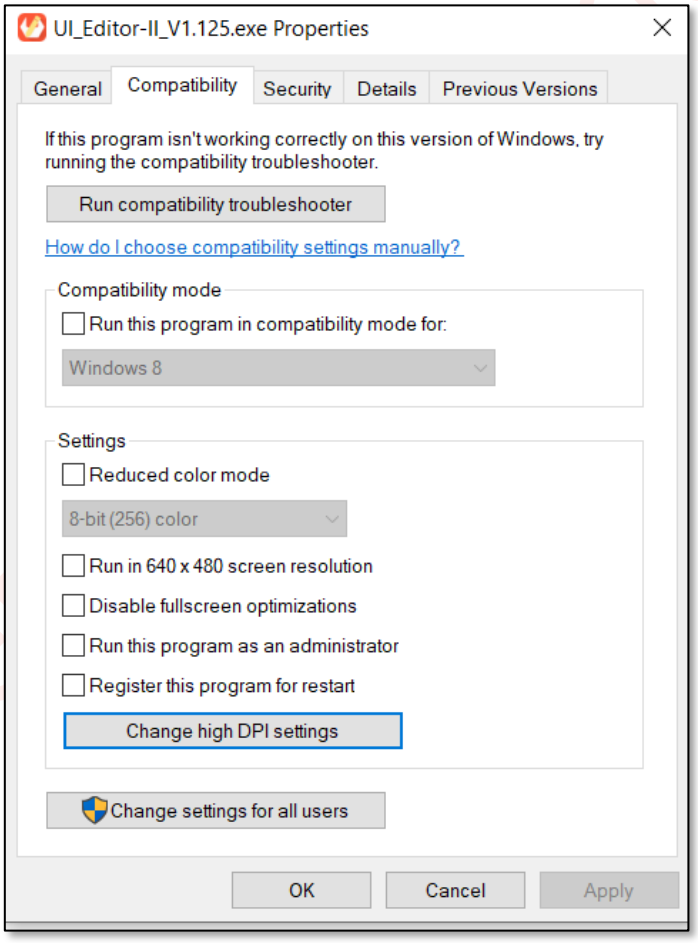


Figure 15-12: Change DPI Setting (1)

Step 3: Check [Override high DPI scaling behavior], and then select [System (Enhanced)]. Next, click [OK] to confirm the operation.

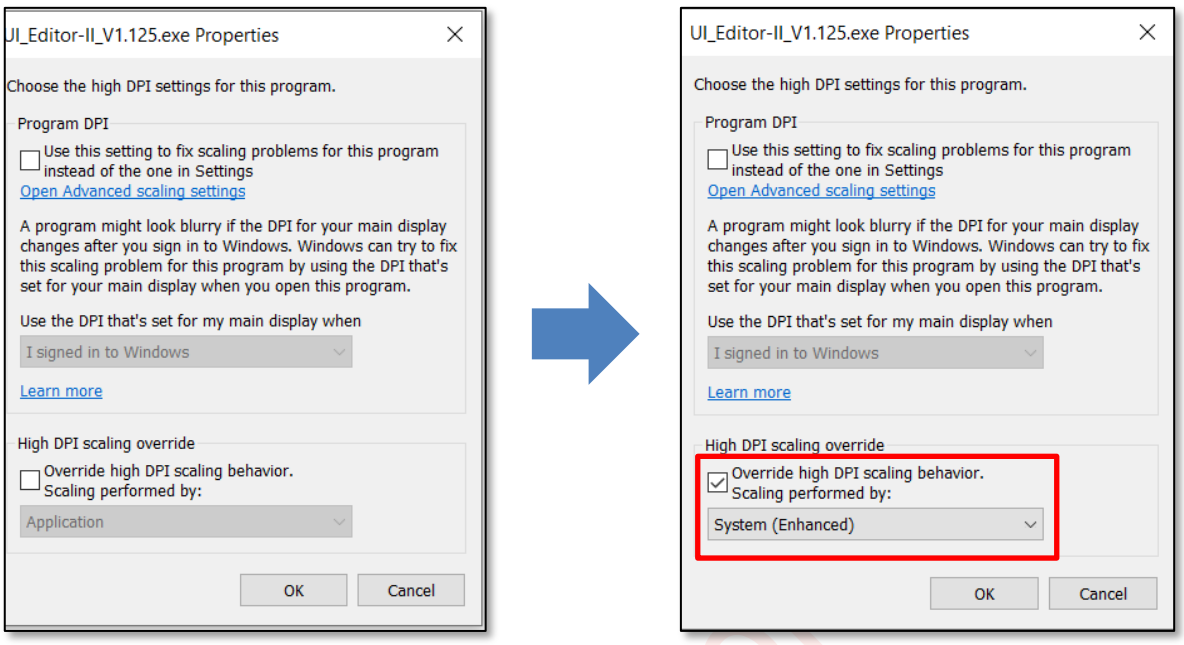


Figure 15-13: Change DPI Setting (2)

Step 4: Click on the [OK] button in the [Compatibility] page to finish the setting.

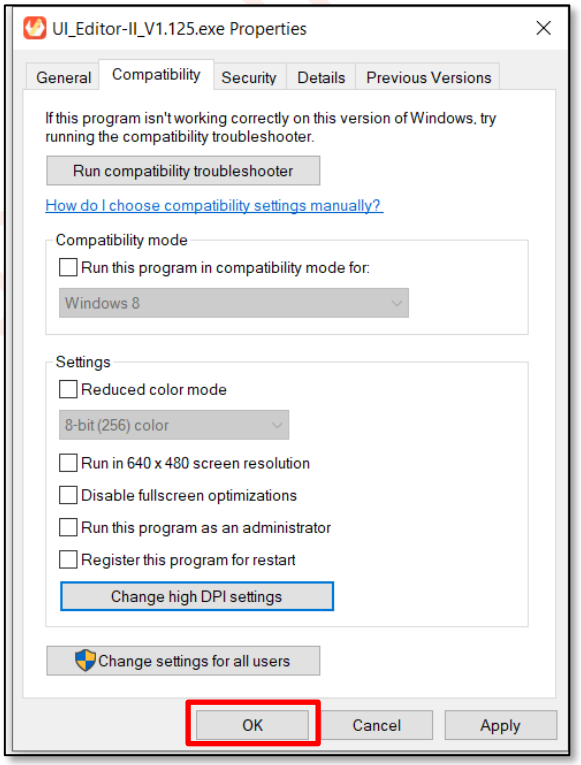


Figure 15-14: Confirm the change

15.15. Computer OS

Preferred OS: Win10 or above. It is suggested that developers operate UI_Editor-III in Full Screen mode.

15.16. Naming Rule

The names of material, widgets, pages, and projects should not include special characters as shown in Table 15-3. There is only one decimal point "." allowed before the file suffix.

Table 15-3: Illegal Symbol List

Mode	EN	EN	EN	CN/EN	CN/EN	EN	EN	CN/EN	CN	CN
Symbol	\	/	:	*	?	<	>	|	。	，

15.17. Material Library

Levetop provides a public Material Library which contains various icons, and pictures etc. Developers may contact Levetop service team for it.

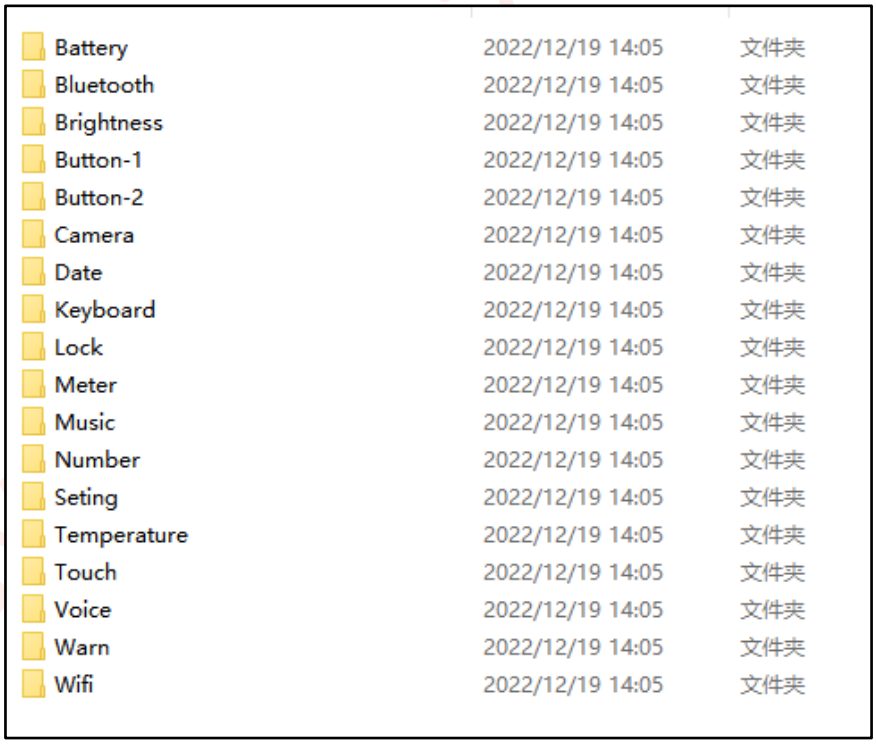


Figure 15-15: Material Library

15.18. dataFormat

15.18.1. Structure of Various dataFormat

1、dataFormat supported by LT7589:

The dataFormat described below is based on a single Pixel.

αRGB8888: Each pixel is represented by 32bits data, as the structure shown in Table 15-4:

Table 15-4: αRGB8888 Data Structure

Data Format	Transparency α	Red	Green	Blue
αRGB8888	Bit 24~32	bit 23~16	bit 15~8	bit 7~0
	α7~α0	R7~R0	G7~G0	B7~B0

αRGB4444: Each pixel is represented by 16bits data, as the structure shown in Table 15-5:

Table 15-5: αRGB4444 Data Structure

Data Format	Transparency α	Red	Green	Blue
αRGB4444	bit 15~12	bit 11~8	bit 7~4	bit 3~0
	α3~α0	R7~R4	G7~G4	B7~B4

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

jpg: The JPG data format is the final format to which BMP and JPG images are converted for display. Note that PNG pictures cannot be set to the jpg format.

15.18.2. dataFormat – Icon and Gif

When generating the UartTFT-V3_Flash.bin, UI_Editor-III will convert the imported pictures based on the dataFormat settings.

When dataFormat is not set → BMP and JPG pictures will be converted to JPG based on the **Project Setting** (RGB Format), and PNG pictures will be converted to αRGB4444 format.

When dataFormat is to be set → Follow the rules listed below:

1、PNG picture cannot be set to JPG

2、BMP or JPG pictures cannot be set to αRGB8888 or αRGB4444.

If a picture has to be used in more than one Icon or Gif widgets, developers must make copies of the picture, and assign different numbers to the copies.

16. Appendix

16.1. Appendix 1 - Programming

16.1.1. Upgrade UartTFT panel

A new UartTFT controller needs to be programmed before put to work. It needs three bin documents, “Bootloader.bin”, "MCU_Code.bin" and "UartTFT-V3_Flash.bin".

MCU_code.bin, just as its name implies, is the MCU code for the Uart TFT panel, and should be programmed to the flash of the chip. UartTFT-V3_Flash.bin contains the pictures, animations, texts, and display flows, and it should be programmed to the external flash chip.

Upgrading methods differ by the model of the UartTFT controller, please refer to the below table.

Table 16-1：LT7589 Upgrading Methods

Model	TFT Interface	Upgrade Bin	Uart Interface USB Interface (LT_Uart_GUI)	SD Card	USB Disk
LT7589	RGB	MCU_Code	V	V	-
		UartTFT-V3_Flash	V	V	-

Note:

‘-‘ : Not available.

‘V’: Available

16.1.2. LT7589 Upgrade

16.1.2.1. Download Bootloader

The LT7589 bootloader, LT7589_USB_SD_Uart_Bootloader, supports upgrade functions through USB, SD card, and Uart port.

The bootloader can be downloaded through the SWD port (SWDIO, SWDCLK, GND). Refer to the programming manual, LT_SWD_ISP_PRG-Kit_Vxx.pdf, for more detail.

16.1.2.2. Download bin files through SD card

Users may also program MCU_Code.bin and UartTFT-V3_Flash.bin to an LT7589 demo board through an SD card.

Please refer to the below procedure:

(1) Format an SD card / USB Disk by USB2.0 format / 2~32GB / FAT32

(2) Make two directories and name them as [MCU_Code], and [UartTFT_Flash] respectively.

(3) Save the bin files that will be programmed to the corresponding directories, as shown below:

(The name of the files and directories MUST be exactly the same as shown in Figure 16-1)

Figure 16-1: Make two directories

(4) Make sure that LT7589 demo board is power off, and then insert the SD card.

(5) Power on the LT7589 demo board. The demo board will detect the SD card automatically.

Note: There is no need to repeatedly upgrade same MCU_Code.bin file. Developers may leave the MCU_Code directory blank to skip the programming of MCU_Code.bin. The same rule applies to the upgrade of UartTFT-V3_Flash.bin.

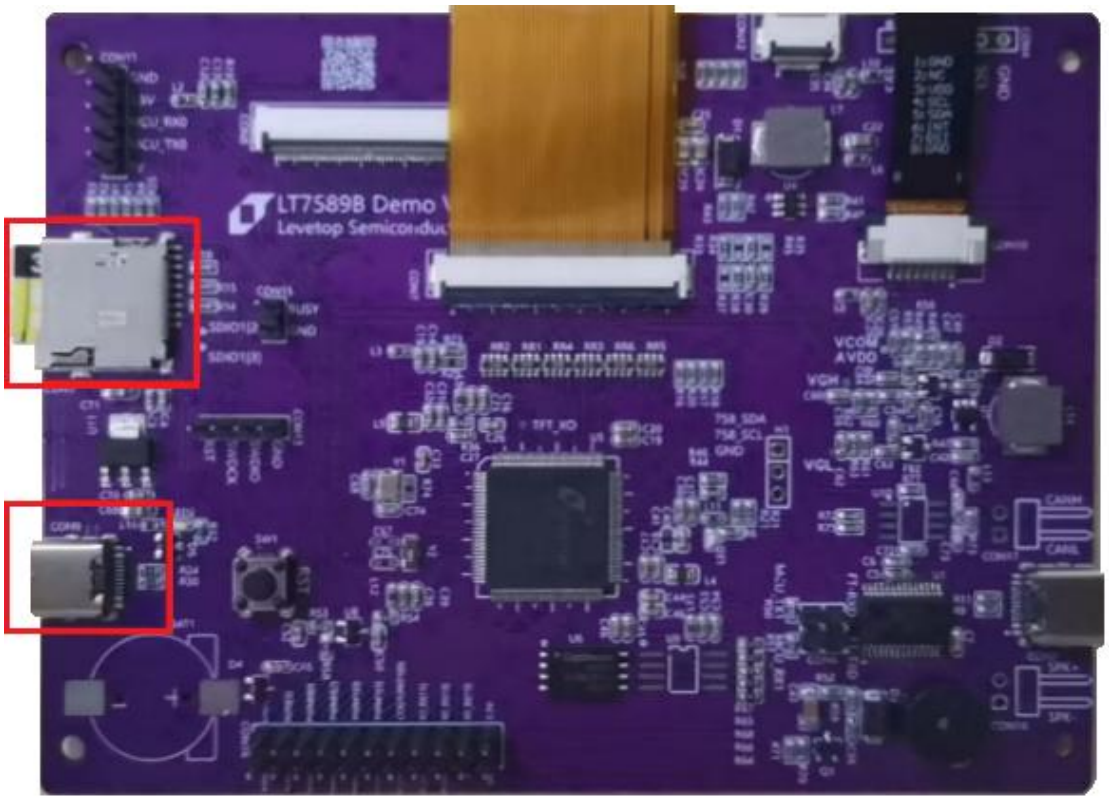


Figure 16-2：SD Card Upgrade

The LCD panel will show as Figure 16-4 when programming a UartTFT-V3_Flash.bin to an LT7589 demo board.

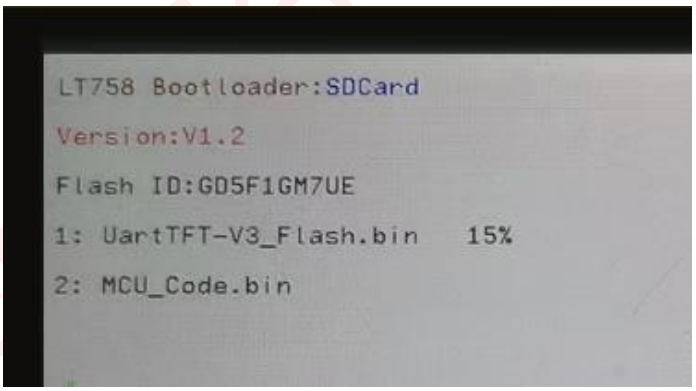


Figure 16-3：Programming UartTFT-V3_Flash.bin

Note: When programming UartTFT-V3_Flash.bin, it usually takes a long time to go through the [erase] and [write] operations because of the characteristics of the SPI Flash.

A CRC checking will be proceeded after the UartTFT-V3_Flash.bin is programmed

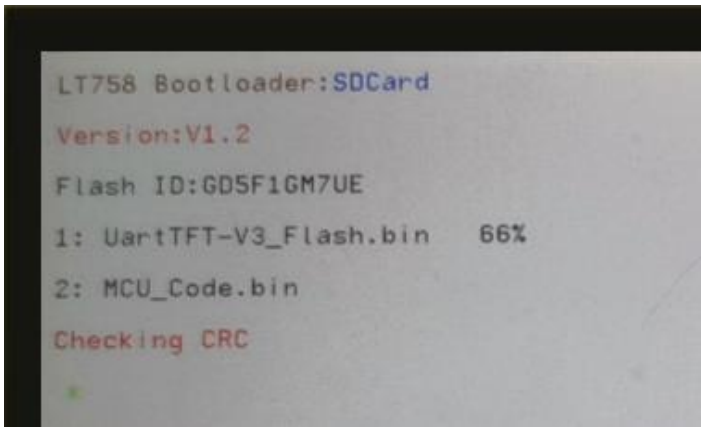


Figure 16-4: CRC Checking

As soon as the CRC checking is passed, the development board will enter the main program.

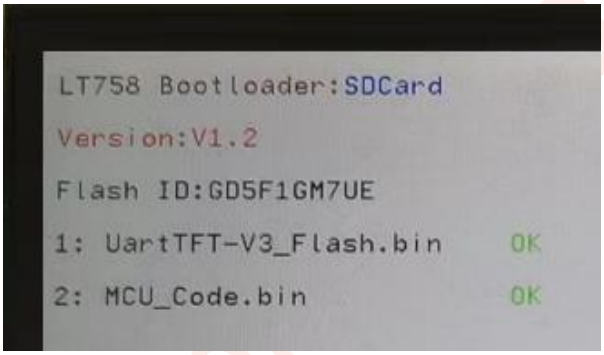


Figure 16-5: Programming Done

16.1.2.3. Download bin files through Uart port

LT_Uart_GUI is a tool that enables developers to upgrade MCU_Code.bin and UartTFT-V3_Flash.bin through Uart interface.

Once an MCU_Code is successfully programmed to the LT7589 demo board, the [BootMode] will be skipped when the MCU code is running. Users may enter the [BootMode] again by the below methods:

1. Send the below command to LT7589 through the Uart port:

   0x5A 0xA5 0x07 0x10 0x70 0x11 0xAA 0x55 0x11 0x99

2. Use the tool, LT_Uart_GUI_Vxxx.exe, to connect with LT7589, and then click on [BootMode]

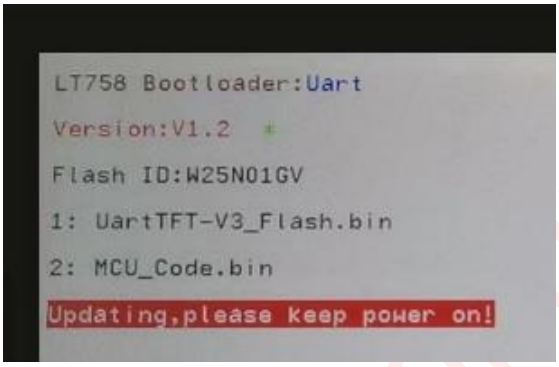


Figure 16-6: Upgrade through the Uart port

To program bin files through the Uart port, please follow below procedure:

6. Activate LT_Uart_GUI_Vxxx.exe (Download it at https://www.levetop.cn/en/index.aspx).

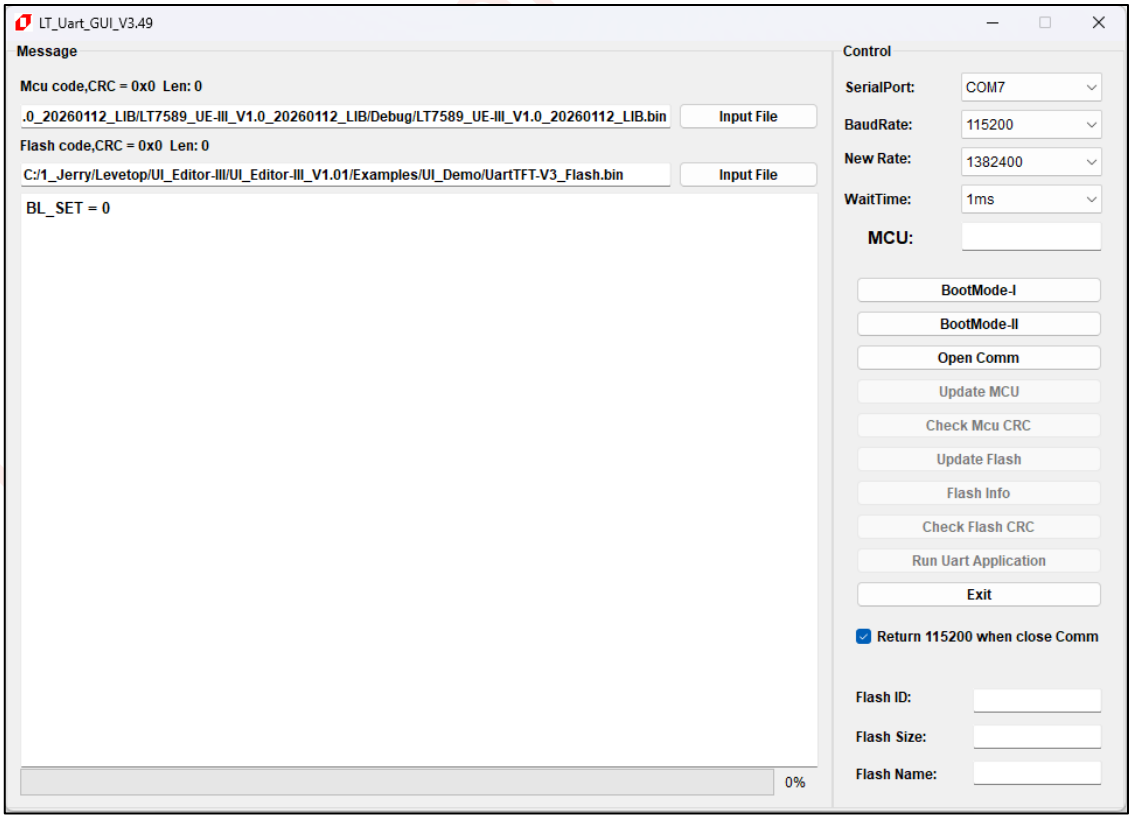


Figure 16-7: Activate LT_Uart_GUI_Vxxxx.exe

7. The software will auto-detect the LT7589 demo board and display the connected COM port. Click on [Open Comm], the MCU type and Bootloader version will be listed, as shown below:

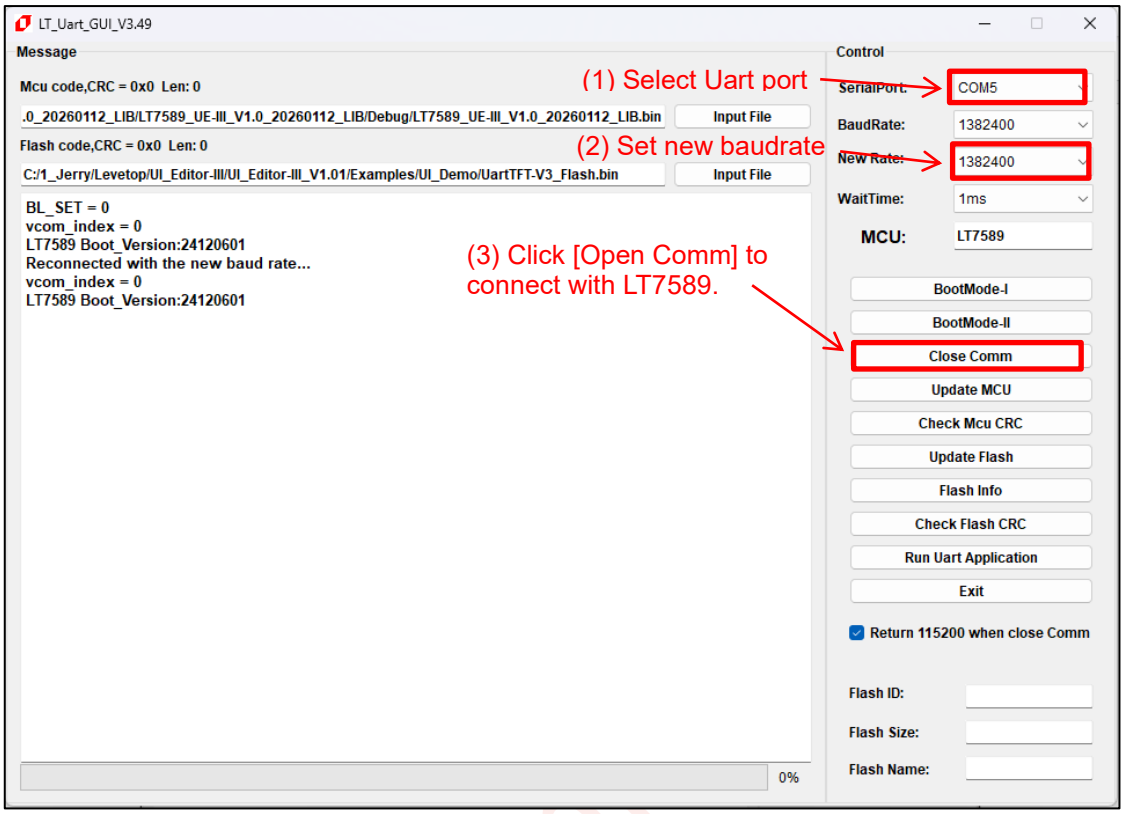


Figure 16-8: Click [Open Comm]

(3) Select the bin files and program them to the LT7589 demo board

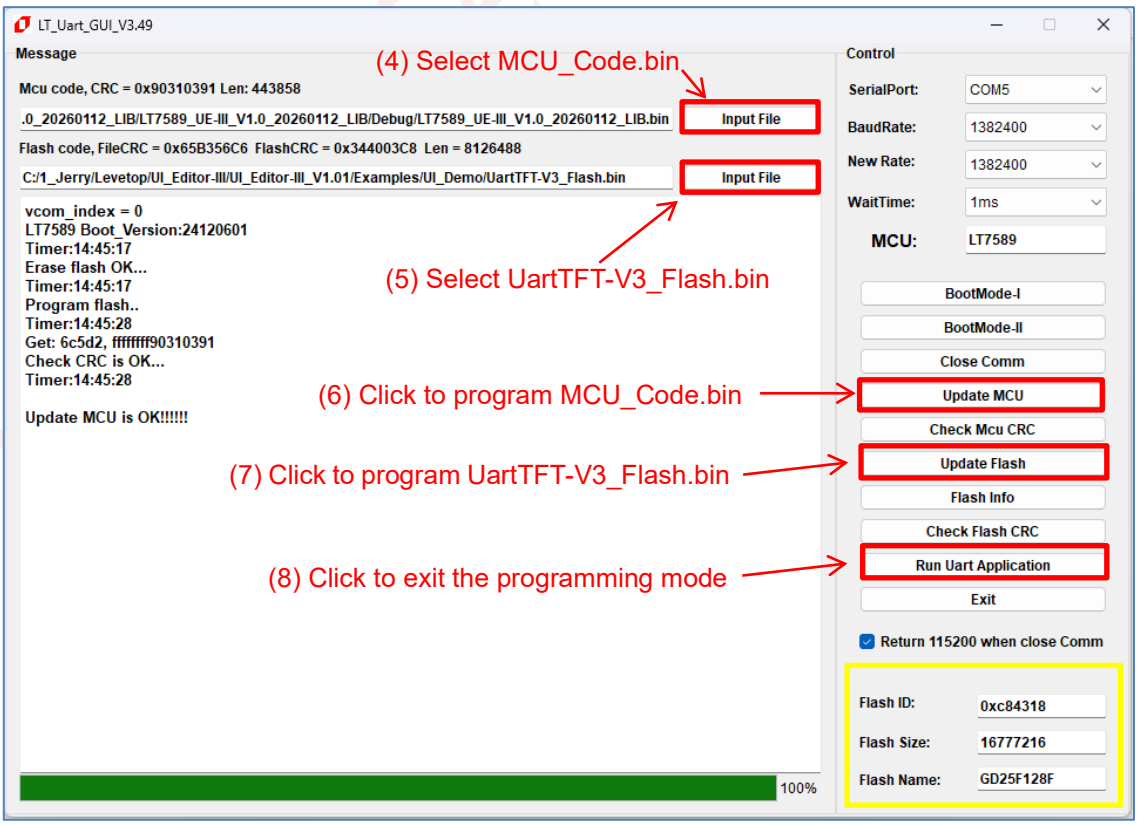


Figure 16-9: Select bin files and start programming

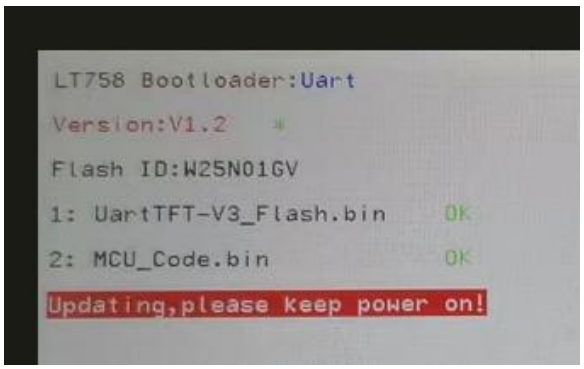


Figure 16-10：Programming Done

Note:

If clicking on [Flash Info], yet the flash information does not show up (as shown in the yellow box of Figure 16-9), then the Flash ID should be added to Flash.ini, as shown in Figure 16-11 and 16-12.

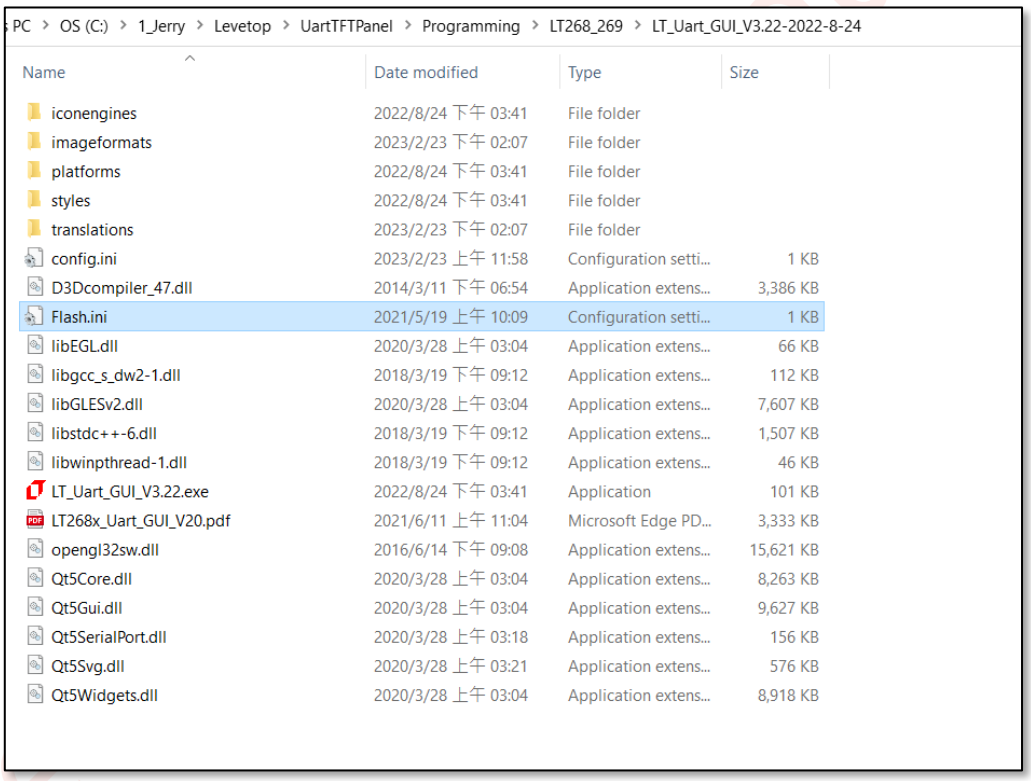


Figure 16-11：Add Flash ID to Flash.ini (1)

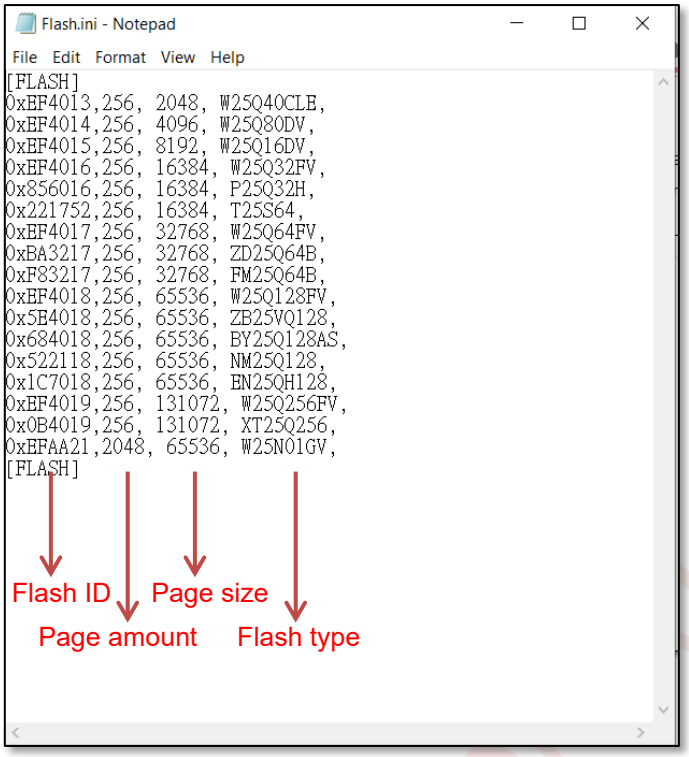


Figure 16-12：Add Flash ID to Flash.ini (2)

Note: Using Uart connection to download bin files is relatively slow, it is suggested that users utilize other programming methods.

16.1.2.4. Download bin files through USB port

LT_Uart_GUI can also enable developers to upgrade MCU_Code.bin and UartTFT-V3_Flash.bin through USB port.

Upgrade bin files through USB port is easy. Please follow the procedure shown below:

8. Hardware configuration:
    1. Connect Busy to GND
    2. Connect CON9 to the computer
    3. The COM port must not be occupied by other program/software

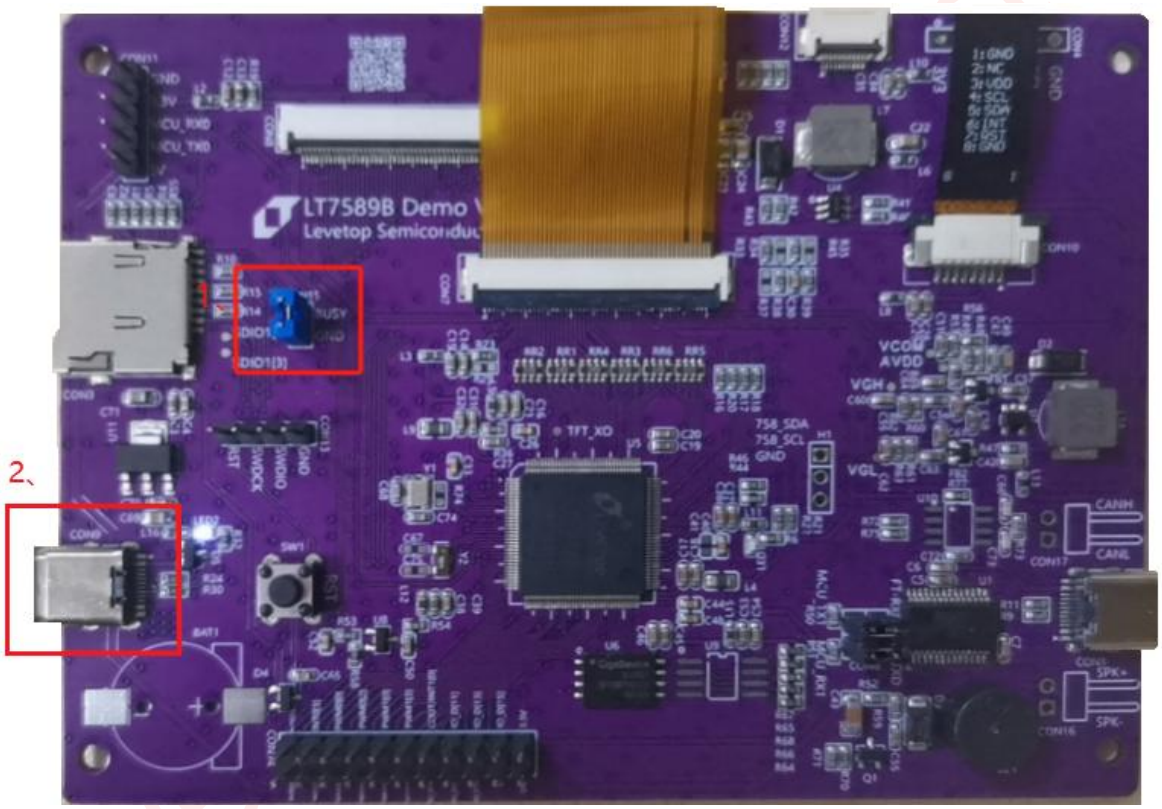


Figure 16-13: Prepare the LT7589 Demo Board

9. Activate LT_Uart_GUI_Vxxx.exe. The The software will auto-detect the LT7589 demo board and display the connected COM port. Click on [Open Comm], the MCU type and Bootloader version will be listed, as shown

below:

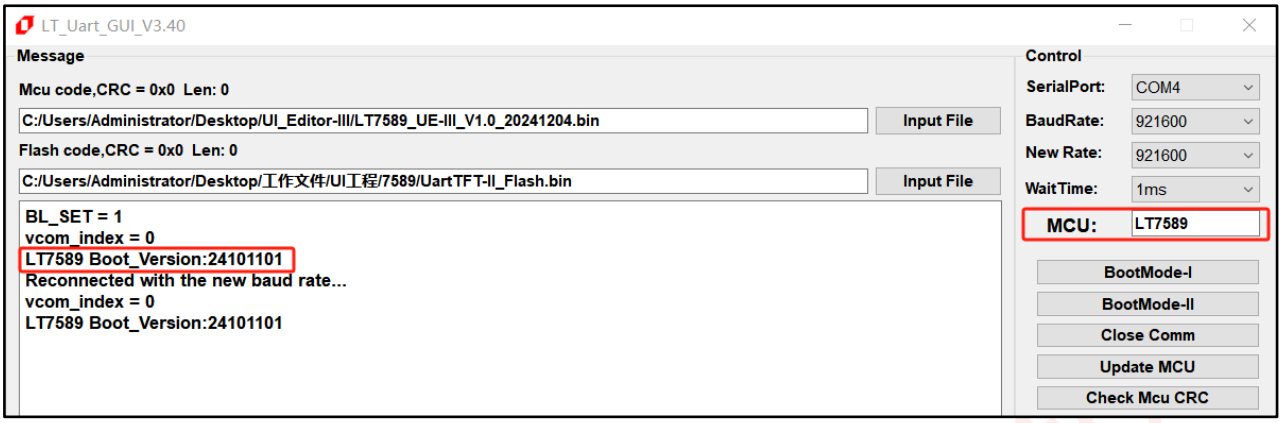


Figure 16-14: Click on [Open Comm]

(3) Follow the procedure illustrated in the *Download bin file through Uart Port* to program MCU_Code.bin and UartTFT-V3_Flash.bin.

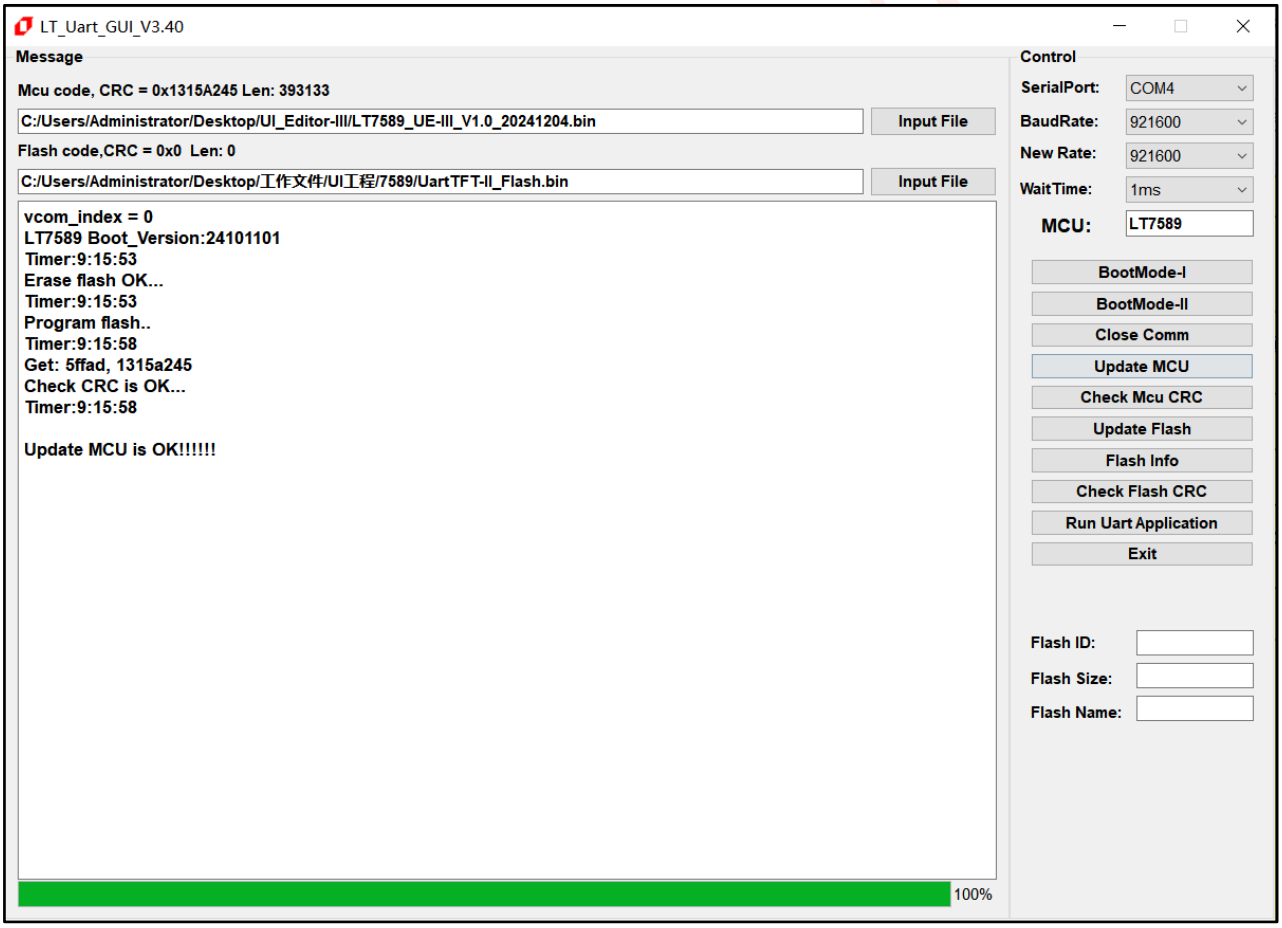


Figure 16-15: Program MCU_Code.bin and UartTFT-II_Flash.bin

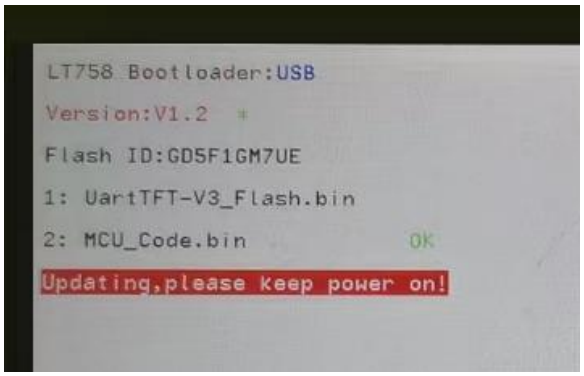


Figure 16-16: Programming Done

16.1.3. List of Prompt Messages

Figure 16-17 shows the list of prompt messages about the upgrading process.

```
/* File function return code (FRESULT) */

/* None.0 - None.19*/
/* (0) Succeeded */
/* (1) A hard error occurred in the low level disk I/O layer */
/* (2) Assertion failed */
/* (3) The physical drive cannot work */
/* (4) Could not find the file */
/* (5) Could not find the path */
/* (6) The path name format is invalid */
/* (7) Access denied due to prohibited access or directory full */
/* (8) Access denied due to prohibited access */
/* (9) The file/directory object is invalid */
/* (10) The physical drive is write protected */
/* (11) The logical drive number is invalid */
/* (12) The volume has no work area */
/* (13) There is no valid FAT volume */
/* (14) The f_mkfs() aborted due to any problem */
/* (15) Could not get a grant to access the volume within defined period */
/* (16) The operation is rejected according to the file sharing policy */
/* (17) LFN working buffer could not be allocated */
/* (18) Number of open files > FF_FS_LOCK */
/* (19) Given parameter is invalid */
```

Figure 16-17: List of Prompt Messages

16.2. Appendix 2 – Applicable IC Models & the Settings

16.2.1. Applicable IC Models

Developers may select their target IC through [MCU type] list in the [Project Setting] Page. Currently, UI_Editor-III only supports LT7589.

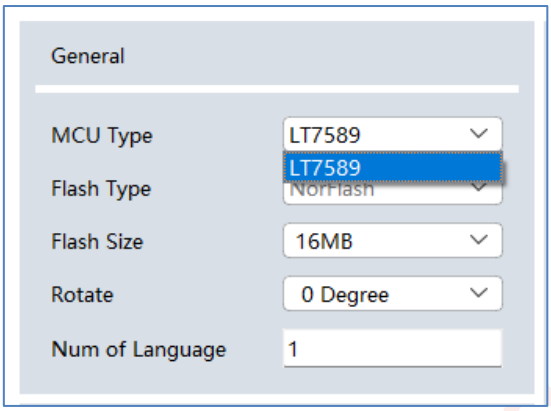


Figure 16-18: IC Options

16.2.2. Setting Limits

The setting limits based on the IC models are listed in Table 16-2, as shown below:

Table 16-2：IC models and the setting limits

IC Model / Parameter	LT7589
User address range	0x0000~0x7FFF
PNG size limitation	No limitation
Circular touch/progress bar	W=H<=Y resolution of the panel
Analog Clock	W=H<=Y resolution of the panel
Trend graph area	No limitation
RGB format	aRGB4444 aRGB8888
Picture size limit for Keyboard widget	No limitation
Area limits for SlideMenu widget	No limitation
Picture size limit for SlideMenu widget	W*H<=384000
Slide to jump – with sliding effects	Support
Popupbox background dimming	Support
Page Picture Compression	NA
Icon & Gif Compression	NA
Encoder	Support

Note: W and H represent the LCD resolution.

16.2.3. Maximum Number of Widgets in a Single Page

The number of widgets in a single page is limited. The IC resources occupied by different widgets vary too. In order to best utilize IC resources, the number of widgets in a single page is limited, based on the IC models. The following table lists all the widgets and their maximum amount allowed in a single page:

Table 16-3: Maximum Number of Widgets in a Single Page

IC Model / Widget Name	UI_Editor-III	LT7589
Button	128	20
Slidemenu	64	8
Popupbox	64	8
Variable Button	256	20
Combo Button	128	20
Circular touch	32	4
Slider Bar	32	4
SingleKey	200	60
Numeric Keypad	128	20
EN_KeyBoard	128	10
CN_KeyBoard	128	10
String_Label	512	300
Static_Text	512	300
Text Scroll	32	4
Text Number Display	512	30
Graphics Number Display	32	30
Analog Clock	8	2
Digital Clock	16	15
Gif	32	20
QRCode	16	16
Audio Play	32	1
Bit Status	128	64
Icon	128	64
Trend Graph	12	8
Encoder	8	1
Timer	8	10
Automatic variable	64	10
needle	16	6
State Button	128	20
Extend Button	128	20

16.2.4. The address list of variables and special registers.

Table 16-4: The address list of variables and special registers.

IC Model / Widget Name	LT7589
User Address range (writeAddr)	0x0000~ 0x7FFF
Page Register	0x7000
Backlight Register	0x7001
Time Register	0x7002~0x7007
Confirm_Time Register	0x7008
Wav Control Register	0x700A
Volume Register	0x700B
RTP Calibration Register	0x700C
Key code trigger Register	0x700D
Auto Backlight Control Register	0x700E
Register for setting the dimming value	0x700F
Register for setting the wait-time to enter sleep mode	0x7010
Register for setting the Uart upgrade mode	0x7011
Register for Variable Association List	0x7014
Register for setting multi-language	0x703F

16.2.5. Precaution: NandFlash

1. The storage space of NandFlash cannot be fully utilized, and each IC must reserve at least 20Blocks as a replacement for bad blocks (each block size is 128KBytes).
2. NandFlash with a size of less than 1GBytes needs to reserve 20Blocks for bad block replacement, 2GByte NandFlash needs to reserve 40Blocks for bad block replacement, and 4GByte NandFlash needs to reserve 80Blocks for bad block replacement.

For example, for a 128MBytes NandFlash, if 20Blocks are reserved for bad block replacement, the actual available space is 131596288 bytes, which is 128 × 1024 × 1024 - 20 × 128 × 1024. (i.e. 125.5MBytes)

16.3. Appendix 3 – Demo Kit

Levetop provides various development boards for each UartTFT controller. Below is an LT7589 development board for 40 pin or 50 pin standard TFT panels

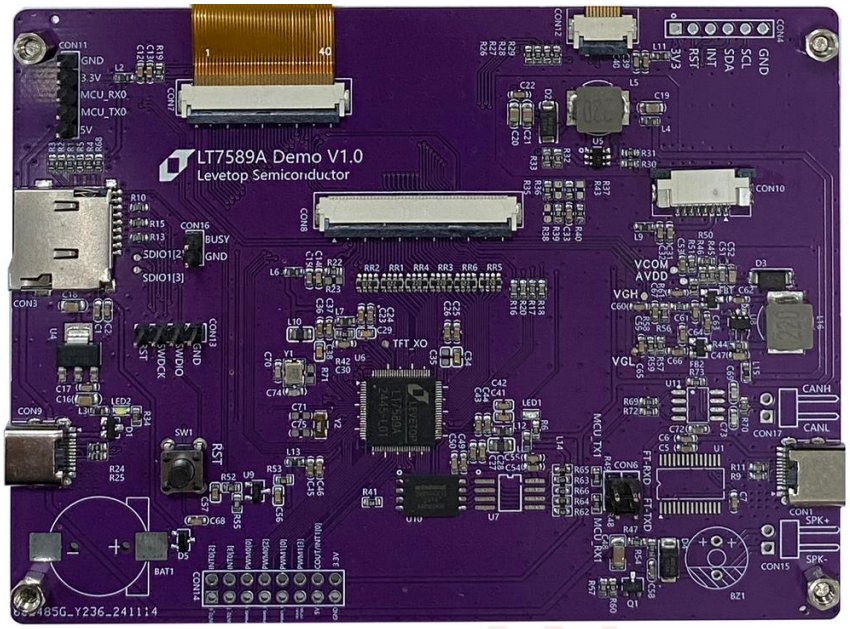


Figure 16-19：LT7589A Development Board（Demo Kit）

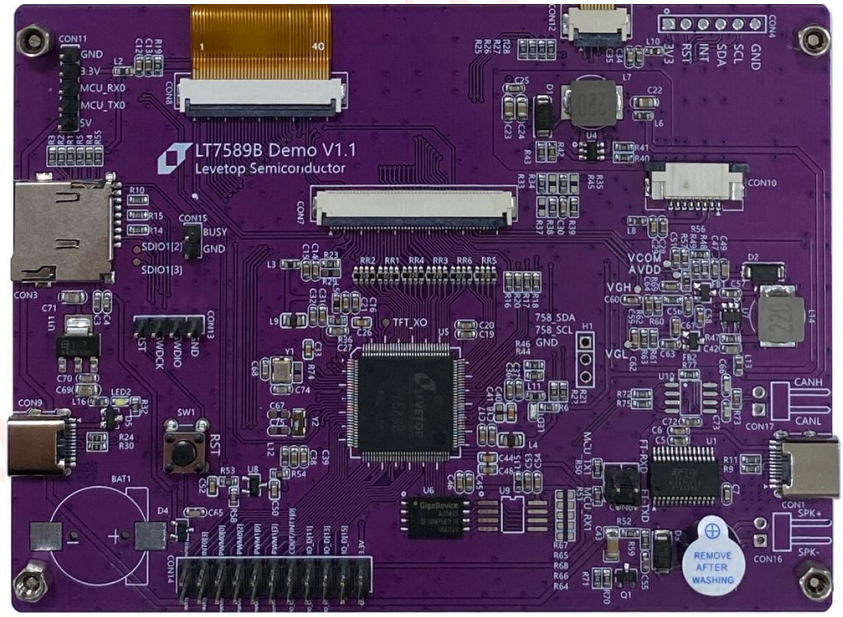


Figure 16-20：LT7589B Development（Demo Kit）

LT7589 development board has embedded an MCU program. There is also a set of demo materials which include picture, GIF animation, fonts, and wav data, pre-programmed to the SPI flash on board in the form of BIN file.

The related demo files can be downloaded through https://www.levetop.cn/en/index.aspx. Users may also contact Levetop sales for related information.

16.4. Appendix 4 – UartTFT Panel Development Flow

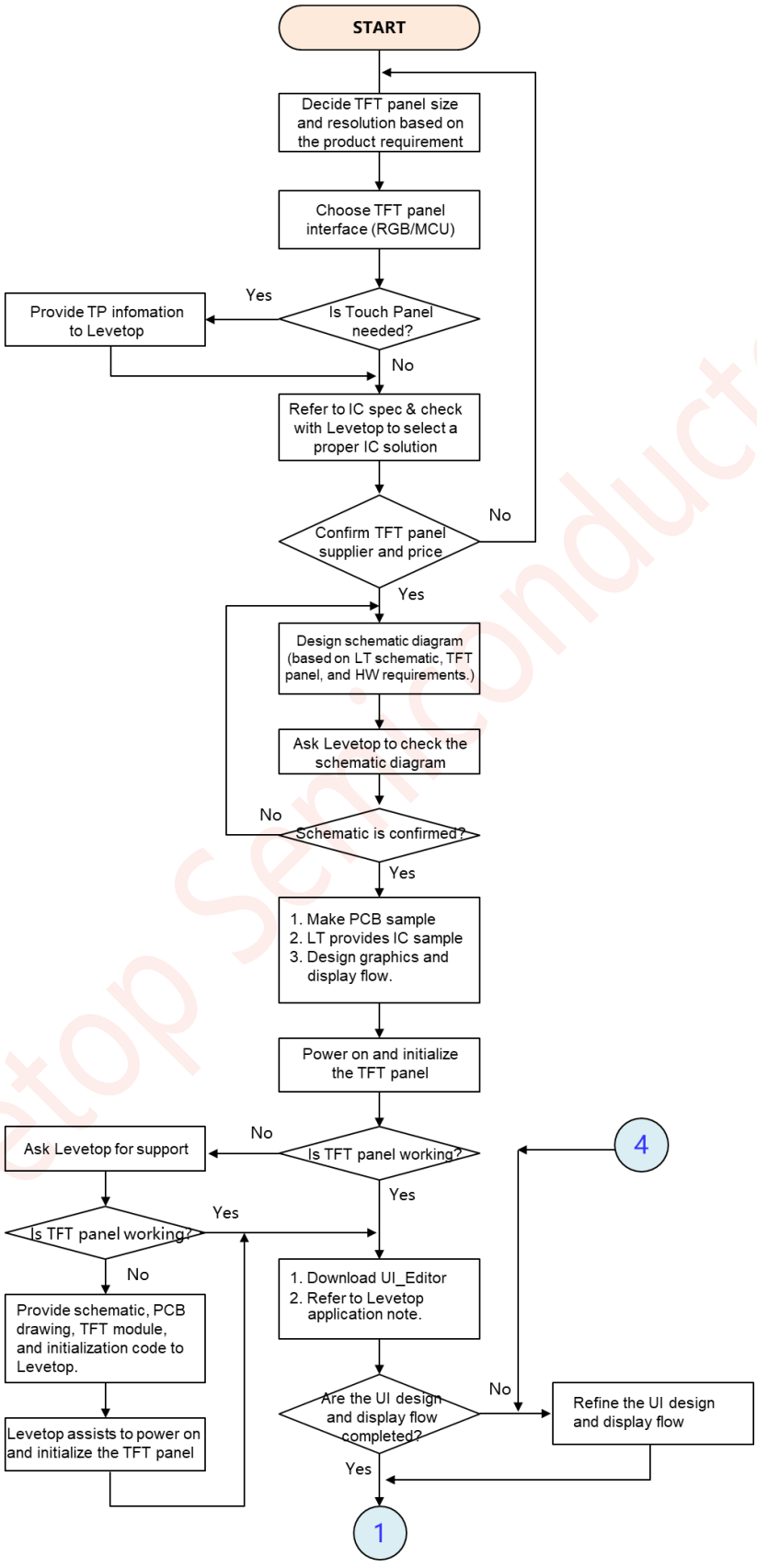


Figure 16-21: Development Flow (1)

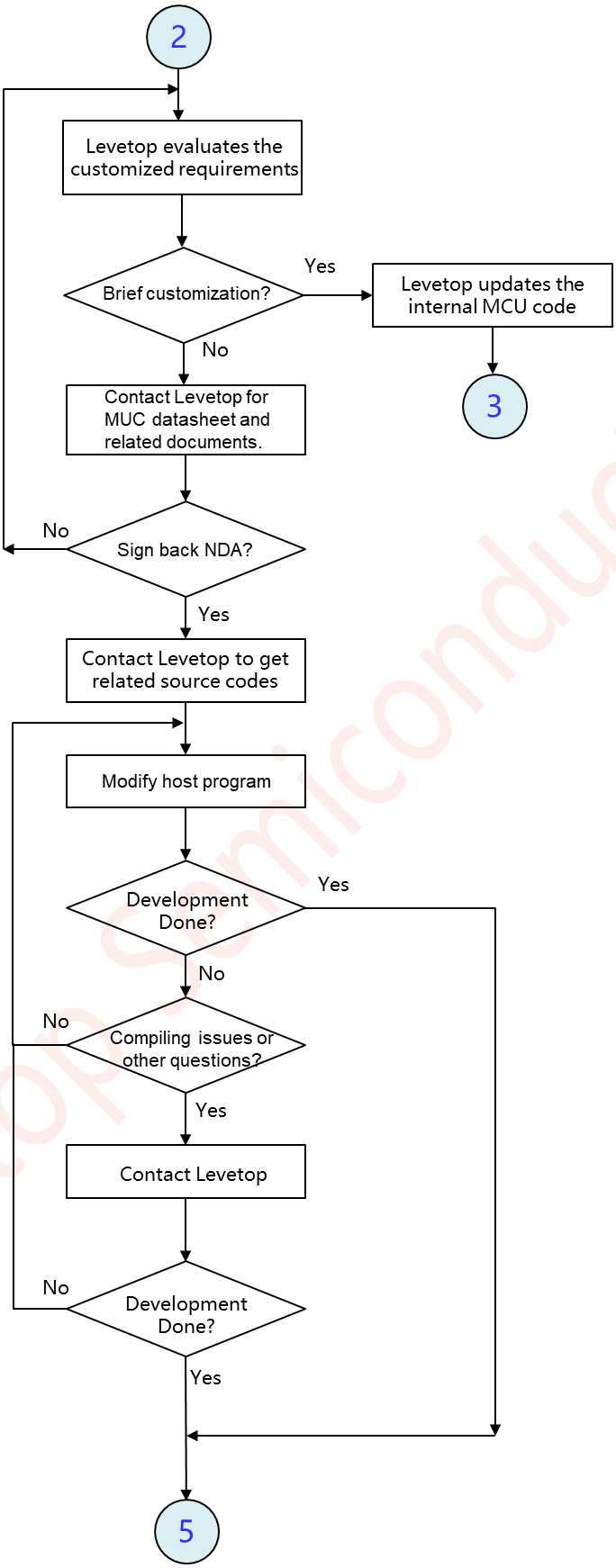


Figure 16-22: Development Flow (2)

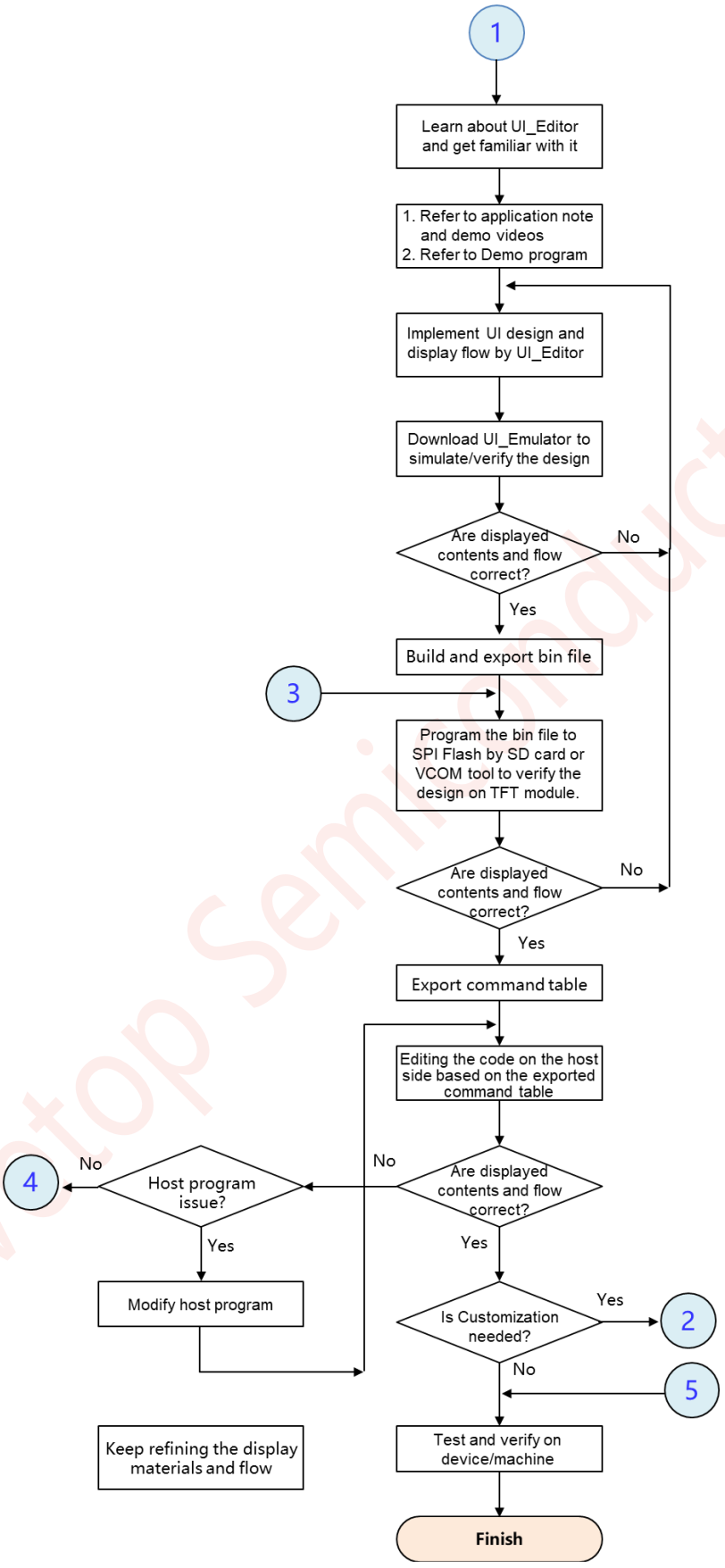


Figure 16-23：Development Flow（3）

16.5. Appendix 5 – UI_Editor vs. UI_Editor-III

The first generation of UI_Editor manipulates UartTFT controllers by fixed commands; whereas, the 3rd generation is designed to utilize variables for more sophisticated operations. The differences between the two generations are listed below:

Table 16-5: The differences in Uart Communication and Commands

Mode	UI_Editor	UI_Editor-III
Communication	Command structure: Header + command code + CRC + ending code	Command structure: Header + length + RD/WR command code + Variable address & Data + CRC Header can be modified. Support Modbus, I2C
Command	Every function corresponds to a command. The number of each function is limited by the serial number (Max. 256)	Every command is operated according to the variables. The number of each function is not limited.

16.6. Appendix 6 – New Features of UI_Editor-III

Table 16-6: New Features of UI_Editor-III

No.	Function	Description
1	Pop-up Box	● Command：Pop-up Box command + Trigger key
2	Input Numbers/Texts	● Numerical and ASCII text keyboard are available for input ● ASCII inputs can be displayed as, *，and the actual input data will be sent to Host MCU for further operation
3	Backlight Control	● Backlight control by pre-designed progress bar or circular progress bar ● Backlight control by the command from Host MCU ● Backlight control by pop-up box
4	Pop-up box settings	● Command: Pop-up Box command + multi-variables control ● Display the listed items by pop-up box, click to choose an item for further operation ● Support “Menu selection“ feature, listed items can be selected by sliding operation
5	Adjust Date/Time	● Sliding the panel to setup Date and time
6	Display data synchronized with the progress bar	● Display options: %, numbers, and symbols
7	Icon switching	● Icon switching in loop by incremental control

16.7. Appendix 7 – OTA Difference Upgrading

16.7.1. Overview

In order to improve the slow upgrade speed when using serial ports for OTA upgrades, Levetop has designed a differential upgrade method. This upgrade method will allocate storage ranges for each type of material in the UI project during compilation, and users can set reserved upgrade space for each type of material in the UI project setting menu based on their needs, on the basis of the storage range specified for existing materials. In the later stage of development, after the UI materials are changed, deleted, or replaced, simply recompile the new UI project and generate a bin file, and then import it together with the old bin file of the UI project into the differential comparison software, Diff_OTA_Tool, provided by Levetop. This software will compare and analyze the differences between the new and old bin files, and generates a file with a ".diff " suffix for the data of the differences. Since the file only contains the differential data between two bin files, the data amount is relatively small. Users may update the diff file to the SPI flash without updating the whole project data. The obtained diff file can be updated to the SPI flash by the software tool, LT_Uart_GUI_Vx.xx (note that version 3.42 or above is required). The diff file can also be updated to the SPI flash by the host MCU through WIFI, Bluetooth, or Uart port.

16.7.2. OTA Difference Upgrading Steps

Step 1: Set up the "OTA Update Addr" Table in UI_Editor

Step 2: Modify the UI project as needed, and then compile it in UI_Editor to generate a new bin file.

Step 3: Use the software, "Diff_OTA_Tool_Vx.xx", to generate a diff file based on the difference between the new bin file and the old one.

Step 4: Update the diff file to the SPI Flash

16.7.3. OTA Update Address Table

1. To set up the OTA Update Address Table in UI_Editor, click on [File] and then select [OTA Update Addr Setting]

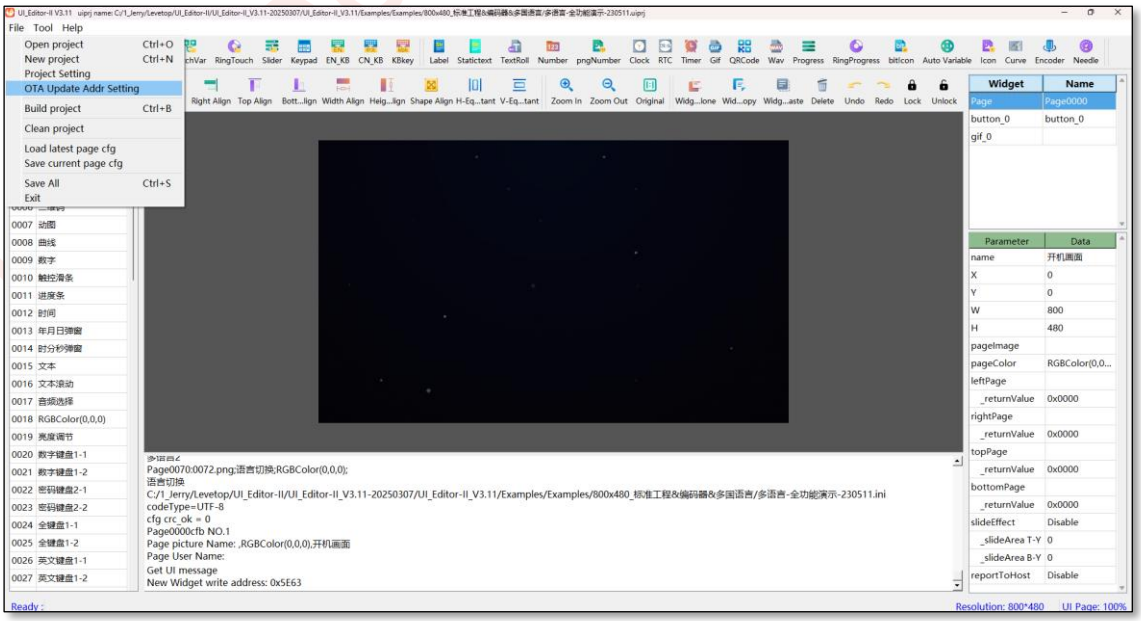


Figure 16-24: OTA Update Address Setting

2. In the pop-up window, check the [OTA Update Addr] box on the left-top corner. As shown in Figure 16-50, the first column is the start address of each kind of marterials; the second column is the data length each kind of materials; the third column is the reserved data length for each kind of materials. The first and second column data will be auto-generated by UI_Editor after compilation. The value in the third column can be set by users, and the default value is 1024K. It is suggested that users set the value of the third column after the UI project is almost done. This is because the project structure is more stable now, and it is eaiser to decide the reserve spaces for each kind of materials.

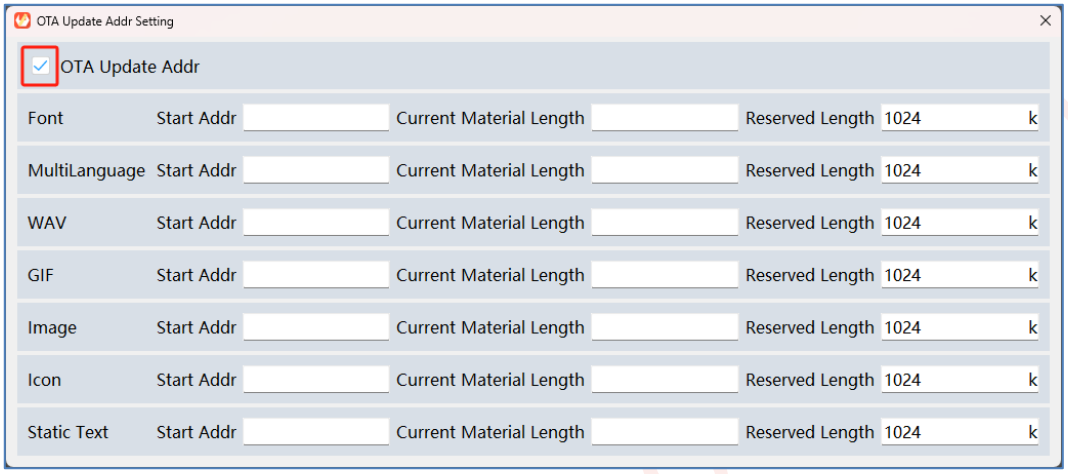


Figure 16-25: OTA Update Addr Setting Table

3. After compilation, the values of the first and second colum will be auto-generated. Users may then decide the reserve spaces for each kind of materials and set the values of the third column, as shown in Figure 16-51. Note that the setting value must be a multiple of 128K (or 256K if [with GBKCode] is checked in the Project Setting page). If the input value is not a multiple of 128K / 256K, a proper value will be auto-adjusted by UI_Editor. After the values of the third column are set, the UI project needs to be compiled again to include the settings. The generated bin file can then be programmed to the SPI Flash.

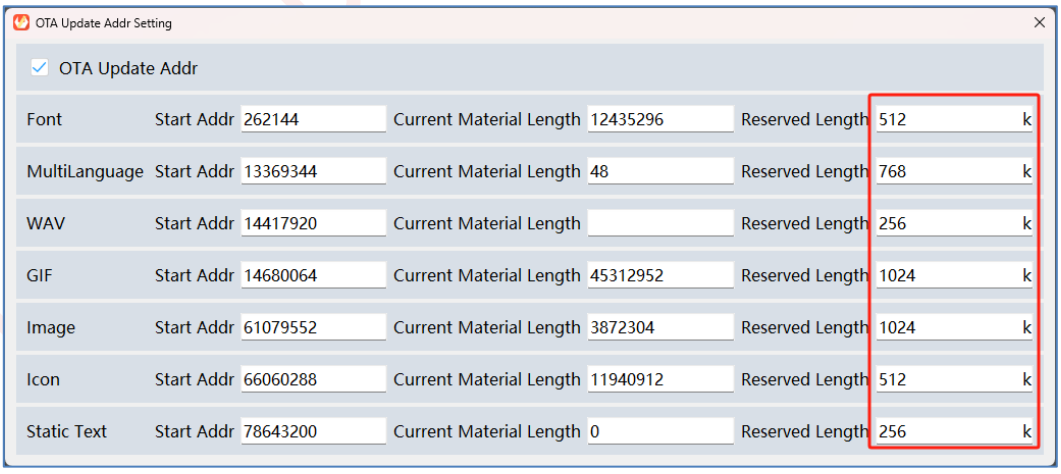


Figure 16-26: Setup the Reserve Spaces

4. When a UI project is modified, there could be changes on the widget location, widget address, widget amounts, and materials (added / deleted / replaced). The information of widgets' location address, and amounts is stored in the first 256K of the generated bin file. If the widget information is changed, the 256K data will be changed as well and be recorded in the diff file. As for the changes on materials, they will be recorded in the reserve spaces set in the previous step. Note that the space occupied by the increased materials cannot exceed the reserve space. In addition, the numbering of the increased materials can only be larger than that of the existed materials. Setting a front or middle number will cause processing errors. When the UI project is done modification and compiled, the generated bin file can be used for OTA difference upgrading. (Note the [OTA Update Addr] box has to be checked, as shown in Figure 16-50.)
5. When the OTA difference upgrading function is enabled, the generated bin files (old and new ones) will be in the same size. This feature can also be used to check if the increased materials exceed the reserve space.

16.7.4. Diff_OTA_Tool

16.7.4.1. Data Structure of the Diff file

10. The data structure of a diff file is as shown in Figure 16-27

Header (32bytes)
Diff Data Block 0
Diff Data Block 1
Diff Data Block 2
Diff Data Block n

Figure 16-27: Data Structure of a Diff File

11. Header (The listed data are in big-endian)

Header (32bytes)	byte0	0x44	Document ID: "DOTA"		Big-Endian
	byte1	0x4F			
	byte2	0x54			
	byte3	0x41			
	byte4		The data length of the Diff Block		
	byte5				
	byte6				
	byte7				
	byte8		The CRC value of the Diff Block		
	byte9				
	byte10				
	byte11				
	byte12		The data amount of the Diff Block		
	byte13				
	byte14				
	byte15				
	byte16		The length of the original document		
	byte17				
	byte18				
	byte19				
	byte20		The CRC value of the upgrading document		
	byte21				
	byte22				
	byte23				
	byte24		The CRC value of the UI data of the upgrading document		
	byte25				
	byte26				
	byte27				
	byte28		The CRC value of the front 28bytes of the upgrading document		
	byte29				
	byte30				
	byte31				

Figure 16-28: Data Structure of the Header (32bytes)

12. Data structure of the block data

Block_0	byte0		The address of the Block data	
	byte1			
	byte2			
	byte3			
	byte4		The length of the Block data	
	byte5			
	byte6			
	byte7			
	byte8		The CRC value of the Block data	
	byte9			
	byte10			
	byte11			
	byte12		The one's complement of the CRC value of the Block data	
	byte13			
	byte14			
	byte15			
	DATA		Block data	

Figure 16-29: Data Structure of the Block Data (16bytes)

16.7.4.2. Software Operation

13. Activate Diff_OTA_Tool, and import the old bin file and the new bin file, as shown in Figure 16-30.

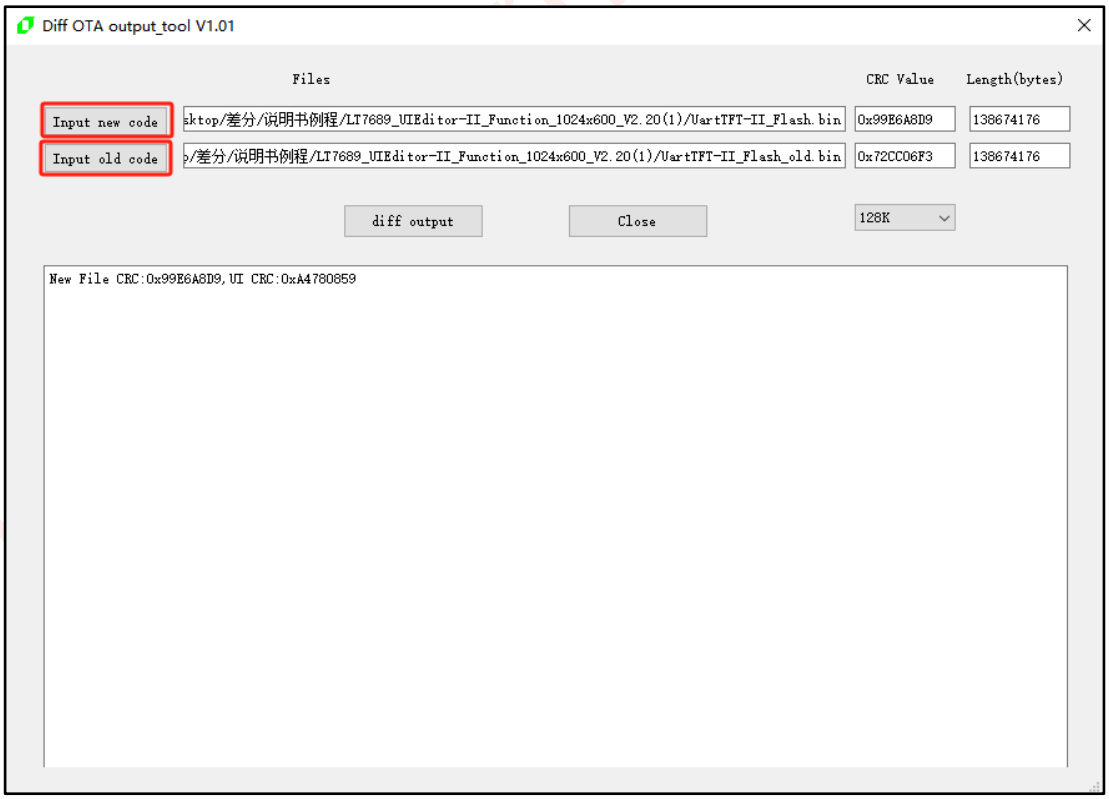


Figure 16-30: Import the old and new bin files

14. Click [diff output] to generate a file with a suffix of " .diff ", as shown in Figure 16-31.

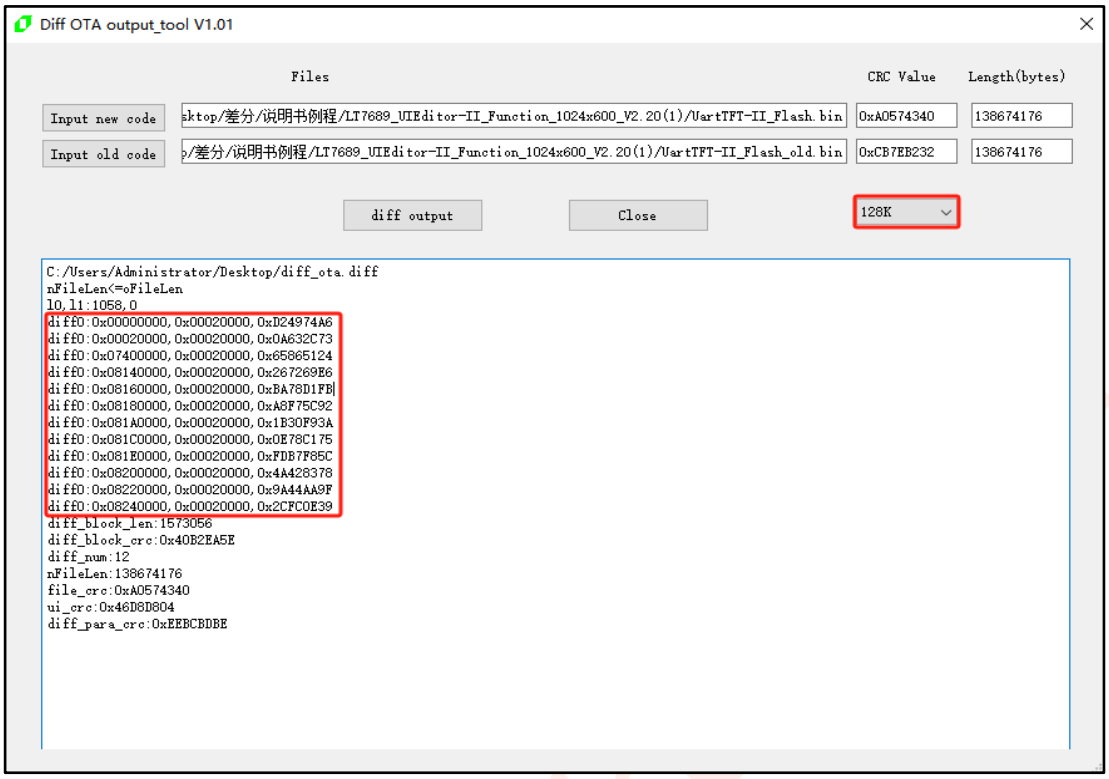


Figure 16-31: Generate the Diff File

As shown in Figure 16-56, when the upper right red box is set to 128K, the data of the old and new bin files are divided into packets every 128K, starting from address 0 for differential comparison. If there is a difference in the data of certain packet between the new and old files, it will be displayed in the message window, as the red box shown in the lower left corner of Figure 16-31.

For NandFlash, it is recommended to compare a packet of every 128K data; while for NorFlash, it is recommended to compare a packet of every 4K or 64K data.

15. When designating the " .diff " file name, do not use the existed file names in the file folder. Using the existed " .diff " file names may result in incorrect data.

16.7.5. Upgrading SPI Flash – LT_Uart_GUI

Once the " .diff " file is ready, users can program it to the SPI Flash through the software tool, LT_Uart_GUI. Note only the v3.42 or above version supports programming the " .diff " file. Users may refer to the programming procedure described in the previous section, *Appendix 1 - Programming*.

The below example explains how users may program a " .diff " file to the SPI Flash on a LT168B development board through the USB interface.

1. Hardware configuration (refer to Figure 16-57)

(1) Connect the Busy pin to the GND pin, as noted ❶

(2) Connect the board to PC through USB port, as noted ❷

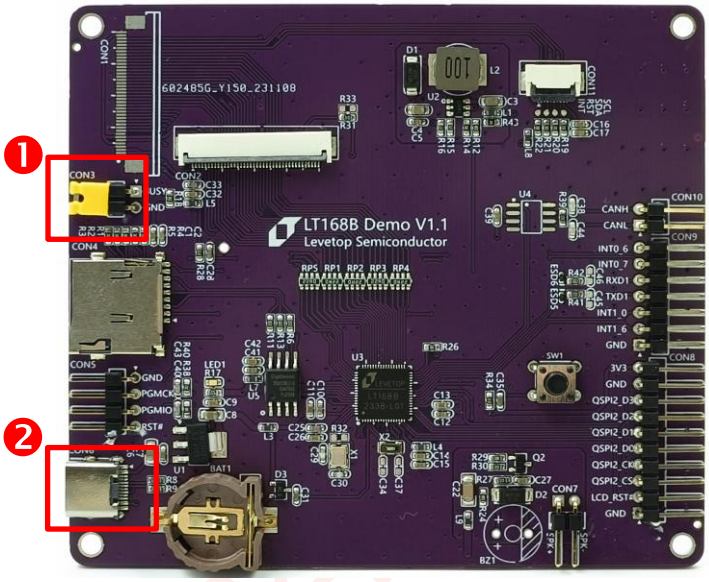


Figure 16-32: 168B Hardware Configuration

2. Activate LT_Uart_GUI, and click on [Input File] to import the " .diff " file. Next, select the connected port, and then click on [Open Comm] to connect the LT168B module.

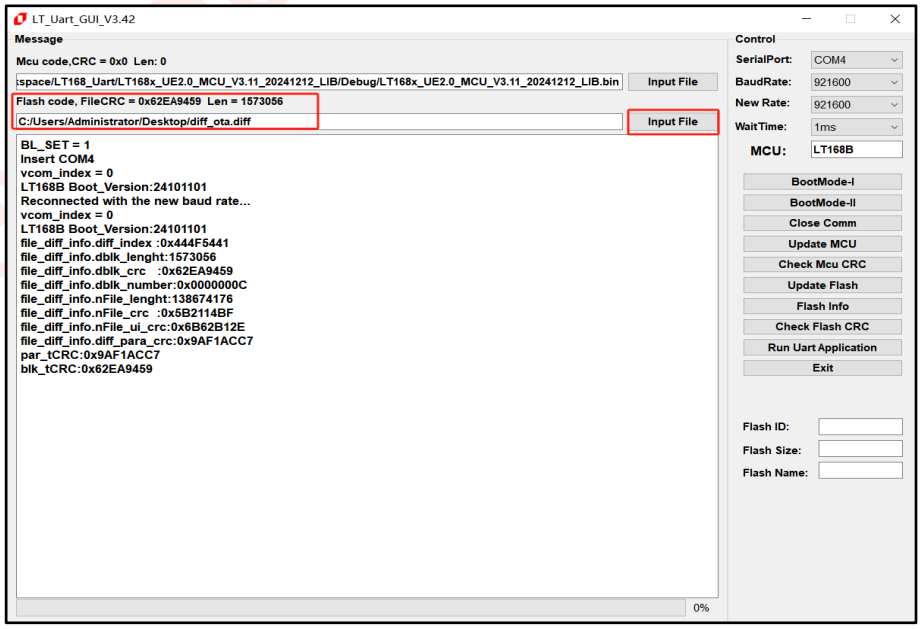


Figure 16-33: Import the .diff File

3. Click on [Update Flash] to start the programming.

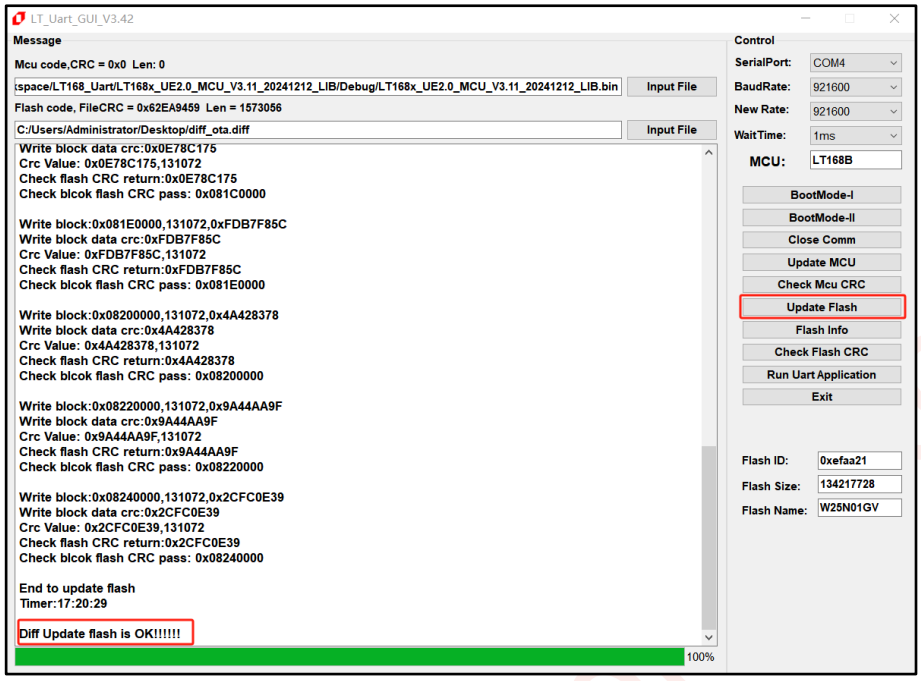


Figure 16-34: Start Programming